32-kbit/s ADPCM with the TMS32010

APPLICATION REPORT: SPRA131

Authors:

Jay Reimer
Digital Signal Processing – Semiconductor Group

Mike McMahan
Corporate Engineering Center

Masud Arjmand
Central Research Laboratories

Digital Signal Processing Solutions
1989

TEXAS INSTRUMENTS

TRADEMARKS

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

32-kbit/s ADPCM with the TMS32010

Abstract

This report discusses 32-kbit/s Adaptive Differential Pulse Code Modulation (ADPCM) transcoders. A half-duplex ADPCM transcoder, which complies with the CCITT recommendation (G.721), can be achieved with a single TMS32010. If the transcoder is used only for private lines, a full duplex non-CCITT ADPCM transcoder is more cost-effective and can be designed with a single TMS32010 processor. Both the CCITT and non-CCITT algorithms and code implementations are covered in the report.

Product Support on the World Wide Web

INTRODUCTION

Digital voice communication is typically transmitted in a 64-kbit/s PCM bit stream. Voice and data communications demand increasing capacities for signal transmission without significant degradation in the quality of the transmitted signal. One of the recommended solutions for accomplishing this task is that of Adaptative Differential Pulse Code Modulation (ADPCM). This solution has been reviewed by CCITT (International Telegraph and Telephone Consultative Committee), and a specific standard* has been recommended. Two solutions, a full-duplex solution and a half-duplex solution, are discussed in this application report. Both follow the model recommended by CCITT for 32-kbit/s ADPCM, although only the half-duplex solution provides a bit-for-bit compatible data stream as required by the recommendation. At 32 kbit/s, the ADPCM solution provides double the channel capacity of the current 64-kbit/s PCM technique. Each solution has been totally incorporated in the internal memory space of the Texas Instruments TMS32010 microprocessor.

This application report presents a brief review of the basic principles of PCM and ADPCM. Hardware requirements, software logic flow, and key features of the TMS32010 microprocessor for the implementation of ADPCM are also given. Source code is provided for the implementation and creation of an ADPCM transmission channel.

DIGITIZATION

Over the past 20 years, the telecommunications industry has changed from totally analog circuits to networks which integrate both analog and digital circuits. Digital signal encoding has the advantages of greater noise immunity, efficient regeneration, easy and effective encryption, and uniformity in transmitting voice and data signals. Increased bandwidth is required to transmit digital signals while maintaining a given analog signal quality at the receiver.

Voice store and forward systems have been changing from totally analog storage media, such as audio tape, to digitized storage which allows random access of stored data, but with the tradeoff of increased storage media requirements.

Signal quality begins with the digitization of the original analog signal. The process of digitization and coding introduces a distortion associated with the quantization of the digitized signal, as shown in Figure 1. This signal distortion or noise is different from the channel noise normally associated with a transmitted signal. After a signal has been digitized, the signal is much less susceptible to channel noise since the signal can be regenerated as well as amplified along the way, thus reducing the possibility of being corrupted by the transmission system. The overall quality of digital transmission is then limited by the digitization process in an error-free transmission system.

Figures 2 and 3 show general representations of a digital communication channel. The actual transmission (and storage of a digital waveform) uses an analog channel. The outside points of the communications channel are the transmitter and receiver, as shown in Figure 2. These are commonly combined in a single device known as a combo-CODEC (CODing and DECoding device). The codec supplies, on the coding or transmitting side, the necessary filtering to bandlimit the analog signal and avoid signal alias and A/D conversion. On the decoding or receiving side, the codec performs a D/A conversion and then interpolates or smooths the resultant signal.

Figure 3 shows the digitized signal modulated for transmission in the network and then demodulated at the receiving end to retrieve the transmitted digital signal.

PCM

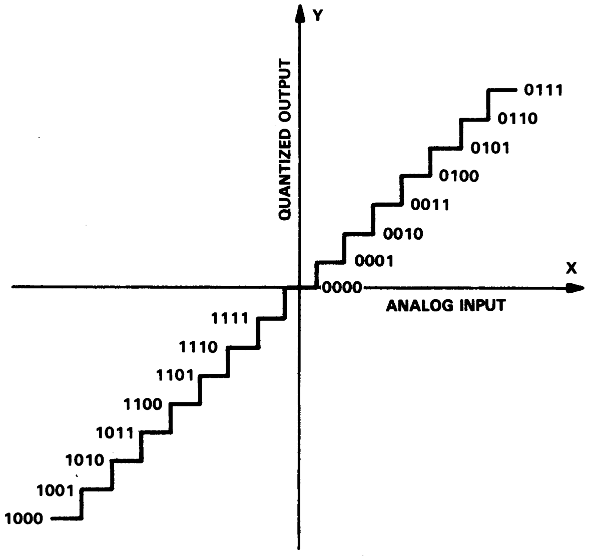
Digitization and coding of the analog signal at the transmitter can be performed in several ways. The complexity of the chosen method is related to availability of encoder memory and to the resultant delay in the encoding process.

When digital signal transmission is implemented, memory and the resultant delay dictate that a simple scheme, such as Pulse Code Modulation (PCM), be implemented. PCM codes each sampled analog value of the input waveform to a unique or discrete value. The digital quantization introduces distortion into the signal waveform, as shown in Figure 1.

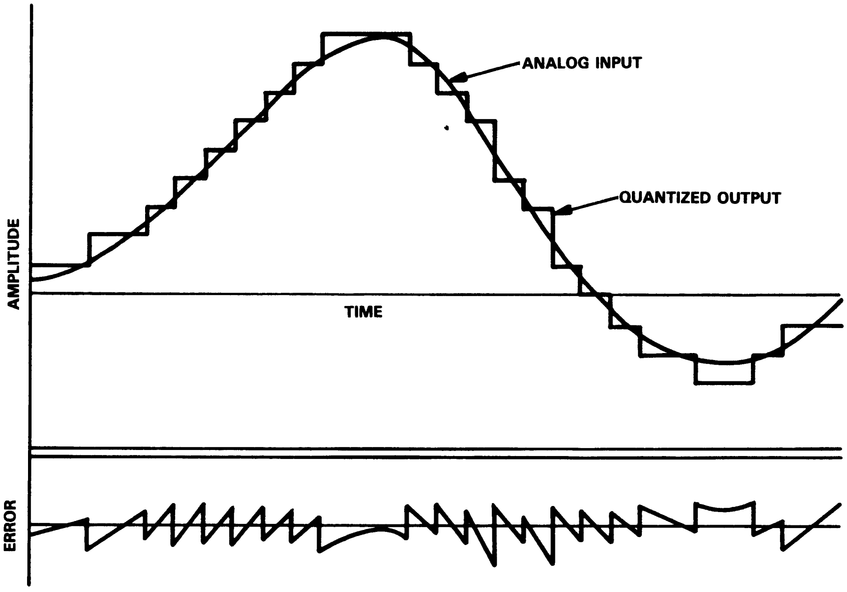
A nonuniform quantization scheme may be used to COMPAND (COMpress and exPAND) the signal in the waveform coding and decoding blocks in the system, generating log-PCM. By using larger quantization steps for large amplitude signals and smaller steps for small amplitude signals, efficient use is made of the data bits for digital transmission while maintaining specific signal-to-quantization noise thresholds. With the two current methods of COMPANDING (A-law and μ-law), the signal quality of a 13-bit digitized signal is maintained while transmitting only 8 bits per sample.

While quantizers remove the irrelevancy in a signal, coders remove the redundancy. In PCM encoding, each sample of the input waveform is independent of all previous samples; no encoder memory is required.

*Recommendation G.721, 32 kbits/s Adaptive Differential Pulse Code Modulation'', CCITT, 1984.



(a) SIGNAL QUANTIZATION



(b) SIGNAL QUANTIZATION ERROR

Figure 1. Quantization Errors in a Digitized Signal

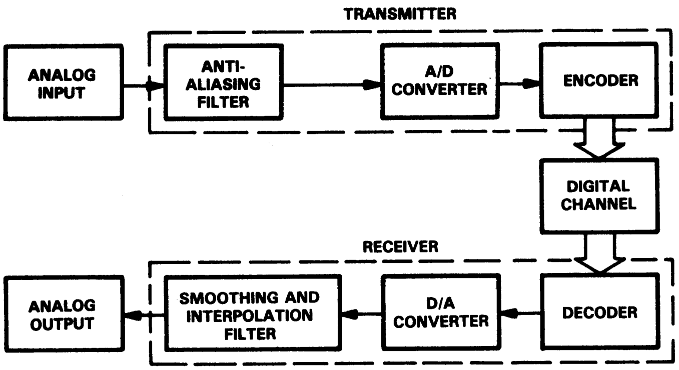


Figure 2. Digital Communication of Waveforms

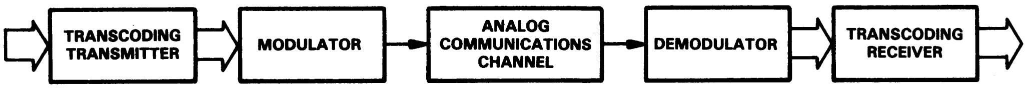


Figure 3. Digital Channel

ADPCM

Analysis of speech waveforms shows a high sample-to-sample correlation. By taking advantage of this property in speech signals, more efficient coding techniques have been designed to further reduce the transmission bit rate while preserving the overall signal quality.

APCM

Adaptive PCM (APCM) is a method that may be applied to both uniform and nonuniform quantizers. It adapts the stepsize of the coder as the signal changes. This accommodates amplitude variations in a speech signal between one speaker and the next, or even between voiced and unvoiced segments of a continuous signal. The adaptation may be instantaneous, taking place every few samples. Alternatively, it may occur over a longer period of time, taking advantage of more slowly varying features. This is known as syllabic adaptation.

The basic concept for an adaptive feedback system, APCM, is shown in Figure 4. An input signal, s(k), in the transmitter is quantized and coded to an output, I(k). This output is also processed by stepsize adaptation logic to create a signal, q(k), that adapts the stepsize in the quantizer. Correspondingly, in the receiver, the received signal, I(k), is processed by an inverse quantizer (i.e., decoded), producing the reconstructed signal, $s_r(k)$. Like the transmitter, the quantized signal, I(k), is processed by adaptation logic to create a stepsize control signal, q(k), for the inverse quantizer.

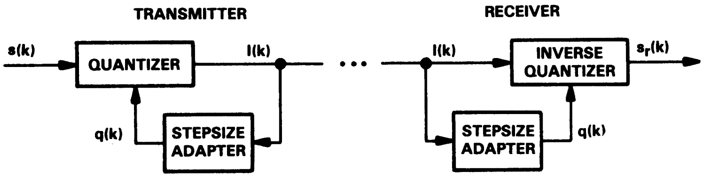


Figure 4. APCM Block Diagram

DPCM

The method of using the sample-to-sample redundancies in the signal is known as differential PCM (DPCM). The overall level of high correlation on a sample-by-sample basis indicates that the difference between adjacent samples produces a waveform with a much lower dynamic range. Correspondingly, an even lower variance can be expected between samples in the difference signal. A signal with a smaller dynamic range may be quantized to a specific signal-to-noise ratio with fewer bits.

A differential PCM system, DPCM, is shown in Figure 5. In Figure 5, the signal difference, d(k), is determined using a signal estimate, $s_e(k)$, rather than the actual previous sample. By using a signal estimate, $s_e(k)$, the transmitter uses the same information available to the receiver. Each successive coding actually compensates for the quantization error in the previous coding. In this way, the reconstructed signal, $s_r(k)$, can be prevented from drifting from the input signal, s(k), as a result of an accumulation of quantization errors. The reconstructed signal, $s_r(k)$, is formed by adding the quantized difference signal, $d_q(k)$, to the previous signal estimate, $s_e(k)$. The sum is the input to predictor logic which determines the next signal estimate. A decoding process is used in both the transmitter and receiver to determine the quantized difference signal, $d_q(k)$, from the transmitted signal, I(k).

ADPCM

ADPCM combines the features of both the APCM and DPCM systems. Figure 6 shows the basic blocks combining adaptation and differencing features in an ADPCM system. Both quantizer adaptation and signal differencing require the storage (in memory) of one or more samples in both the transmitter and receiver. Furthermore, the transmitter must use some method to ensure that the receiver is operating synchronously. This is accomplished by using only the transmitted signal, I(k), to determine stepsize adaptation in the quantizer and inverse quantizer and to predict the next signal estimate. In this way, the blocks in the receiver can be identical to those in the transmitter. Additionally, the specific adaptation techniques are designed to be convergent and thereby help provide quick recovery following transmission errors.

The ADPCM system, as used in digital telephony, is not an original signal coding system, but is actually a transcoder, converting between log-PCM and ADPCM codes. Currently there are a large number of systems using log-PCM for transmission. The ADPCM system incorporates both an adaptive quantizer and an adaptive predictor. The adaptive quantizer contains speed-control and scale-factor adaptation. A measure of the rate-of-change of the difference signal provides a means of determining the speed control. The scale factor adjustments to the difference signal adapt the fit of the quantization levels to minimize the signal-to-noise ratio. With speed control, the system can take advantage of both the instantaneous and syllabic adaptation rates, thereby adapting better to both speech and data signals. In the adaptive predictor, the prediction filter coefficients are updated by a gradient algorithm. Predictor adaptation improves the performance of the predictor for nonstationary signals (e.g., speech).

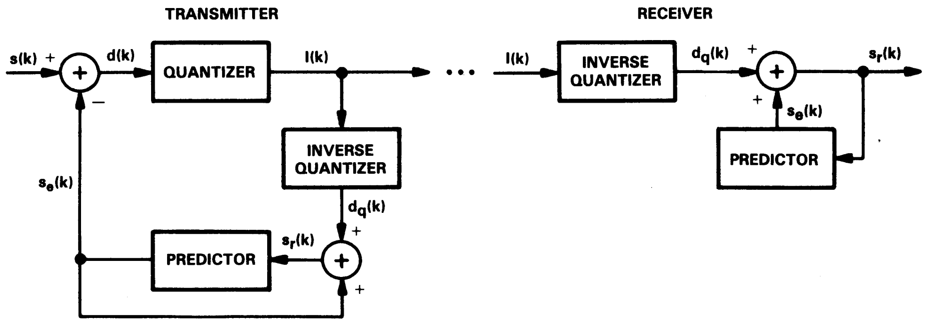


Figure 5. DPCM Block Diagram

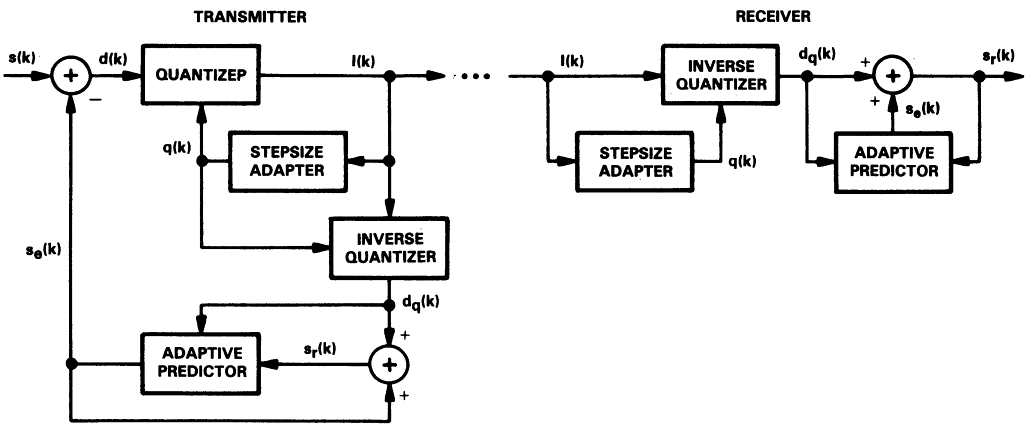


Figure 6. ADPCM Block Diagram

THE ADPCM ALGORITHM

The ADPCM algorithm has a receiver imbedded in the transmitter. This is important since, if the signal feedback used to determine the signal estimate, $s_e(k)$, and consequently the quantized difference signal, $d_q(k)$, is the same as in the decoder, then the compensation for quantization errors can be made with subsequent difference samples. Since the decoder is actually imbedded in the encoder, each of the common blocks for transmitting and receiving is discussed in the following paragraphs.

Figures 7 and 8 show block diagrams of an ADPCM transmitter and receiver as specified by CCITT.

Encoder

The function of the encoder or transmitter, shown in Figure 7, is to receive a 64-kbit/s log-PCM signal and transcode it to a 32-kbit/s ADPCM signal. This is accomplished by converting the log-PCM signal, s(k), to a linear signal, $s_l(k)$, from which an estimate, $s_e(k)$, of the signal is subtracted to obtain a difference signal, d(k). The next step is to adaptively quantize this difference signal, d(k), by first taking the log (base 2), then normalizing by the quantization scale factor, y(k), and finally coding the result, I(k). A more uniform signal-to-noise ratio can be achieved by coding the log of the signal rather than the linear representation. The normalization provides the adaptation to the quantization and is based on past coded samples. Adaptation is controlled bimodally, being comprised of a fast adaptation factor for signals with large amplitude fluctuations (i.e., speech) and a slow adaptation factor for signals which vary more slowly (i.e., data). A speed-control factor, $a_l(k)$, weights the fast and the slow adaptation factors to form a single quantization scale factor, y(k).

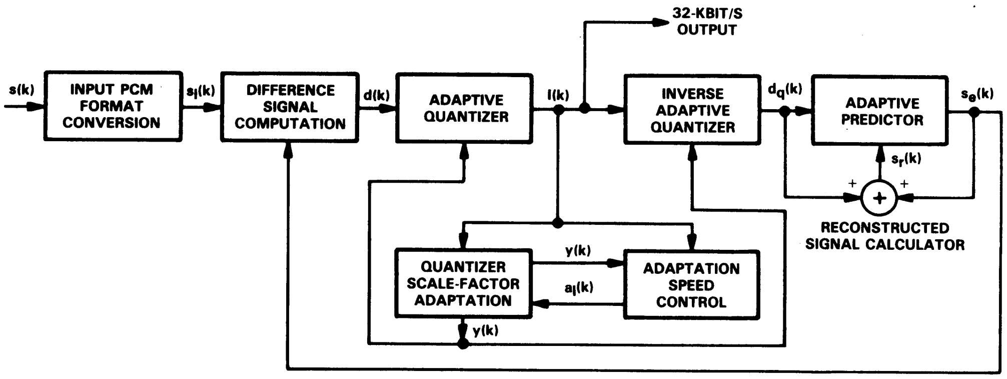
The inverse adaptive quantizer uses the same signal, I(k), that has been transmitted to reconstruct a quantized version of the difference, $d_q(k)$, and the same adaptive quantization characteristics as the adaptive quantizer section.

The quantized difference signal, $d_q(k)$, is input to an adaptive predictor which uses this input to compute a signal estimate, $s_e(k)$. The signal estimate, $s_e(k)$, is combined with the difference signal, $d_q(k)$, to determine a reconstructed signal, $s_r(k)$, which is the output in the decoder. This output is then subtracted from the next input sample to complete the feedback loop.

The adaptive predictor makes use of both an all-pole filter and an all-zero filter. The all-pole filter is a second-order filter with constrained adaptive coefficient values designed to match the slowly varying aspects of the speech signal. Since an all-pole predictor is particularly sensitive to errors, the predictor makes use of a sixth-order all-zero filter to offer signal stability even with transmission errors.

Decoder

The function of the decoder or receiver, shown in Figure 8, is to receive a 32-kbit/s ADPCM signal and transcode it to a 64-kbit/s log-PCM signal. To accomplish this, the decoder utilizes many of the elements used by the encoder. The received data, I(k), is processed by an inverse adaptive quantizer, identical to the one in the corresponding encoder, to determine a quantized difference signal, $d_q(k)$. By filtering the difference signal, $d_q(k)$, through the adaptive predictor together with the previously reconstructed signal, $s_r(k)$, a signal estimate, $s_e(k)$, is obtained. The signal estimate, $s_e(k)$, is added to the difference signal, $d_q(k)$, to compute the reconstructed signal, $s_r(k)$. The reconstructed signal, $s_r(k)$, is converted from a linear-PCM to a log-PCM signal, $s_p(k)$, which is then output following a synchronous



32-KBIT/S OUTPUT
s(k)
INPUT PCM FORMAT CONVERSION
sl(k)
DIFFERENCE SIGNAL COMPUTATION
d(k)
ADAPTIVE QUANTIZER
I(k)
INVERSE ADAPTIVE QUANTIZER
dq(k)
ADAPTIVE PREDICTOR
se(k)
sr(k)
+
+
RECONSTRUCTED SIGNAL CALCULATOR
y(k)
al(k)
QUANTIZER SCALE-FACTOR ADAPTATION
ADAPTATION SPEED CONTROL
y(k)

Figure 7. ADPCM Encoder Block Diagram (Diagram taken from CCITT Recommendation G.721)

coding adjustment. The coding adjustment limits the errors in tandem codings of a signal.

Note that the algorithm design achieves a convergence of the states of the encoder and decoder in spite of transmission errors. This convergence is a part of each of the adaptation computations and is demonstrated equationally in the following sections. The convergence is brought about by the inclusion of ($1\text{-}2^{-N}$) terms which provide a finiteness to the memory of the adaptation parameters.

Adaptive Quantization

Adaptive quantization, a multistage process, is used to determine the quantization scale factor and the speed control that controls the rate at which the scale factor is adapted. Quantization is actually a four-bit quantization (a sign bit plus three-bit magnitude), since a four-bit signal is the transmitted output of the ADPCM transcoder. The adaptive quantizer block can be noted in Figure 7.

The difference signal, d(k), an input to the quantization process, is calculated by subtracting the signal estimate, $s_e(k)$, from the linear-PCM signal, $s_l(k)$.

$$d(k) = s_l(k) - s_e(k) \tag{1}$$

This difference signal is normalized by taking the log (base 2) and subtracting from it the quantizer scale factor, y(k).

$$|I(k)| \leftarrow \log_2 |d(k)| - y(k) \tag{2}$$

Table 1 is used to provide the magnitude of the quantization result, |I(k)|, from this normalized input. The sign bit of the ADPCM output value, I(k), is the sign of the difference signal, d(k).

The quantizer scale factor, y(k), is comprised of two parts, and therefore bimodal in nature. The two parts, $y_l(k)$ and $y_u(k)$, are weighted by the speed-control factor, $a_l(k)$. For speech signals, $a_l(k)$ will tend toward a value of one; for voiceband data, $a_l(k)$ will tend toward zero. Refer to both Figures 7 and 8 for the inclusion of the quantizer scale factor and speed-control factor adaptation blocks.

$$y(k) = a_l(k)y_u(k-1) + [1 - a_l(k)]\ y_l(k-1) \tag{3}$$

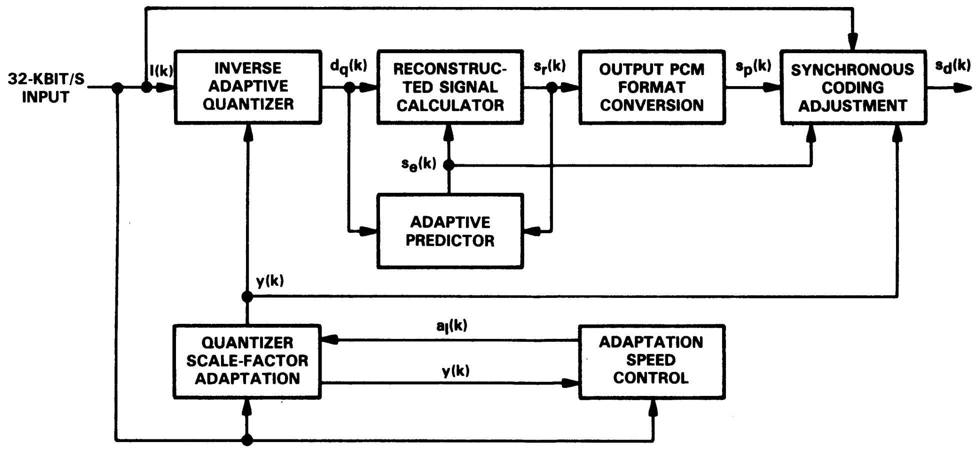
where $0 \leq a_l(k) \leq 1$

One of the factors, $y_u(k)$, is considered to be unlocked, since it can adapt quickly to rapidly changing signals (e.g., speech) and has a relatively short-term memory. This factor, $y_u(k)$, is recursively determined from the quantizer factor, y(k), and the discrete function, W(I).

$$y_u(k) = [1 - 2^{-5}]\ y(k) + 2^{-5}W[I(k)] \tag{4}$$

where $1.06 \leq y_u(k) \leq 10.00$

The factor, W(I), found in Table 2, is a function of I which causes $y_u(k)$ to adapt by larger steps for larger values of I. This gives $y_u(k)$ the freedom to track a signal almost instantaneously. Since y(k) is in the logarithmic domain, W(I) is effectively a multiplier of the scale factor.



32-KBIT/S INPUT
I(k)
INVERSE ADAPTIVE QUANTIZER
dq(k)
RECONSTRUCTED SIGNAL CALCULATOR
sr(k)
OUTPUT PCM FORMAT CONVERSION
sp(k)
SYNCHRONOUS CODING ADJUSTMENT
sd(k)
se(k)
ADAPTIVE PREDICTOR
y(k)
al(k)
QUANTIZER SCALE-FACTOR ADAPTATION
y(k)
ADAPTATION SPEED CONTROL

Figure 8. ADPCM Decoder Block Diagram
(Diagram taken from CCITT Recommendation G.721)

Table 1. I/O Characteristics of the Normalized Quantizer

Normalized Quantizer Input Range $\log_2\|d(k)\| - y(k)$	$\|I(k)\|$	Normalized Quantizer Output $\log_2\|d_q(k)\| - y(k)$
[3.16, $+\infty$)	7	3.34
[2.78, 3.16)	6	2.95
[2.42, 2.78)	5	2.59
[2.04, 2.42)	4	2.23
[1.58, 2.04)	3	1.81
[0.96, 1.58)	2	1.29
[-0.05, 0.96)	1	0.53
($-\infty$, -0.05)	0	-1.05

The other factor, $y_l(k)$, adapts more slowly and tracks signals which change slowly (e.g., voiceband data). This factor includes a lowpass filtering of the unlocked factor, $y_u(k)$. By including $y_u(k)$ in the manner shown, $y_l(k)$ is implicitly limited to the same range of values as the explicit limit placed on $y_u(k)$. Furthermore, the unity limit of $a_l(k)$ provides the same limit implicitly for $y(k)$ as for $y_l(k)$ and $y_u(k)$.

$$y_l(k) = [1 - 2^{-6}]\, y_l(k-1) + 2^{-6} y_u(k) \qquad (5)$$

A speed-control factor, $a_l(k)$, adjusts the relative weighting of these two scale factors by making use of the short- and long-term averages, $d_{ms}(k)$ and $d_{ml}(k)$, respectively, of the coded output to determine how rapidly the signal is changing. The combined scale factor, $y(k)$, cannot be larger than either the unlocked, $y_u(k)$, or locked $y_l(k)$, terms. Therefore, $a_l(k)$ is limited to one even if the predicted speed control, $a_p(k)$, is larger than one.

$$a_l(k) = \begin{cases} 1 & ,\text{if } a_p(k-1) > 1 \\ a_p(k-1) & ,\text{if } a_p(k-1) \le 1 \end{cases} \qquad (6)$$

Note that $a_p(k)$ is implicitly limited to a maximum value of 2, while the speed-control factor used to mix the two scale factors is capped at a value of 1. In determining $a_p(k)$, an additional term of 1/8 is added each time if the difference in the short- and long-term averages becomes too large (i.e., $| d_{ms}(k) - d_{ml}(k) | \ge 2^{-3} d_{ml}(k)$) or if there is an idle channel (i.e., $y(k) < 3$). Where neither of these conditions exist, a uniform, slowly varying signal can be assumed, such as occurs in data transmission.

Table 2. Scale-Factor Multipliers

$\|I\|$	7	6	5	4	3	2	1	0
W(I)	69.25	21.25	11.50	6.12	3.12	1.69	0.25	-0.75

$$a_p(k) = \begin{cases} [1 - 2^{-4}]\, a_p(k-1) + 2^{-3}, \text{ if } \\ \qquad |d_{ms}(k) - d_{ml}(k)| \geq 2^{-3} d_{ml}(k) \\ [1 - 2^{-4}]\, a_p(k-1) + 2^{-3}, \text{ if } y(k) < 3 \\ [1 - 2^{-4}]\, a_p(k-1), \text{ otherwise} \end{cases} \quad (7)$$

The short-, $d_{ms}(k)$, and long-term, $d_{ml}(k)$, averages of the transmitted ADPCM signal, I(k), are actually determined by averaging a weighted function, F(I), of the transmitted I, shown in Table 3.

$$d_{ms}(k) = [1 - 2^{-5}]\, d_{ms}(k-1) + 2^{-5} F\,[I(k)] \quad (8)$$

$$d_{ml}(k) = [1 - 2^{-7}]\, d_{ml}(k-1) + 2^{-7} F\,[I(k)] \quad (9)$$

The scale-factor and speed-control adaptations are a part of both the encoder and decoder logic. The adaptive quantization block has been specifically included in Figure 7, showing the encoder. For the decoder, the adaptive quantizer is included as part of the synchronization block to aid in the reduction of errors in tandem codings.

Table 3. Rate-of-Change Weighting Function

|I|	7	6	5	4	3	2	1	0
F(I)	7	3	1	1	1	0	0	0

Inverse Adaptive Quantization

Inverse adaptive quantization is a process in which the four-bit ADPCM signal, I(k), is used to determine the normalized log of the difference signal from Table 1. The result is actually a quantized version of the difference signal, $d_q(k)$, determined by adding the scale factor, y(k), to the value specified by Table 1 and calculating, the inverse log (base 2) of this sum.

$$d_q(k) = \log_2^{-1} [\{\log_2 |d_q(k)| - y(k)\} + y(k)] \quad (10)$$

For both the encoder and decoder, this quantized difference signal is the input to the reconstruction signal calculator and the adaptive predictor, as shown in Figures 7 and 8.

Adaptive Prediction

The adaptive predictive filter is a two-pole, six-zero filter used to determine the signal estimate. The combination of both poles and zeroes allows the filter to model more effectively any general input signal. The sixth-order all-zero section helps to stabilize the filter and prevent it from drifting into oscillation. For both the poles and the zeroes, the coefficients, $a_i(k)$ and $b_i(k)$, respectively, are adapted. This adaptation is based upon a gradient algorithm to further adjust the filter model to the input signal. Figures 9 and 10 show the sixth-order and second-order filters, respectively.

The signal estimate, $s_e(k)$, represents the sum of the all-pole filter and the all-zero filter. Since the sum of the all-zero filter is used to aid the determination of the pole coefficients, it is also extracted as a separate sum, $s_{ez}(k)$. The reconstructed signal, the output in the receiver, is the sum determined by the quantized difference signal $d_q(k)$, and the signal estimate, $s_e(k)$.

$$s_e(k) = \sum_{i=1}^{2} a_i(k-1) s_r(k-i) + s_{ez}(k) \quad (11)$$

$$s_{ez}(k) = \sum_{i=1}^{6} b_i(k-1) d_q(k-i) \quad (12)$$

$$s_r(k-i) = s_e(k-i) + d_q(k-i) \quad (13)$$

The adaptation of the pole coefficients, $a_i(k)$, is shown in the equations below. The gradient function is determined from a signal, p(k), that is equivalent to the reconstructed signal minus the contribution of the pole filter output. Stability of the filter is further provided by explicitly limiting the coefficients.

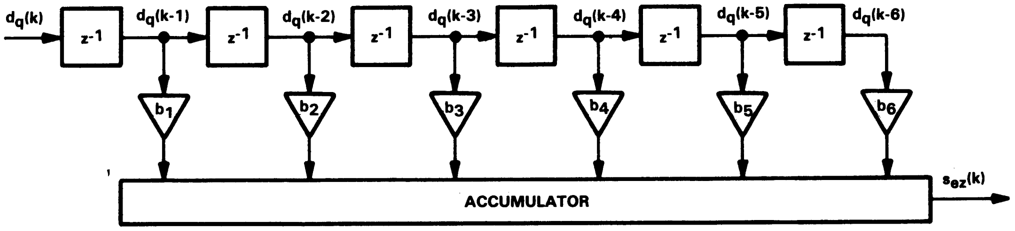


Figure 9. Sixth-Order All-Zero (FIR) Filter

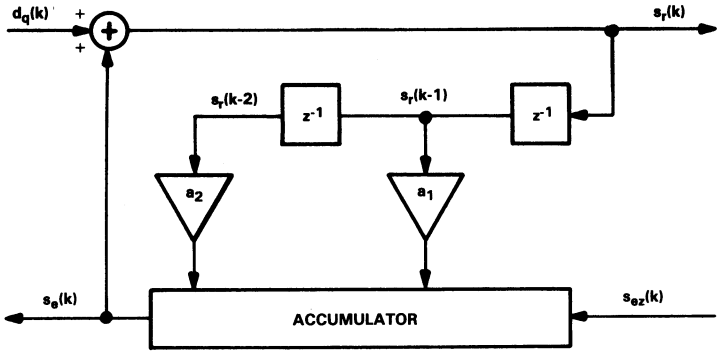


Figure 10. Second-Order IIR Filter

$$a_1(k) = [1 - 2^{-8}]\, a_1(k-1) + 3 \cdot 2^{-8} \text{sgn}\, [p(k)]\, \text{sgn}[p(k-1)] \qquad (14)$$

where $|a_1(k)| \leq 1 - 2^{-4} - a_2(k)$

$$a_2(k) = [1 - 2^{-7}]\, a_2(k-1) + 2^{-7}\{\text{sgn}\, [p(k)]\, \text{sgn}\, [p(k-2)] - f[a_1(k-1)]\, \text{sgn}\, [p(k)]\, \text{sgn}\, [p(k-1)]\} \qquad (15)$$

where $|a_2(k)| \leq 0.75$

$$p(k) = d_q(k) + s_{ez}(k) \qquad (16)$$

$$f(a_1) = \begin{cases} 4a_1 & , \text{if}\, |a_1| \leq 1/2 \\ 2\text{sgn}(a_1), & \text{if}\, |a_1| > 1/2 \end{cases} \qquad (17)$$

where $\text{sgn}(0) = +1$

For the coefficients, $b_i(k)$, of the sixth-order all-zero filter, the adaptation procedure is similar, but the limit is implicit in the equations to a maximum of ± 2. The gradient function, in this case, is determined by the current difference signal, $d_q(k)$, and corresponding difference signal, $d_q(k-i)$, at the specific filter tap.

$$b_i(k) = [1 - 2^{-8}]\, b_i(k-1) + 2^{-7}\text{sgn}\, [d_q(k)]\, \text{sgn}\, [d_q(k-i)] \qquad (18)$$

where $i = 1,2, \ldots 6$ and $-2 \leq b_i(k) \leq +2$

Signal Conversion

Signal conversion consists of the conversion from an 8-bit log-PCM representation of a signal to a 13-bit linear PCM representation (note Figure 7), or the reverse (note Figure 8). Signal conversions of this type are described in the application report on COMPANDING ROUTINES FOR THE TMS32010. In the encoder, the log-PCM signal, s(k), is expanded to create the linear-PCM value, $s_l(k)$. The decoder, on the other hand, compresses the reconstructed signal, $s_r(k)$, to create the log-PCM signal, $s_p(k)$.

Reconstructed Signal Synchronization

To avoid a cumulative distortion in synchronous tandem codings, an adjustment to the reconstructed signal is specified. The adjustment block, shown in Figure 8, estimates the quantization of the encoder by determining a difference signal and executing the adaptive quantization logic. The quantization result is an estimate of the received value of I(k).

The difference signal, $d_x(k)$, is determined by subtracting the signal estimate, $s_e(k)$, from the linear-PCM signal, $s_{lx}(k)$, which is itself determined by expanding the log-PCM signal, $s_p(k)$.

$$d_x(k) = s_{lx}(k) - s_e(k) \qquad (19)$$

The adaptive quantization process produces the estimate of the ADPCM code value, $I_d(k)$. If the estimate implies a difference signal that is lower than the received interval boundary, the log-PCM code is changed to the next most positive value. An estimate implying a difference signal larger than the received interval boundary requires the log-PCM code to be changed to the next most negative value; otherwise, the log-PCM value is left unchanged. The adjusted log-PCM value is denoted as $s_d(k)$ in the following equation to differentiate it from the input value, $s_p(k)$.

$$s_d(k) = \begin{cases} s_p^+(k), & d_x(k) < \text{lower interval boundary} \\ s_p^-(k), & d_x(k) \geq \text{upper interval boundary} \\ s_p(k), & \text{otherwise} \end{cases} \qquad (20)$$

where

$s_d(k)$ = output PCM of the decoder

$s_p^+(k)$ = next more positive PCM level (if $s_p(k)$ is the most positive level, then $s_p^+(k) = s_p(k)$)

$s_p^-(k)$ = next more negative PCM level (if $s_p(k)$ is the most negative level, then $s_p^-(k) = s_p(k)$)

FULL-DUPLEX IMPLEMENTATION OF ADPCM ON A TMS32010

The specific implementation of ADPCM presented here involves the use of a single TMS320M10 to accomplish a full-duplex transcoder. The TMS320M10 is a masked ROM, microcomputer version of the TMS32010, which requires no external program memories. A full-duplex transcoder provides transmission in both directions simultaneously. Such a transcoder is depicted in Figure 11. A complete system diagram of a full-duplex communications channel is shown in Figure 12. In comparison to current systems that modulate a 64-kbit/s A-law or μ-law PCM signal on a carrier for transmission, the described system transcodes the 64-kbit/s code to a 32-kbit/s code. This 32-kbit/s code, which requires correspondingly less bandwidth, is modulated on the carrier for transmission.

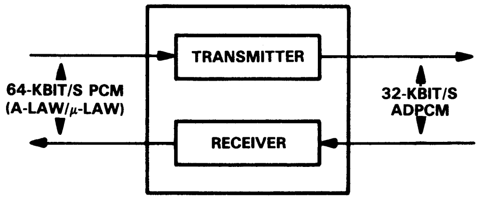


Figure 11. Full-Duplex ADPCM Transcoder

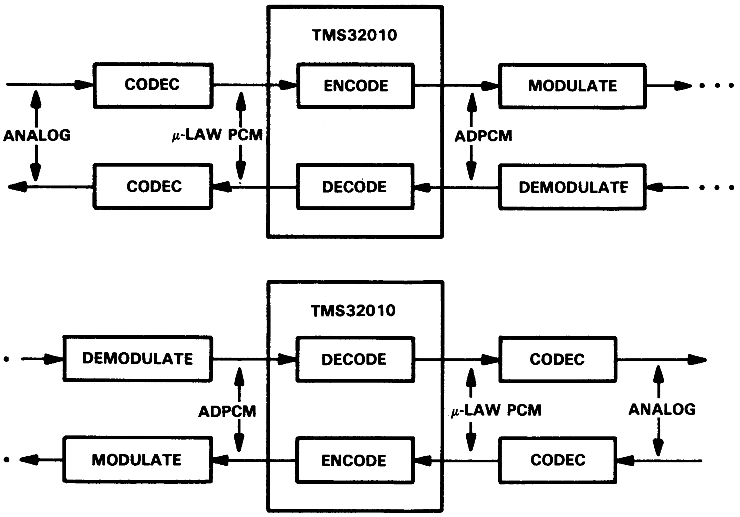


Figure 12. Full-Duplex Telecommunications Channel

Hardware Logic and I/O

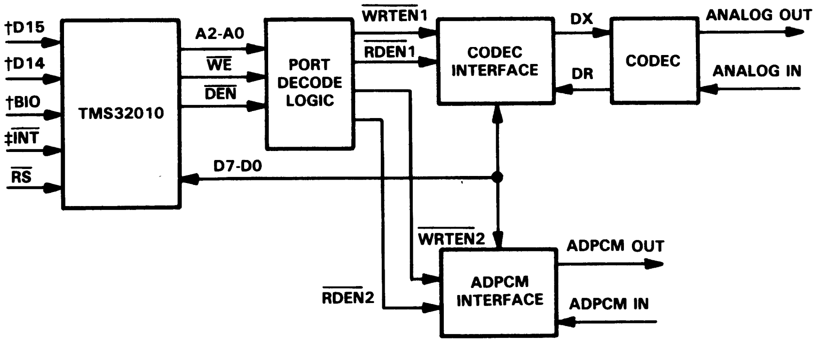
The hardware required to implement the ADPCM system consists of an addition to an existing circuit. As shown in Figure 13, the TMS32010 addresses the external I/O blocks through its port addressing structure. The lower three address lines, A2-A0, form a port address that can be decoded by port decode logic to provide specific enable lines (e.g., $\overline{\text{WRTEN1}}$ and $\overline{\text{RDEN1}}$) to the various peripheral blocks. The TMS32010 reads and writes the 64-kbit/s data through the codec interface eight bits at a time. The sampling frequency is 8 kHz. For this full-duplex implementation, one sample is written and one sample is read every 125 μs.

Figure 13 also shows the serial interface to the codec that provides the μ-law companded PCM data, although this is not part of the transcoding system itself. The log-PCM signal may already be available (e.g., in existing digital telecom networks) and, as such, may be interfaced to the TMS32010 either directly as parallel data or serially through conversion logic. Parallel codecs are also becoming available to reduce the hardware logic and interface required for those systems which do not already include a codec. The TMS32010 is available at crystal and clock input rates of 20.5 MHz which may be divided down to provide the codec timing and further reduce the logic requirement.

At the other end of the transcoder function, the TMS32010 reads and writes the 32-kbit/s ADPCM data through the ADPCM interface four bits at a time for each 125-μs period. This interface provides four-bit parallel data which may be serialized, if required, for transmission or storage.

Software Logic and Flow

Tables 4 and 5 list the various blocks in the algorithm, directly relating them to Figures 7 and 8 by the signal names given in the description and function. The blocks are listed in the order in which they are executed. Also listed is processor demand or loading which consists of the amount of program memory used to implement the given function and the number of instruction cycles executed in worst case. There are more blocks in the table than are shown in the figures (e.g., the algorithm uses the adaptive predictor at one point to produce the signal estimate, and later returns to update or adapt the predictor coefficients). Each block has been implemented using the equations given in previous sections concerning the ADPCM algorithm. For convenience, the equations implemented in each block are listed in the description section for the block. A more detailed description of the TMS32010 implementation is given in the next section.



† Half-duplex, CCITT bit-compatible, version only
‡ Full-duplex version only

Figure 13. System Interface of a TMS32010 ADPCM Transcoder

Table 4. Full-Duplex Transmitter

Order	Function	Description	CPU Clocks	Program Memory (Words)
1.	INPUT PCM	Read an 8 bit μ-law PCM sample [s(k)] and linearize it to a 12-bit sample [s_l(k)].	7	0004
2.	COMPUTE SIGNAL ESTIMATE	Calculate the signal estimate [s_e(k)] from the previous data samples [d_q(k)] and reconstructed samples [s_r(k)] through the predictor filter. (12),(11)	30	001E
3.	COMPUTE ADAPTIVE QUANTIZER	Calculate speed control [a_l(k)] and quantizer scale factor [y(k)] from past quantizer output [I(k)]. (6),(3)	33	0021
4.	COMPUTE DIFFERENCE SIGNAL	Calculate the difference signal [d(k)] from the current sample [s_l(k)] and signal estimate [s_e(k)]. (1)	3	0003
5.	COMPUTE QUANTIZED OUTPUT	Calculate the log of the difference signal [d(k)] and adaptively quantize the result to yield the ADPCM output [I(K)]. (2)	46	00AD
6.	OUTPUT ADPCM	Write the ADPCM output [I(k)].	2	0001
7.	COMPUTE RECON-STRUCTED SIGNAL	Calculate the inverse of the adaptively quantized signal [d_q(k)] and the reconstructed signal difference [s_r(k)]. (10),(13)	43	0027
8.	COMPUTE SCALE FACTOR	Calculate the updates for the scale-factor adaptation. (4),(5)	46	002F
9.	COMPUTE SPEED CONTROL	Calculate the update for the speed-control adaptation. (8),(9),(7)	30	001B
10.	COMPUTE PREDICTOR ADAPTATION	Calculate the updates for the adaptive predictor filter coefficients. (18),(16),(17),(14),(15)	102	006B

Table 5. Full-Duplex Receiver

Order	Function	Description	CPU Clocks	Program Memory (Words)
1.	INPUT ADPCM	Read the ADPCM input [I(k)].	2	0001
2.	COMPUTE SIGNAL ESTIMATE	Calculate the signal estimate [s_e(k)] from the previous data samples [d_q(k)] and reconstructed samples [s_r(k)] through the predictor filter. (12),(11)	30	001E
3.	COMPUTE ADAPTIVE QUANTIZER	Calculate speed control [a_l(k)] and quantizer scale factor [y(k)] from the past quantizer output [I(k)]. (6),(3)	33	0021
4.	COMPUTE QUANTIZED DIFFERENCE	Calculate the inverse of the adaptively quantized signal [d_q(k)]. (10)	47	002F
5.	COMPUTE SCALE FACTOR	Calculate the updates for the scale-factor adaptation. (4),(5)	48	002F
6.	COMPUTE SPEED CONTROL	Calculate the update for the speed-control adaptation. (8),(9),(7)	29	001B
7.	COMPUTE RECON-STRUCTED SIGNAL	Calculate the reconstructed signal [s_r(k)]. (13)	3	0003
8.	COMPUTE PREDICTOR ADAPTATION	Calculate the updates for the adaptive predictor filter coefficients. (18),(16),(17),(14),(15)	90	006B
9.	COMPUTE LOG-PCM	Convert the reconstructed linear-PCM signal [s_r(k)] to a μ-law PCM signal [s_p(k)].	39	0074
10.	OUTPUT PCM	Write the μ-law output [s_p)k)].	2	0001
11.	WAIT	Spin until the next interrupt.	—	0006

Implementation and Advantages of TMS32010 Architecture

This implementation is only concerned with μ-law PCM, although A-law PCM may also be used. Additional information on log-PCM companding is found in an application report, COMPANDING ROUTINES FOR THE TMS32010. The implementation is simplified here so that the expansion is a simple table lookup which saves 21 instruction cycles over the algorithmic approach.

The processing of the signal through the predictor filter is similar to the processing discussed in the application report, IMPLEMENTATION OF FIR/IIR FILTERS WITH THE TMS32010. The filter used in this ADPCM algorithm is a combination of a second-order IIR filter and a sixth-order all-zero or FIR filter. The filters are shown in Figures 9 and 10, respectively, with the system interaction shown in Figure 14.

Several manipulations of data format occur in adapting the predictor coefficients. In updating the coefficients of the all-zero filter (the B_i's), the coefficients that are normally Q14 numbers are loaded with a shift allowing the calculations to be done in a Q29 representation. This greatly simplifies the subtraction of the leakage term and the prediction gain. The leakage term, which occurs here in the predictor coefficient adaptation and also in the speed-control and scale-factor adaptation, controls the rate of change of the parameter away from zero and towards the absolute maximum limits of the particular parameter. The prediction gain also uses

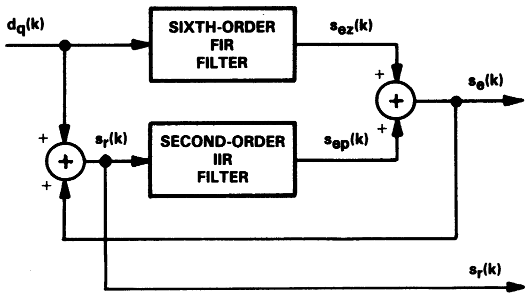


Figure 14. Predictor Filter

an approach whereby the signs are actually stored as a signed Q11 value. In this way, the product is a Q22 value of the correct sign and can be added to the B value, equivalent to a Q29 value times 2^{-7}. As with the filter process itself, the signs of the Dq values are propagated through each filter tap delay with the LTD instruction. An example for one of the B_i values is shown in Figure 15.

A similar process takes place in adapting the prediction coefficients (A_i's) in the second-order filter, although the fixed-point representation of the coefficients is Q26. The remaining requirement is to limit-check the A_i values.

The adaptive quantization section requires that the log (base 2) of the difference signal be taken, the result normalized, and the normalized value quantized. Taking the log (base 2) of a number is accomplished by using the approximation

$$\log_2(1 + x) = x \tag{21}$$

```
*;*************************************************************
*;
*;  COMPUTE COEFFICIENTS OF THE 6TH-ORDER PREDICTOR
*;
*;  Bi(k) = [1 - 2**-8] * Bi(k-1)
*;          + 2**-7 * SGN[DQ(k)] * SGN[DQ(k-i)]
*;
*;        FOR  i = 1 ... 6
*;        AND Bi IS IMPLICITLY LIMITED TO +/- 2
*;
*;        NOTATION:   Bn   -- 16b TC (Q14)
*;                    SDQn -- +2048 IF DQn POSITIVE (Q11)
*;                            -2048 IF DQn NEGATIVE (Q11)
*;
*;*************************************************************
*;
GETB6  LT   SDQ6       * (Q11)
       LAC  B6,15      * (Q29)
       SUB  B6,7       * B6 * 2**-8  (Q29)
       MPY  SDQ        * SGN(SDQ)*SGN(SDQ6)*2**-7 (Q29)
       LTD  SDQ5       * (Q11)
       SACH B6,1       * (Q14)
       .
       .
       .
```

Figure 15. Predictor Coefficient Adaptation Code

The characteristic of the result is the bit position of the most significant one digit in the absolute value. The result can be represented as a Q7 value. Finding the most significant digit is most efficiently done by a binary search technique. This technique is discussed in the application report, FLOATING-POINT ARITHMETIC WITH THE TMS32010. Since the exponent is part of the number instead of being stored in a separate register, one of the auxiliary registers is loaded with the exponent value. The auxiliary register stores it in memory and adds it to the mantissa in the accumulator. A short example of this is shown in the excerpted code in Figure 16 where the signal has an assumed exponent value of 9.

Normalization of this log value is simply a subtraction of a scale factor which may be as large in fixed magnitude as the largest logarithmic value represented in Q7 notation. The result of the subtraction may be a negative value. Since the normalized result is to be quantized in a nonuniform manner and one of the quantization levels could contain both positive and negative values, the normalized result is scaled by adding a fixed value of 2048. Nonuniform quantization can be performed by a binary-type search technique. The normalization and quantization are included in the program shown in Figure 16.

```
       SACL D          * STORE THE DIFFERENCE SIGNAL
       SACH DS         * SAVE THE SIGN OF D
*;
*;*****************************************************************
*;  ADAPTIVE QUANTIZER
*;
*;     INPUTS:  DIFFERENCED PCM SAMPLE  -- D
*;              QUANTIZER SCALE FACTOR  -- Y
*;
*;     OUTPUT:  4-BIT QUANTIZED, NORMALIZED DIFFERENCE -- I
*;
*;     NOTATION:  D    -- 16b TC (Q0)
*;                Y    -- 13b SM (Q9) POSITIVE VALUE ONLY
*;
*;*****************************************************************
*;
*;                             DS
*;           +-------------------------------+
*;           |                               |
*;           |                               V
*;   D    +--+----+  DL   +-------+  DLN  +--------+   I
*; ------>|  LOG  |------>|  SUB  |------>|  QUAN  |------>
*;        +-------+       +-------+       +--------+
*;                            ^
*;                            | Y
*;
*;*****************************************************************
*;
*;  COMPUTE THE LOG OF THE DIFFERENCE SIGNAL
*;
AQUAN  ABS             * LOGS OF NEGATIVES ARE UNREAL.
       SACL TEMP1      * SAVE THE ABSOLUTE DIFFERENCE.
GETEXP SUB  ONE,8      * BEGIN BINARY SEARCH.
       .
       .
       .
EXP9   LARK 0,9        * EXP = 9; STORE IN AR0.
       LAC  TEMP1,14   * LOAD TO ISOLATE 8 MSB'S.
```

Figure 16. Adaptive Quantization Code

```
        SACH TEMP1        * SAVE MANTISSA.
        LAC  TEMP1        * RELOAD FOR MANTISSA RECOMBINATION.
        B    GETMAN
        .
        .
        .
GETMAN  AND  M127         * MASK TO RETAIN ONLY SEVEN BITS.
        SAR  0,TEMP1      * MOVE EXPONENT TO MEMORY FROM AR0.
        ADD  TEMP1,7      * ADD EXPONENT TO MANTISSA FOR LOG VALUE.
*;
*;   SCALE BY SUBTRACTION
*;
SUBTB   ADD  ONE,11       * ADD AN OFFSET OF 2048.
        SUB  TEMP3        * TEMP3 = Y(K) >> 2

*;
*;   4-BIT QUANTIZER
*;
*; QUANTIZATION TABLE FOR 32KB OUTPUT (OFFSET: 2048)
*;
ITAB1   EQU  2041
ITAB2   EQU  2171
ITAB3   EQU  2250
ITAB4   EQU  2309
ITAB5   EQU  2358
ITAB6   EQU  2404
ITAB7   EQU  2453
*;
QUAN    SUB  K2309        * ITAB4
        BGEZ CI4TO7
CI0TO3  ADD  K138         * ITAB2          I  = 0-3
        BGEZ CI2TO3
CI0TO1  ADD  K130         * ITAB1          I  = 0-1
        BGEZ IEQ1
IEQ0    LACK 0
        B    GETIM
IEQ1    LACK 1
        B    GETIM
CI2TO3  SUB  K79          * ITAB3          I  = 2-3
        BGEZ IEQ3
IEQ2    LACK 2
        B    GETIM
IEQ3    LACK 3
        B    GETIM
CI4TO7  SUB  K95          * ITAB6          I  = 4-7
        BGEZ CI6TO7
CI5TO6  ADD  K46          * ITAB5          I  = 5-6
        BGEZ IEQ5
IEQ4    LACK 4
        B    GETIM
IEQ5    LACK 5
        B    GETIM
CI6TO7  SUB  K49          * ITAB6          I  = 6-7
        BGEZ IEQ7
```

Figure 16. Adaptive Quantization Code (Continued)

```
IEQ6    LACK 6
        B    GETIM
IEQ7    LACK 7
GETIM   SACL IM           * ACCUM = |I|
        XOR  DS           * ADD SIGN BIT AND FLIP IF NECESSARY.
        AND  M15          * MASK FINAL FOUR-BIT VALUE.
        SACL I            * SAVE ADPCM OUTPUT VALUE.
        .
        .
```

Figure 16. Adaptive Quantization Code (Concluded)

Determining the inverse of the 4-bit quantized ADPCM value involves another technique. The code to complete this task is much shorter in program memory requirement and somewhat faster in execution time. The same type of approximation is involved in determining the antilog as used in taking the log,

$$\log_2^{-1}(x) = 1 + x \qquad (22)$$

After separating the exponent from the mantissa in the log representation, the quantized difference signal may be recovered by using the exponent to select a scaling factor. The scaling factor or multiplier is used to shift the mantissa to the proper representation, either right or left. Some of the multipliers may be stored as negative values rather than positive values, using the sign of the result to determine whether the answer is obtained from the high half of the accumulator (effectively a right shift) or from the low half of the accumulator (a left shift). The program for this process is shown in Figure 17.

```
*;
*;*****************************************************
*;  INVERSE ADAPTIVE QUANTIZER
*;
*;     INPUTS:  4-BIT QUANTIZED SAMPLES -- IM
*;              QUANTIZER SCALE FACTOR  -- Y
*;
*;     OUTPUT:  RECONSTRUCTED DIFFERENCE SIGNAL  -- DQ
*;
*;     NOTATION:  Y    -- 13b SM (Q9) POSITIVE VALUE ONLY
*;                DQ   -- 16b TC (Q0)
*;                SDQ  -- +2048 IF DQ POSITIVE (Q11)
*;                        -2048 IF DQ NEGATIVE (Q11)
*;
*;*****************************************************
*;
*;                          DQS
*;            +-----------------------------+
*;            |                             |
*;            |                             V
*;    I    +---+---+ DQLN  +-------+  DQL  +--------+  DQ
*;  ------>|RECONST|------>|  ADD  |------>|ANTILOG |------>
*;         +-------+       +-------+       +--------+
*;                             ^
*;                             | Y
*;
*;*****************************************************
*;
*;  CONVERT QUANTIZED DIFFERENCE BACK TO LOG DOMAIN
*;
```

Figure 17. Inverse Adaptive Quantization Code

```
IAQUAN LAC   IM
       ADD   INQTAB       * RECONSTRUCTION TABLE
       TBLR TEMP1         * READ NORMALIZED VALUE.
*;
*;  ADD NORMALIZING SCALE FACTOR BACK IN
*;
ADDA   LAC   TEMP1
       ADD   TEMP3        * Y >> 2
       AND   M2047
       SACL TEMP2
*;
*;  CONVERT THE LOG VALUE TO THE LINEAR DOMAIN
*;
*;
ALOG   LAC   TEMP2,9      * EXTRACT EXPONENT.
       SACH TEMP1         * SAVE EXPONENT VALUE.
       LACK 127
       AND   TEMP2        * MASK FOR LOG MANTISSA ONLY.
       ADD   ONE,7        * 1+x
       SACL TEMP2         * EXTRACT MANTISSA.
       LT    TEMP2        * PREPARE TO SHIFT.
       LAC   TEMP1
       ADD   SHIFT        * LOOK UP MULTIPLIER.
       TBLR TEMP3
       MPY   TEMP3        * MULTIPLY MANTISSA BY SHIFT FACTOR.
       PAC
       BLZ   LEFTSF       * NEGATIVE VALUES CORRESPOND TO LEFT SHIFT.
       SACH DQ,1          * RIGHT SHIFT; SAVE MAGNITUDE OF DQ.
       B     ADDSGN
LEFTSF ABS                * LEFT SHIFT; RESTORE MAGNITUDE.
       SACL DQ            * SAVE MAGNITUDE OF DQ.
ADDSGN LAC   ONE,11       * ASSUME POSITIVE AND SAVE THE SIGN.
       SACL SDQ           *   (SIGN IS Q11; REMEMBER FILTER.)
       LAC   I            * CHECK SIGN OF SAMPLE.
       SUB   ONE,3
       BLZ   QSFA         * FINISHED FOR POSITIVE VALUES (I<8).
       ZAC
       SUB   DQ           * COMPUTE TWO'S COMPLEMENT OF THE MAGNITUDE.
       SACL DQ            * SAVE NEGATIVE DQ VALUE.
       LAC   MINUS,11     * SIGN IS Q11; REMEMBER FILTER.
       SACL SDQ           * SAVE SIGN.
       .
       .
       .
*;
*; INVERSE QUANTIZING TABLE
*;
IQTAB  BSS   0
       DATA 65401
       DATA 68
       DATA 165
       DATA 232
       DATA 285
       DATA 332
       DATA 377
       DATA 428
```

Figure 17. Inverse Adaptive Quantization Code (Continued)

```
*;
*; SHIFT MULTIPLIER TABLE
*;
SHFT    BSS  0
        DATA 256
        DATA 512
        DATA 1024
        DATA 2048
        DATA 4096
        DATA 8192
        DATA 16384
        DATA -1
        DATA -2
        DATA -4
        DATA -8
        DATA -16
        DATA -32
        DATA -64
        DATA -128
```

Figure 17. Inverse Adaptive Quantization Code (Concluded)

The adaptation of the speed-control and the scale-factor parameters, used to adapt the stepsize in the adaptive quantizer and inverse adaptive quantizer, requires multiple uses of the technique of adjusting the fixed-point representation. The Q point is adjusted for convenience of the table constants which are part of the adaptation process and for saving the output value from the accumulator. Some limit-checking must also take place in calculating the unlocked-scale factor and the speed-control parameter.

In the calculation of the locked-scale factor and its inclusion in the mixing process for determining the overall scale factor used for stepsize quantization, the parameter is maintained with a greater resolution (19 bits of value plus its sign) than can be stored in a single memory. Calculations involving this parameter must then become two stage, both in terms of accumulations and in determining products. The code involving this parameter is listed in Figures 18 and 19.

```
*;
*;*************************************************************
*;
*;   QUANTIZER SCALE FACTOR ADAPTATION
*;
*;      INPUT:    I  : 32KB CODED SAMPLES
*;
*;      OUTPUT:  YU,YL  : NEXT SAMPLE SCALE FACTOR
*;
*;      NOTATION:  Y     -- 13b SM (Q9) POSITIVE VALUE ONLY
*;                 YU    -- 13b SM (Q9) POSITIVE VALUE ONLY
*;                 YL    -- 19b SM (Q15) POSITIVE VALUE ONLY
*;
*;*************************************************************
*;

       .
       .
       .

*;
*;   UPDATE SLOW ADAPTATION SCALE FACTOR -- CONSTANT = 1/64
*;
*;   YL(k) = (1-2**-6)*YL(k-1) + 2**-6 * YU(k)
*;
FILTE  LAC   YLH,6        * SHIFT YL LEFT BY 6.
       SACL  TEMP1        * TEMP1 = YLH * 2**6
       LAC   YLL,6
       SACL  TEMP2
       SACH  TEMP3        * TEMP3 | TEMP2 = YLL * 2**6
       LAC   TEMP3        * SUPPRESS SIGN EXTENSION.
       AND   M63
       SACL  TEMP3
       ZALH  TEMP1
       ADDH  TEMP3
       ADDS  TEMP2        * ACCUM = YL * 2**6
       SUBH  YLH
       SUBS  YLL          * ACCUM = YL * 2**6 - YL
       ADD   YU,6         * ACCUM = YL * 2**6 - YL + YU
       SACL  TEMP1
       SACH  TEMP2        * RESULT = YL (SHIFTED LEFT BY 6)
       LAC   TEMP1,10     * SHIFT RESULT RIGHT 6 --> q15
       SACH  TEMP1
       LAC   TEMP1
       AND   M1023        * MASK SIGN EXTENSION.
       ADD   TEMP2,10
       SACL  YLL          * SAVE YLL.
       SACH  YLH
       LACK  7            * MASK UPPER 13 BITS.
       AND   YLH
       SACL  YLH          * SAVE YLH.
       .
       .
       .
```

Figure 18. Quantizer Scale-Factor Adaptation: Locked-Factor Calculation

```
               .
               .
               .
*;
*;   FORM LINEAR COMBINATION OF FAST AND SLOW SCALE FACTORS
*;
*;   Y(k) = (1-AL(k))*YL(k-1) + AL(k)*YU(k-1)
*;
MIX     LAC  YLL,10       * SHIFT YL RIGHT BY 6.
        SACH TEMP3        * (IE SCALE YL TO MATCH YU SINCE YL
        LAC  TEMP3        *  CONTAINS 6 MORE LSB'S)
        AND  M1023
        ADD  YLH,10
        SACL TEMP3        * LOW HALF
        LAC  YU
        SUB  TEMP3        * YU-(YLL>>6)
        SACL TEMP3
        ZALH YLH
        ADDS YLL
        LT   AL           * AL IS IN 1.Q6
        MPY  TEMP3
        APAC              * YL + AL*(YU-(YLL>>6))
        SACL TEMP3
        SACH TEMP2        * TEMP2 | TEMP3 = Y * 2**6
        LAC  TEMP3,10     * SHIFT RIGHT BY 6.
        SACH TEMP3
        LAC  TEMP3
        AND  M1023        * MASK SIGN EXTENSION.
        ADD  TEMP2,10
        AND  M8191
        SACL Y            * SAVE Y.
        LAC  Y,14
        SACH TEMP3        * SAVE Y >> 2 .
        .
        .
        .
```

Figure 19. Quantizer Scale-Factor Adaptation: Mixing

CCITT IMPLEMENTATION OF ADPCM ON A TMS32010

The implementation of ADPCM that produces a bit-for-bit compatible solution with the CCITT test vectors uses a single TMS320M10 to accomplish a half-duplex transcoder. This solution can provide capability as either a transmitter or a receiver using either A-law or μ-law companding.

Hardware Logic and I/O

The hardware system for this transcoder implementation differs from Figure 13 in that data pins D15 and D14 are used to determine the mode of operation. Table 6 shows the operating mode for the various combined states of the data pin inputs.

Table 6. Operating Mode Selection

D15*	D14*	Operating Mode
L	L	μ-law transmitter
L	H	μ-law receiver
H	L	A-law transmitter
H	H	A-law receiver

*H = High logic level
L = Low logic level

Additionally, as has been noted in Figure 13, the interrupt or sample timing is an input to the $\overline{\text{INT}}$ pin in the full-duplex implementation; here it is an input to the $\overline{\text{BIO}}$ pin. Each 125-μs period, the TMS32010 reads a 64-kbit/s sample from the codec and writes a 32-kbit/s sample to the ADPCM interface, or it reads the 4-bit ADPCM sample and writes an 8-bit PCM sample to the codec.

For real-time execution, the TMS32010 requires the use of a 25-MHz clock input.

Software Logic and Flow

Tables 7 and 8 list the various blocks in the algorithm, directly relating them to Figures 7 and 8 by the signal names given in the description and function. No differentiation is made between the transmitter or receiver using A-law or μ-law. The blocks are listed in the order in which they are executed. Also listed is processor demand or loading which consists of the amount of program memory used to implement the given function and the number of instruction cycles executed in worst case. There are more blocks in the tables than are shown in the figures (e.g., the algorithm uses the adaptive predictor at one point to produce the signal estimate, and later returns to update or adapt the predictor coefficients). Each block has been implemented using the equations given in previous sections concerning the ADPCM algorithm. For convenience, the equations implemented in each block are listed in the description section for the block. Additional details of the TMS32010 implementation are given in the next section, especially as they differ from the full-duplex implementation. The appendix contains a complete listing of the code.

Table 7. CCITT Transmitter

Order	Function	Description	CPU Clocks	Program Memory (Words)
1.	INPUT PCM	Read an 8 bit log-PCM sample [s(k)] and linearize it to a 12-bit sample [s_l(k)].	25 μ-law 24 A-law	0024 μ-law 0031 A-law
2.	COMPUTE SIGNAL ESTIMATE	Calculate the signal estimate [s_e(k)] from the previous data samples [d_q(k)] and reconstructed samples [s_r(k)] through the predictor filter. (12), (11)	396	0167
3.	COMPUTE ADAPTIVE QUANTIZER	Calculate speed control [a_l(k)] and quantizer scale factor [y(k)] from past quantizer output [I(k)]. (6), (3)	30	001E
4.	COMPUTE DIFFERENCE SIGNAL	Calculate the difference signal [d(k)] from the current sample [s_l(k)] and signal estimate [s_e(k)]. (1)	3	0003
5.	COMPUTE QUANTIZED OUTPUT	Calculate the log of the difference signal [d(k)] and adaptively quantize the result to yield the ADPCM output [I(k)]. (2)	42	00AD
6.	OUTPUT ADPCM	Write the ADPCM output [I(k)].	6	0005
7.	COMPUTE RECON-STRUCTED SIGNAL	Calculate the inverse of the adaptively quantized signal [d_q(k)] and the reconstructed signal difference [s_r(k)]. (10), (13)	66	00B0
8.	COMPUTE SCALE FACTOR	Calculate the updates for the scale-factor adaptation. (4), (5)	33	0022
9.	COMPUTE SPEED CONTROL	Calculate the update for the speed-control adaptation. (8), (9), (7)	30	001C
10.	COMPUTE PREDICTOR ADAPTATION	Calculate the updates for the adaptive predictor filter coefficients. (18), (16), (17), (14), (15)	111	0074
11.	WAIT	Spin until the next sample is available.	2+	0004

Table 8. CCITT Receiver

Order	Function	Description	CPU Clocks	Program Memory (Words)
1.	INPUT ADPCM	Read the ADPCM input [$I(k)$].	10	0009
2.	COMPUTE SIGNAL ESTIMATE	Calculate the signal estimate [$s_e(k)$] from the previous data samples [$d_q(k)$] and reconstructed samples [$s_r(k)$] through the predictor filter. (12), (11)	396	0167
3.	COMPUTE ADAPTIVE QUANTIZER	Calculate speed control [$a_l(k)$] and quantizer scale factor [$y(k)$] from past quantizer output [$I(k)$]. (6), (3)	30	001E
4.	COMPUTE QUANTIZED DIFFERENCE AND RECON-STRUCTED SIGNAL	Calculate the inverse of the adaptively quantized signal [$d_q(k)$] and the reconstructed signal [$s_r(k)$]. (10), (13)	66	00B0
5.	COMPUTE SCALE FACTOR	Calculate the updates for the scale-factor adaptation. (4), (5)	33	0022
6.	COMPUTE SPEED CONTROL	Calculate the update for the speed-control adaptation. (8), (9), (7)	30	001C
7.	COMPUTE PREDICTOR ADAPTATION	Calculate the updates for the adaptive predictor filter coefficients. (18), (16), (17), (14), (15)	111	0074
8.	COMPUTE LOG-PCM	Convert the reconstructed linear-PCM signal [$s_r(k)$] to a log-PCM signal [$s_p(k)$].	35 μ-law 33 A-law	0074 μ-law 0072 A-law
9.	SYNCHRON-OUS CODING ADJUSTMENT	Calculate an ADPCM signal from the output [$s_p(k)$] and adjust to create [$s_d(k)$] if it differs from [$I(k)$].	63	00DA
10.	OUTPUT PCM	Write the log-PCM output [$s_d(k)$].	4	0003
11.	WAIT	Spin until the next interrupt.	2+	0004

Implementation and Advantages of TMS32010 Architecture

Many of the same features are used in the bit-compatible implementation as were discussed in the full-duplex implementation. Some changes are imperative, since performance to the recommended specification requires executing certain calculations in a floating-point representation. These changes or additions require further modifications in order to limit the required amount of program memory to the internal memory space of the TMS32010.

One of the first observed requirements is that the processor must be capable of doing either A-law or μ-law companding and function as either a transmitter or a receiver. The burden of determining the mode of operation is simplified by selecting one of the four modes from information available at the time of reset, and then executing from one of the four control loops until the next reset. Each loop, therefore, tests the $\overline{\text{BIO}}$ pin to determine when the next input sample is ready, rather than depending on the hardware interrupt.

The requirement of selecting either A-law of μ-law companding also means that a table lookup approach is beyond the program memory capacity. The conversion must be done algorithmically to reduce the amount of memory. Figures 20 and 21 illustrate μ-law companding as it is implemented in this algorithm.

```
XMTMU   IN      SCRACH,ADC
EXPNDU  LAC     SCRACH,8      ; SEEE MMMM 0000 0000
        XOR     KFF00         ; INVERT FROM TRANSMISSION FORMAT
        SACL    TEMP1         ; SAVE VALUE FOR PCM SIGN
        AND     M32767        ; 0EEE MMMM 0000 0000
        SACH    TEMP2,4       ; SAVE EXPONENT VALUE
        AND     M4095         ; 0000 MMMM 0000 0000
        ADD     BIAS,7        ; 0001 MMMM 1000 0000
        SACL    SCRACH
        LAC     TEMP1
        SACH    TEMP1         ; SIGN = FFFF OR 0000
        LACK    SBASE
        ADD     TEMP2,1       ; CALCULATE PCM SHIFT ADDRESS
        CALA
        SUB     BIAS,12       ; 0000000X XXXXXXXX XXXX0000 00000000
        SACH    SAMPLE,4
        LAC     SAMPLE        ; 000XXXXX XXXXXXXX
        XOR     TEMP1         ; POS - DO NOTHING : NEG - 1's COMP
        SUB     TEMP1         ; POS - DO NOTHING : NEG - 2's COMP
        SACL    SAMPLE
        .
        .
        .
*;
SBASE   LAC     SCRACH,5      ; 00000000 0000001M MMM10000 00000000
        RET
        LAC     SCRACH,6      ; 00000000 000001MM MM100000 00000000
        RET
        LAC     SCRACH,7      ; 00000000 00001MMM M1000000 00000000
        RET
        LAC     SCRACH,8      ; 00000000 0001MMMM 10000000 00000000
        RET
        LAC     SCRACH,9      ; 00000000 001MMMM1 00000000 00000000
        RET
        LAC     SCRACH,10     ; 00000000 01MMMM10 00000000 00000000
        RET
        LAC     SCRACH,11     ; 00000000 1MMMM100 00000000 00000000
        RET
        LAC     SCRACH,12     ; 00000001 MMMM1000 00000000 00000000
        RET
```

Figure 20. μ-Law Expansion Code

```
        LAC     SR              ; GET RECONSTRUCTED SIGNAL
*;
*; COMPRESS--CONVERT TO PCM
*;
CMPRSU  SACH    TEMP4           ; SAVE SIGN OF SR
        ABS
        ADD     BIAS            ; ADD BIAS
        SACL    SCRACH          ; SAVE BIASED PCM VALUE
        SUB     ONE,9           ; EXP = 7 - 4 OR 3 - 0
        BGEZ    SCL427
SCL023  ADD     THREE,7         ; EXP = 3 - 2 OR 1 - 0
        BGEZ    SCL223
SCL021  ADD     ONE,6           ; EXP = 1 OR 0
        BGEZ    SCALE1
SCALE0  LAC     M15,1           ; EXP = 0
        AND     SCRACH          ; MASK FOR MANTISSA
        SACL    SCRACH
        ADD     BIAS
        SACL    SAMPLE          ; BIASED QUANTIZED VALUE
        LAC     SCRACH,15
        LARK    0,0
        B       FINI
SCALE1  LAC     M15,2           ; EXP = 1
        AND     SCRACH          ; MASK FOR MANTISSA
        SACL    SCRACH
        ADD     BIAS,1
        SACL    SAMPLE          ; BIASED QUANTIZED VALUE
        LAC     SCRACH,14
        LARK    0,1
        B       FINI
        .
        .
        .
FINI    SACH    SCRACH          ; SAVE NORMALIZED MANTISSA
        LAC     SCRACH
        SAR     0,TEMP1
        ADD     TEMP1,4         ; ADD EXPONENT
CLNUP   ADD     TEMP4,7
        AND     M255
        SACL    SCRACH          ; 2's COMPLEMENT OF MULAW-PCM
        LAC     SAMPLE          ; REMOVE BIAS FROM QUANTIZED VALUE
        SUB     BIAS
        XOR     TEMP4
        SUB     TEMP4
        SACL    SAMPLE          ; 2's COMPLEMENT OF QUANTIZED SAMPLE
*;
        CALL    AQUAN
*;
        CALL    SYNC
*;
        XOR     M255            ; FLIP BITS FOR TRANSMISSION
        SACL    SCRACH
        OUT     SCRACH,DAC
```

Figure 21. μ-Law Compression Code

The predictor filter implementation is also modified from what has been previously presented. In the CCITT recommendation, the processing of the signal through the predictor filter is performed in a floating-point format. This requirement leads to several modifications. First, all input signals to the filter, $d_q(k)$ and $s_r(k)$, must be converted to a floating-point notation. The conversion to this notation is accomplished by a binary search of the original fixed-point word. As previously mentioned, this technique is explained in some detail in the application report, FLOATING-POINT ARITHMETIC WITH THE TMS32010. Second, the filter coefficients, $a_i(k)$ and $b_i(k)$, must also be floated for each sample so that a floating-point multiply can be executed for each filter tap.

Accumulation of the filter taps is carried out in fixed-point notation. Fixing a floating-point number is equivalent to the scaling presented for taking the anti-log of a number. Some of the floating-point results must be left-shifted, while others need to be right-shifted. The shift is accomplished by use of a scaling factor or multiplier, selected by the exponent sum of the floating-point multiply. Positive multipliers are used to indicate what is effectively a right shift with the result being stored from the high half of the accumulator. Negative multipliers indicate that the result is in the low half of the accumulator and is used for values which have been left shifted.

The process of a single filter tap, not including the code to float the signal and the coefficient, is shown in Figure 22.

```
*;*****************************************************************
*;
*;   COMPUTE SEZ -- PARTIAL SIGNAL ESTIMATE
*;
*;   SEZ(k) = B1(k-1)*DQ(k-1) + ... + B6(k-1)*DQ(k-6)
*;
*;       MULTIPLIES ARE DONE IN FLOATING POINT
*;         DQ's ARE STORED IN FLOATING-POINT NOTATION
*;         B's ARE FLOATED EACH PASS
*;
*;       NOTATION: DQnEXP    -- 4 bits + OFFSET
*;                 DQnMAN*8 -- 9 bits
*;                 Bn        -- 16 b TC ; q14
*;                 SEZ       -- 16 b TC ; q0
*;
*;*****************************************************************
*;
SIGDIF  LAC     B6,14     ; COMPUTE B6*DQ5.
        CALL    FLOAT     ; RET/W MANTISSA IN TEMP1; EXP IN ACC.
        ADD     DQ5EXP
        SACL    SUM1
        LAR     0,SUM1    ; EXP OF PRODUCT.
        LT      DQ5MAN    ; DQnMAN SCALED BY 2**3.
        LAC     THREE,7   ; PRODUCT FUDGE FACTOR (48*8).
        MPY     TEMP1
        LTA     *,0       ; B6MAN*(DQ5MAN*8)+(48*8)
        AND     KFF80     ; SAVE ONLY 8 MSB'S.
        SACL    TEMP1
        MPY     TEMP1     ; APPLY SHIFT FACTOR.
        PAC
        BLZ     RS1       ; EXP >= 26
        SACH    SUM1,1    ; EXP <  26
CHK1    ZALS    B6        ; CHECK SIGN OF PRODUCT.
        XOR     SDQ6
        AND     K32768
        BZ      POS1
NEG1    ZAC               ; NEGATE IF NECESSARY.
        SUB     SUM1
```

Figure 22. Predictor Filter Execution

```
        SACL    SUM1
POS1    LAC     B5,14   ; COMPUTE B5*DQ4.
        .
        .
        .
RS1     ABS             ; MAKE POSITIVE BEFORE MASK.
        AND     M32767  ; KEEP LOWER 15 BITS.
        SACL    SUM1    ; SAVE RESULT.
        B       CHK1
```

Figure 22. Predictor Filter Execution (Concluded)

SUMMARY

The TMS32010 provides an efficient solution to transcoding a 64-kbit/s PCM signal to a 32-kbit/s bit stream. Transcoding, as described in this application report, is an effective way to maintain the signal quality provided by 7-bit PCM while reducing the data rate.

The basic ADPCM algorithm has been implemented in two slightly different ways. One solution provides CCITT bit-for-bit compatibility. Using this algorithm, a half-duplex transcoder is created that can transcode either A-law or μ-law signals as either a transmitter or a receiver. No external program memory is required for this implementation, although it does require the use of a 25-MHz TMS32010 microprocessor. The second described solution is particularly attractive since it uses a single, 20.5-MHz TMS32010 microprocessor that requires no external program memory to perform a real-time full-duplex (non-CCITT) channel transcoding.

In selecting one of these two solutions, the primary consideration is the network interfacing requirement. For systems that only have analog interfaces to other parts of the network, the full-duplex solution will provide the best choice. On the other hand, a network that may include a digital interface to other ADPCM transcoders will probably require the CCITT bit-compatible solution. Both solutions provide high-quality signal transcoding.

A complete assembled code listing is provided in the appendix of this report and is also available in 1600-BPI VAX/VMS tape format. The software may be purchased by ordering the TMS32010 Software Exchange Library, TMDC3240212-18, from Texas Instruments. For further information, please contact your nearest TI sales representative.

REFERENCES

"Recommendation G.721, 32 kbit/s Adaptive Differential Pulse Code Modulation," *CCITT* (1984).

N.S. Jayant (ed.), *Waveform Quantization and Coding,* IEEE Press (1976).

N.S. Jayant and Peter Noll, *Digital Coding of Waveforms,* Prentice-Hall (1984).

L.R. Rabiner and R.W. Schafer, *Digital Processing of Speech Signals,* Prentice-Hall (1978).

J.C. Bellamy, *Digital Telephony,* John Wiley & Sons (1982).

Bernhard E. Keiser, *Digital Telephony: Speech Digitization,* George Washington University (1981).

Companding Routines for the TMS32010, Texas Instruments Incorporated (1984).

Floating-Point Arithmetic with the TMS32010, Texas Instruments Incorporated (1984).

Implementation of FIR/IIR Filters with the TMS32010, Texas Instruments Incorporated (1984).

TMS32010 User's Guide, Texas Instruments Incorporated (1983).

Appendix

ADPCM Assembly Language Programs

```
                   COPY    INPUT.ASM
                   IDT     'CCITT'
                   OPTION  XREF
               *;
               *;*******************************************************
               *;*******************************************************
               *; This is the source module for a half-duplex CCITT
               *; compatible 32-kbps ADPCM speech system. The transmitter
               *; assumes that log-PCM data will be available at each
               *; interrupt (every 125 microseconds) in the lower 8 bits of
               *; the data bus via I/O port 1 and it supplies ADPCM data on
               *; the lower 4 bits of the bus via port 2. The receiver does
               *; the inverse.  An interrupt in this case is determined by
               *; polling the BIO line, with a low signal level signifying
               *; the presence of a new data sample.
               *;
               *; The 'R' reset function in the CCITT spec is implemented
               *; with a hardware reset. At the time of reset, it is
               *; assumed that the operating mode has been established and
               *; input via the upper two bits of the data bus.  The bit
               *; condition is read from port 0 so as not to disrupt any
               *; pending data sample on either of the other two ports.
               *; Since it is anticipated that the mode pins will be
               *; selected and maintained in a manner similar to a hardwire
               *; selection, the actual port from which the mode is read
               *; is arbitrary.
               *;
               *;*******************************************************
               *;
               *; System I/O channel assignments
               *;
          0001 ADC     EQU     1        ; codec input
          0001 DAC     EQU     1        ; codec output
          0002 CCITT   EQU     2        ; adpcm output/input
          0000 CTL     EQU     0        ; control input to select mode
               *;                       ; 0000 = mulaw transmitter
               *;                       ; 4000 = mulaw receiver
               *;                       ; 8000 = alaw transmitter
               *;                       ; C000 = alaw receiver
               *;
               *;*******************************************************
     0000              AORG    0
               *;
     0000 F900         B       RESET    ; power-up reset
     0001 0542
               *;
               *;*******************************************************
               *; INTERRUPT HANDLING ROUTINE -- SYSTEM HANDLES CODEC
               *; SAMPLES ON A SAMPLE BY SAMPLE BASIS.
               *;*******************************************************
               *;
     0002 F900 INTRPT B        INTRPT
     0003 0002
               *;
```

```
               *;
               *;  MU-LAW TRANSMITTER
               *;
     0004 411B XMTMU   IN      SCRACH,ADC  ; input mu-law PCM
               *;
               *;*******************************************************
               *; MU-LAW TO LINEAR PCM EXPANSION
               *;
               *;      INPUT:  MU-LAW PCM SAMPLE  -- S  (SCRACH)
               *;
               *;      OUTPUT: LINEAR PCM SAMPLE  -- SL (SAMPLE)
               *;
               *;      NOTATION:  S  -- 8b SM (Q4)
               *;                 SL -- 14b TC (Q0)
               *;
               *;*******************************************************
               *;
               *;                  S    +--------+  SL
               *;              ------->| EXPAND |------->
               *;                       +--------+
               *;
               *;*******************************************************
               *;
     0005 281B EXPNDU LAC      SCRACH,8     ; seee mmmm 0000 0000
     0006 7875        XOR      KFF00        ; SEEE MMMM 0000 0000
     0007 5021        SACL     TEMP1        ; save value for PCM sign
     0008 7974        AND      M32767       ; 0EEE MMMM 0000 0000
     0009 5C22        SACH     TEMP2,4      ; save exponent value
     000A 7972        AND      M4095        ; 0000 MMMM 0000 0000
     000B 074D        ADD      BIAS,7       ; 0001 MMMM 1000 0000
     000C 501B        SACL     SCRACH
     000D 2021        LAC      TEMP1
     000E 5821        SACH     TEMP1        ; sign = FFFF or 0000
     000F 7E24        LACK     SBASE
     0010 0122        ADD      TEMP2,1      ; calculate PCM shift address
     0011 7F8C        CALA
     0012 1C4D        SUB      BIAS,12 ; 0000000X XXXXXXXX XXXX0000 00000000
     0013 5C26        SACH     SAMPLE,4
     0014 2026        LAC      SAMPLE       ; 000X XXXX XXXX XXXX
     0015 7821        XOR      TEMP1        ; pos - no change : neg - 1's compl
     0016 1021        SUB      TEMP1        ; pos - no change : neg - 2's compl
     0017 5026        SACL     SAMPLE
               *;
               *; Now convert PCM value in SAMPLE to ADPCM value in I
               *;
     0018 F800 GETI   CALL     SIGDIF
     0019 01B3
     001A F800        CALL     AQUAN
     001B 02AA
     001C 5001        SACL     I
     001D 4A01        OUT      I,CCITT      ; output ADPCM
     001E F800        CALL     PRDICT
     001F 0355
     0020 F600 MULAWX BIOZ     XMTMU        ; wait for next sample
     0021 0004
     0022 F900        B        MULAWX
     0023 0020
```

```
                *;
0024 251B  SBASE  LAC     SCRACH,5  ; 00000000 0000001M MMM10000 00000000
0025 7F8D         RET
0026 261B         LAC     SCRACH,6  ; 00000000 000001MM MM100000 00000000
0027 7F8D         RET
0028 271B         LAC     SCRACH,7  ; 00000000 00001MMM M1000000 00000000
0029 7F8D         RET
002A 281B         LAC     SCRACH,8  ; 00000000 0001MMMM 10000000 00000000
002B 7F8D         RET
002C 291B         LAC     SCRACH,9  ; 00000000 001MMMM1 00000000 00000000
002D 7F8D         RET
002E 2A1B         LAC     SCRACH,10 ; 00000000 01MMMM10 00000000 00000000
002F 7F8D         RET
0030 2B1B         LAC     SCRACH,11 ; 00000000 1MMMM100 00000000 00000000
0031 7F8D         RET
0032 2C1B         LAC     SCRACH,12 ; 00000001 MMMM1000 00000000 00000000
0033 7F8D         RET
```

```
                *;
                *;  A-LAW TRANSMITTER
                *;
0034 411B  XMTA   IN      SCRACH,ADC   ; input A-law PCM
                *;
                *;**********************************************************
                *; A-LAW TO LINEAR PCM EXPANSION
                *;
                *;      INPUT:  A-LAW PCM SAMPLE  -- S  (SCRACH)
                *;
                *;      OUTPUT: LINEAR PCM SAMPLE  -- SL (SAMPLE)
                *;
                *;      NOTATION:  S  -- 8b SM (Q4)
                *;                 SL -- 14b TC (Q0)
                *;
                *;**********************************************************
                *;
                *;                S     +--------+  SL
                *;               ------->| EXPAND |------->
                *;                      +--------+
                *;
                *;**********************************************************
                *;
0035 281B  EXPNDA LAC     SCRACH,8     ; sEEE MMMM 0000 0000
0036 7848         XOR     K32768       ; SEEE MMMM 0000 0000
0037 5021         SACL    TEMP1        ; save value for PCM sign
0038 7974         AND     M32767       ; 0EEE MMMM 0000 0000
0039 5C22         SACH    TEMP2,4      ; save exponent value
003A 7972         AND     M4095        ; 0000 MMMM 0000 0000
003B 501B         SACL    SCRACH
003C 2021         LAC     TEMP1
003D 5821         SACH    TEMP1        ; sign = FFFF or 0000
003E 7E52         LACK    SBASEA
003F 0222         ADD     TEMP2,2      ; calculate PCM shift address
0040 7F8C         CALA
0041 5C26         SACH    SAMPLE,4
0042 2026         LAC     SAMPLE       ; 000X XXXX XXXX XXXX
0043 7821         XOR     TEMP1        ; pos - no change : neg - 1's compl
0044 1021         SUB     TEMP1        ; pos - no change : neg - 2's compl
0045 5026         SACL    SAMPLE
                *
                *;
                *; Now convert PCM value in SAMPLE to ADPCM value in I
                *;
0046 F800         CALL    SIGDIF
0047 01B3
0048 F800         CALL    AQUAN
0049 02AA
004A 5001         SACL    I
004B 4A01         OUT     I,CCITT      ; output ADPCM
004C F800         CALL    PRDICT
004D 0355
004E F600  ALAWX  BIOZ    XMTA         ; wait for next sample
004F 0034
0050 F900         B       ALAWX
0051 004E
                *;
```

```
0052 261B  SBASEA LAC       SCRACH,6   ; 00000000 000000MM MM000000 00000000
0053 0D4C         ADD       ONE,13     ; 00000000 000000MM MM100000 00000000
0054 7F8D         RET
0055 7F80         NOP
0056 261B         LAC       SCRACH,6   ; 00000000 000001MM MM100000 00000000
0057 0671         ADD       BIASA,6    ; 00000000 000001MM MM100000 00000000
0058 7F8D         RET
0059 7F80         NOP
005A 271B         LAC       SCRACH,7   ; 00000000 00000MMM M0000000 00000000
005B 0771         ADD       BIASA,7    ; 00000000 00001MMM M1000000 00000000
005C 7F8D         RET
005D 7F80         NOP
005E 281B         LAC       SCRACH,8   ; 00000000 0000MMMM 00000000 00000000
005F 0871         ADD       BIASA,8    ; 00000000 0001MMMM 10000000 00000000
0060 7F8D         RET
0061 7F80         NOP
0062 291B         LAC       SCRACH,9   ; 00000000 000MMMM0 00000000 00000000
0063 0971         ADD       BIASA,9    ; 00000000 001MMMM1 00000000 00000000
0064 7F8D         RET
0065 7F80         NOP
0066 2A1B         LAC       SCRACH,10  ; 00000000 00MMMM00 00000000 00000000
0067 0A71         ADD       BIASA,10   ; 00000000 01MMMM10 00000000 00000000
0068 7F8D         RET
0069 7F80         NOP
006A 2B1B         LAC       SCRACH,11  ; 00000000 0MMMM000 00000000 00000000
006B 0B71         ADD       BIASA,11   ; 00000000 1MMMM100 00000000 00000000
006C 7F8D         RET
006D 7F80         NOP
006E 2C1B         LAC       SCRACH,12  ; 00000000 MMMM0000 00000000 00000000
006F 0C71         ADD       BIASA,12   ; 00000001 MMMM1000 00000000 00000000
0070 7F8D         RET
```

```
                 *;
                 *;  MU-LAW RECEIVER
                 *;
0071 4201        RCVMU  IN        I,CCITT       ; input ADPCM
                 *;
0072 2001               LAC       I             ; determine magnitude of ADPCM
0073 5002               SACL      IM
0074 134C               SUB       ONE,3
0075 FA00               BLZ       DO32KU
0076 007A
0077 2002               LAC       IM
0078 786D               XOR       M15
0079 5002               SACL      IM
                 *;
                 *; compute pcm output
                 *;
007A F800        DO32KU CALL      SIGDIF
007B 01B3
007C F800               CALL      PRDICT
007D 0355
                 *;
                 *;*****************************************************************
                 *; LINEAR TO U-LAW PCM COMPRESSION/U-LAW TO LINEAR EXPANSION
                 *;
                 *;     INPUT:  LINEAR PCM SAMPLE  -- SR
                 *;
                 *;     OUTPUT: A-LAW PCM SAMPLE   -- SP  (SCRACH)
                 *;             LINEAR PCM SAMPLE  -- SLX (SAMPLE)
                 *;
                 *;     NOTATION:  SR  -- 16b TC (Q0)
                 *;                SP  -- 8b SM (Q4)
                 *;                SLX -- 14b TC (Q0)
                 *;
                 *;*****************************************************************
                 *;
                 *;     SR   +----------+       SP    +--------+  SLX
                 *;    ------>! COMPRESS !-----*------>! EXPAND !------->
                 *;          +----------+      !      +--------+
                 *;                            !                   SP
                 *;                            +--------------------->
                 *;
                 *;*****************************************************************
                 *;
007E 2013               LAC       SR            ; get reconstructed signal
                 *;
                 *; compress--convert to pcm
                 *;
007F 5824        CMPRSU SACH      TEMP4         ; save sign of SR
0080 7F88               ABS
0081 004D               ADD       BIAS          ; add bias
0082 501B               SACL      SCRACH        ; save biased PCM value
0083 194C               SUB       ONE,9         ; exp = 7 - 4 or 3 - 0
0084 FD00               BGEZ      SCL427
0085 00B3
0086 077D        SCL023 ADD       THREE,7       ; exp = 3 - 2 or 1 - 0
0087 FD00               BGEZ      SCL223
0088 009E
```

```
A0259 0089 064C  SCL021 ADD    ONE,6         ; exp = 1 or 0
A0260 008A FD00         BGEZ   SCALE1
      008B 0095
A0261 008C 216D  SCALE0 LAC    M15,1         ; exp = 0
A0262 008D 791B         AND    SCRACH        ; mask for mantissa
A0263 008E 501B         SACL   SCRACH
A0264 008F 004D         ADD    BIAS
A0265 0090 5026         SACL   SAMPLE        ; biased quantized value
A0266 0091 2F1B         LAC    SCRACH,15
A0267 0092 7000         LARK   0,0
A0268 0093 F900         B      FINI
      0094 00E6
A0269 0095 226D  SCALE1 LAC    M15,2         ; exp = 1
A0270 0096 791B         AND    SCRACH        ; mask for mantissa
A0271 0097 501B         SACL   SCRACH
A0272 0098 014D         ADD    BIAS,1
A0273 0099 5026         SACL   SAMPLE        ; biased quantized value
A0274 009A 2E1B         LAC    SCRACH,14
A0275 009B 7001         LARK   0,1
A0276 009C F900         B      FINI
      009D 00E6
A0277 009E 174C  SCL223 SUB    ONE,7         ; exp = 3 or 2
A0278 009F FD00         BGEZ   SCALE3
      00A0 00AA
A0279 00A1 236D  SCALE2 LAC    M15,3         ; exp = 2
A0280 00A2 791B         AND    SCRACH        ; mask for mantissa
A0281 00A3 501B         SACL   SCRACH
A0282 00A4 024D         ADD    BIAS,2
A0283 00A5 5026         SACL   SAMPLE        ; biased quantized value
A0284 00A6 2D1B         LAC    SCRACH,13
A0285 00A7 7002         LARK   0,2
A0286 00A8 F900         B      FINI
      00A9 00E6
A0287 00AA 246D  SCALE3 LAC    M15,4         ; exp = 3
A0288 00AB 791B         AND    SCRACH        ; mask for mantissa
A0289 00AC 501B         SACL   SCRACH
A0290 00AD 034D         ADD    BIAS,3
A0291 00AE 5026         SACL   SAMPLE        ; biased quantized value
A0292 00AF 2C1B         LAC    SCRACH,12
A0293 00B0 7003         LARK   0,3
A0294 00B1 F900         B      FINI
      00B2 00E6
A0295 00B3 197D  SCL427 SUB    THREE,9       ; exp = 7 - 6 or 5 - 4
A0296 00B4 FD00         BGEZ   SCL627
      00B5 00CB
A0297 00B6 0A4C  SCL425 ADD    ONE,10        ; exp = 5 or 4
A0298 00B7 FD00         BGEZ   SCALE5
      00B8 00C2
A0299 00B9 256D  SCALE4 LAC    M15,5         ; exp = 4
A0300 00BA 791B         AND    SCRACH        ; mask for mantissa
A0301 00BB 501B         SACL   SCRACH
A0302 00BC 044D         ADD    BIAS,4
A0303 00BD 5026         SACL   SAMPLE        ; biased quantized value
A0304 00BE 2B1B         LAC    SCRACH,11
A0305 00BF 7004         LARK   0,4
A0306 00C0 F900         B      FINI
      00C1 00E6
```

```
A0307 00C2 266D  SCALE5 LAC    M15,6         ; exp = 5
A0308 00C3 791B         AND    SCRACH        ; mask for mantissa
A0309 00C4 501B         SACL   SCRACH
A0310 00C5 054D         ADD    BIAS,5
A0311 00C6 5026         SACL   SAMPLE        ; biased quantized value
A0312 00C7 2A1B         LAC    SCRACH,10
A0313 00C8 7005         LARK   0,5
A0314 00C9 F900         B      FINI
      00CA 00E6
A0315 00CB 1B4C  SCL627 SUB    ONE,11        ; exp = 7 or 6
A0316 00CC FD00         BGEZ   SCALE7
      00CD 00D7
A0317 00CE 276D  SCALE6 LAC    M15,7         ; exp = 6
A0318 00CF 791B         AND    SCRACH        ; mask for mantissa
A0319 00D0 501B         SACL   SCRACH
A0320 00D1 064D         ADD    BIAS,6
A0321 00D2 5026         SACL   SAMPLE        ; biased quantized value
A0322 00D3 291B         LAC    SCRACH,9
A0323 00D4 7006         LARK   0,6
A0324 00D5 F900         B      FINI
      00D6 00E6
A0325 00D7 1C4C  SCALE7 SUB    ONE,12        ; exp = 7
A0326 00D8 FA00         BLZ    NORMAL        ; mag > 8191 ?
      00D9 00DF
A0327 00DA 2767  SATCH  LAC    K63,7
A0328 00DB 5026         SACL   SAMPLE        ; save max biased quantized value
A0329 00DC 7E7F         LACK   127           ; set maximum mulaw magnitude
A0330 00DD F900         B      CLNUP
      00DE 00EA
A0331 00DF 286D  NORMAL LAC    M15,8
A0332 00E0 791B         AND    SCRACH        ; mask for mantissa
A0333 00E1 501B         SACL   SCRACH
A0334 00E2 074D         ADD    BIAS,7
A0335 00E3 5026         SACL   SAMPLE        ; biased quantized value
A0336 00E4 281B         LAC    SCRACH,8
A0337 00E5 7007         LARK   0,7
A0338 00E6 581B  FINI   SACH   SCRACH        ; save normalized mantissa
A0339 00E7 201B         LAC    SCRACH
A0340 00E8 3021         SAR    0,TEMP1
A0341 00E9 0421         ADD    TEMP1,4       ; add exponent
A0342 00EA 0724  CLNUP  ADD    TEMP4,7
A0343 00EB 7947         AND    M255
A0344 00EC 501B         SACL   SCRACH        ; signed magnitude of mulaw-PCM
A0345 00ED 2026         LAC    SAMPLE        ; remove bias from quantized value
A0346 00EE 104D         SUB    BIAS
A0347 00EF 7824         XOR    TEMP4
A0348 00F0 1024         SUB    TEMP4
A0349 00F1 5026         SACL   SAMPLE        ; 2's complement of quantized sample
A0350                 *;
A0351 00F2 F800         CALL   AQUAN
      00F3 02AA
A0352                 *;
A0353 00F4 F800         CALL   SYNC
      00F5 0188
A0354                 *;
A0355 00F6 7847         XOR    M255          ; flip bits for transmission
A0356 00F7 501B         SACL   SCRACH
```

```
A0357 00F8 491B         OUT     SCRACH,DAC  ; output mu-law PCM
A0358 00F9 F600  MULAWR BIOZ    RCVMU       ; wait for next sample
      00FA 0071
A0359 00FB F900         B       MULAWR
      00FC 00F9
```

```
A0361                  *;
A0362                  *;  A-LAW RECEIVER
A0363                  *;
A0364 00FD 4201  RCVA   IN      I,CCITT      ; input ADPCM
A0365                  *;
A0366 00FE 2001         LAC     I            ; determine magnitude of ADPCM
A0367 00FF 5002         SACL    IM
A0368 0100 134C         SUB     ONE,3
A0369 0101 FA00         BLZ     DO32KA
      0102 0106
A0370 0103 2002         LAC     IM
A0371 0104 786D         XOR     M15
A0372 0105 5002         SACL    IM
A0373                  *;
A0374                  *; compute pcm output
A0375                  *;
A0376 0106 F800  DO32KA CALL    SIGDIF
      0107 01B3
A0377 0108 F800         CALL    PRDICT
      0109 0355
A0378                  *;
A0379                  *;*************************************************************
A0380                  *; LINEAR TO A-LAW PCM COMPRESSION/A-LAW TO LINEAR EXPANSION
A0381                  *;
A0382                  *;      INPUT:  LINEAR PCM SAMPLE  -- SR
A0383                  *;
A0384                  *;      OUTPUT: A-LAW PCM SAMPLE   -- SP  (SCRACH)
A0385                  *;              LINEAR PCM SAMPLE  -- SLX (SAMPLE)
A0386                  *;
A0387                  *;      NOTATION:  SR  -- 16b TC (Q0)
A0388                  *;                 SP  -- 8b SM (Q4)
A0389                  *;                 SLX -- 14b TC (Q0)
A0390                  *;
A0391                  *;*************************************************************
A0392                  *;
A0393                  *;      SR   +----------+        SP    +---------+  SLX
A0394                  *;     ------>! COMPRESS !------*------->! EXPAND !------->
A0395                  *;           +----------+       !       +---------+
A0396                  *;                               !                     SP
A0397                  *;                               +---------------------->
A0398                  *;
A0399                  *;*************************************************************
A0400                  *;
A0401 010A 2013         LAC     SR           ; get reconstructed signal
A0402                  *;
A0403                  *; compress--convert to pcm
A0404                  *;
A0405 010B 5824  CMPRSA SACH    TEMP4        ; save sign of SR
A0406 010C 7F88         ABS
A0407 010D 0024         ADD     TEMP4        ; add 1 for negative vals
A0408 010E 501B         SACL    SCRACH       ; save PCM value
A0409 010F 194C         SUB     ONE,9        ; exp = 7 - 4 or 3 - 0
A0410 0110 FD00         BGEZ    SCL4T7
      0111 013F
A0411 0112 077D  SCL0T3 ADD     THREE,7      ; exp = 3 - 2 or 1 - 0
A0412 0113 FD00         BGEZ    SCL2T3
      0114 012A
```

```
A0413 0115 064C  SCLOT1 ADD    ONE,6         ; exp = 1 or 0
A0414 0116 FD00         BGEZ   SCAL1A
      0117 0121
A0415 0118 226D  SCAL0A LAC    M15,2         ; exp = 0
A0416 0119 791B         AND    SCRACH        ; mask for mantissa
A0417 011A 501B         SACL   SCRACH
A0418 011B 014C         ADD    ONE,1
A0419 011C 5026         SACL   SAMPLE        ; quantized value
A0420 011D 2E1B         LAC    SCRACH,14
A0421 011E 7000         LARK   0,0
A0422 011F F900         B      FINISH
      0120 0172
A0423 0121 226D  SCAL1A LAC    M15,2         ; exp = 1
A0424 0122 791B         AND    SCRACH        ; mask for mantissa
A0425 0123 501B         SACL   SCRACH
A0426 0124 014D         ADD    BIAS,1
A0427 0125 5026         SACL   SAMPLE        ; quantized value
A0428 0126 2E1B         LAC    SCRACH,14
A0429 0127 7001         LARK   0,1
A0430 0128 F900         B      FINISH
      0129 0172
A0431 012A 174C  SCL2T3 SUB    ONE,7         ; exp = 3 or 2
A0432 012B FD00         BGEZ   SCAL3A
      012C 0136
A0433 012D 236D  SCAL2A LAC    M15,3         ; exp = 2
A0434 012E 791B         AND    SCRACH        ; mask for mantissa
A0435 012F 501B         SACL   SCRACH
A0436 0130 024D         ADD    BIAS,2
A0437 0131 5026         SACL   SAMPLE        ; quantized value
A0438 0132 2D1B         LAC    SCRACH,13
A0439 0133 7002         LARK   0,2
A0440 0134 F900         B      FINISH
      0135 0172
A0441 0136 246D  SCAL3A LAC    M15,4         ; exp = 3
A0442 0137 791B         AND    SCRACH        ; mask for mantissa
A0443 0138 501B         SACL   SCRACH
A0444 0139 034D         ADD    BIAS,3
A0445 013A 5026         SACL   SAMPLE        ; quantized value
A0446 013B 2C1B         LAC    SCRACH,12
A0447 013C 7003         LARK   0,3
A0448 013D F900         B      FINISH
      013E 0172
A0449 013F 197D  SCL4T7 SUB    THREE,9       ; exp = 7 - 6 or 5 - 4
A0450 0140 FD00         BGEZ   SCL6T7
      0141 0157
A0451 0142 0A4C  SCL4T5 ADD    ONE,10        ; exp = 5 or 4
A0452 0143 FD00         BGEZ   SCAL5A
      0144 014E
A0453 0145 256D  SCAL4A LAC    M15,5         ; exp = 4
A0454 0146 791B         AND    SCRACH        ; mask for mantissa
A0455 0147 501B         SACL   SCRACH
A0456 0148 044D         ADD    BIAS,4
A0457 0149 5026         SACL   SAMPLE        ; quantized value
A0458 014A 2B1B         LAC    SCRACH,11
A0459 014B 7004         LARK   0,4
A0460 014C F900         B      FINISH
      014D 0172
```

```
A0461 014E 266D  SCAL5A LAC    M15,6         ; exp = 5
A0462 014F 791B         AND    SCRACH        ; mask for mantissa
A0463 0150 501B         SACL   SCRACH
A0464 0151 054D         ADD    BIAS,5
A0465 0152 5026         SACL   SAMPLE        ; quantized value
A0466 0153 2A1B         LAC    SCRACH,10
A0467 0154 7005         LARK   0,5
A0468 0155 F900         B      FINISH
      0156 0172
A0469 0157 1B4C  SCL6T7 SUB    ONE,11        ; exp = 7 or 6
A0470 0158 FD00         BGEZ   SCAL7A
      0159 0163
A0471 015A 276D  SCAL6A LAC    M15,7         ; exp = 6
A0472 015B 791B         AND    SCRACH        ; mask for mantissa
A0473 015C 501B         SACL   SCRACH
A0474 015D 064D         ADD    BIAS,6
A0475 015E 5026         SACL   SAMPLE        ; quantized value
A0476 015F 291B         LAC    SCRACH,9
A0477 0160 7006         LARK   0,6
A0478 0161 F900         B      FINISH
      0162 0172
A0479 0163 1C4C  SCAL7A SUB    ONE,12        ; exp = 7
A0480 0164 FA00         BLZ    NORMLA        ; mag > 8191 ?
      0165 016B
A0481 0166 2767  SATCHA LAC    K63,7
A0482 0167 5026         SACL   SAMPLE        ; save maximum quantized value
A0483 0168 7E7F         LACK   127           ; save maximum alaw magnitude
A0484 0169 F900         B      CLNUPA
      016A 0176
A0485 016B 286D  NORMLA LAC    M15,8
A0486 016C 791B         AND    SCRACH        ; mask for mantissa
A0487 016D 501B         SACL   SCRACH
A0488 016E 074D         ADD    BIAS,7
A0489 016F 5026         SACL   SAMPLE        ; quanitzed value
A0490 0170 281B         LAC    SCRACH,8
A0491 0171 7007         LARK   0,7
A0492 0172 581B  FINISH SACH   SCRACH        ; save normalized mantissa
A0493 0173 201B         LAC    SCRACH
A0494 0174 3021         SAR    0,TEMP1
A0495 0175 0421         ADD    TEMP1,4       ; add exponent
A0496 0176 0724  CLNUPA ADD    TEMP4,7
A0497 0177 7947         AND    M255
A0498 0178 501B         SACL   SCRACH        ; signed magnitude of alaw-PCM
A0499 0179 2026         LAC    SAMPLE
A0500 017A 7824         XOR    TEMP4
A0501 017B 1024         SUB    TEMP4
A0502 017C 5026         SACL   SAMPLE        ; 2's complement of quantized sample
A0503                *;
A0504 017D F800         CALL   AQUAN
      017E 02AA
A0505                *;
A0506 017F F800         CALL   SYNC
      0180 0188
A0507                *;
A0508 0181 787F         XOR    M0080         ; flip bits for transmission
A0509 0182 501B         SACL   SCRACH
A0510 0183 491B         OUT    SCRACH,DAC    ; output A-law PCM
```

```
A0511 0184 F600  ALAWR  BIOZ     RCVA         ; wait for next sample
      0185 00FD
A0512 0186 F900         B        ALAWR
      0187 0184
```

```
A0514            *;
A0515            *;**************************************************************
A0516            *; SYNCHRONOUS CODING ADJUSTMENT
A0517            *;
A0518            *;     INPUT:  LOG PCM SAMPLE      -- SP  (SCRACH)
A0519            *;             RECEIVED ADPCM      -- I
A0520            *;             REGENERATED ADPCM   -- ID  (TEMP1)
A0521            *;
A0522            *;     OUTPUT: ADJUSTED LOG PCM    -- SD  (SCRACH)
A0523            *;
A0524            *;     NOTATION:  I    -- 4b SM (Q0)
A0525            *;                ID   -- 4b SM (Q0)
A0526            *;                SP   -- 8b SM (Q4)
A0527            *;                SD   -- 8b SM (Q4)
A0528            *;
A0529            *;**************************************************************
A0530            *;
A0531            *;                 I     +-------+
A0532            *;              ------->|       |
A0533            *;                ID    |       |    SD
A0534            *;              ------->| SYNC  |--------->
A0535            *;                SP    |       |
A0536            *;              ------->|       |
A0537            *;                      +-------+
A0538            *;
A0539            *;**************************************************************
A0540            *;
A0541 0188 7838  SYNC   XOR    EIGHT
A0542 0189 5021         SACL   TEMP1        ; flip the polarity bit in ID
A0543 018A 234C         LAC    ONE,3
A0544 018B 7801         XOR    I            ; flip the polarity bit in I
A0545 018C 1021         SUB    TEMP1        ; IM - ID
A0546 018D FA00         BLZ    IDGTIM       ; ID > IM ... -
      018E 01A0
A0547 018F FF00         BZ     IDEQIM       ; ID = IM
      0190 01B1
A0548 0191 201B  IDLTIM LAC    SCRACH       ; ID < IM ... +
A0549 0192 106F         SUB    M127
A0550 0193 FC00         BGZ    SUBONE
      0194 019B
A0551 0195 FF00         BZ     MAXPOS
      0196 0199
A0552 0197 074C         ADD    ONE,7        ; SD = SP + 1 : 0 <= SP < 127
A0553 0198 7F8D         RET                 ,
A0554 0199 7E7F  MAXPOS LACK   127          ; SD = 127    : SP = 127
A0555 019A 7F8D         RET
A0556 019B 104C  SUBONE SUB    ONE
A0557 019C FF00         BZ     ANOMLE       ; SD = 0      : SP = 128
      019D 019F
A0558 019E 006F         ADD    M127         ; SD = SP + 1 : 255 >= SP > 128
A0559 019F 7F8D  ANOMLE RET
A0560 01A0 201B  IDGTIM LAC    SCRACH
A0561 01A1 174C         SUB    ONE,7
A0562 01A2 FD00         BGEZ   ADDONE
      01A3 01A8
A0563 01A4 006F         ADD    M127         ; SD = SP - 1 : 0 < SP <= 127
A0564 01A5 FA00         BLZ    ANOMLY
```

```
01A6 01AF
01A7 7F8D                RET
01A8 106F   ADDONE SUB   M127
01A9 FD00          BGEZ  MAXNEG
01AA 01AD
01AB 084C          ADD   ONE,8       ; SD = SP - 1 : 255 > SP >= 128
01AC 7F8D          RET
01AD 7EFF   MAXNEG LACK  255         ; SD = 255    : SP = 255
01AE 7F8D          RET
01AF 7E80   ANOMLY LACK  128         ; SD = 128    : SP = 0
01B0 7F8D          RET
01B1 201B   IDEQIM LAC   SCRACH      ; SD = SP
01B2 7F8D          RET
```

```
                         COPY     SIGDIF.ASM
                   *;
                   *;*************************************************************
                   *; SIGDIF
                   *;
                   *; Implements the following modules (per CCITT spec):
                   *;
                   *;     DELAY D -- delay of DQ and SR derivatives
                   *;     FMULT   -- Bn * DQn, An * SRn
                   *;     DELAY A -- (implicit in use of last frames data)
                   *;     ACCUM   -- Accumulate partial products for SEZ, SE
                   *;
                   *;     LIMA    -- compute AL(k)
                   *;     MIX     -- compute Y(k)
                   *;
                   *;*************************************************************
                   *;
                   *; compute SEZ-- partial signal estimate
                   *;
                   *;  SEZ(k) = B1(k-1)*DQ(k-1) + ... + B6(k-1)*DQ(k-6)
                   *;
                   *;     Multiplies are done in floating pt
                   *;       DQ's are stored in f.p. notation
                   *;       B's are floated each pass
                   *;
                   *;*************************************************************
                   *; FLOATING POINT MULTIPLY (FMULT)
                   *;
                   *;     INPUT:  QUANTIZED DIFFERENCE   -- DQn (DQnEXP/DQnMAN)
                   *;             PREDICTOR COEFFICIENTS -- Bn
                   *;
                   *;     OUTPUT: FILTER TAP OUTPUTS     -- WBn (SUMn)
                   *;
                   *;     NOTATION: DQnEXP    -- 4b + offset
                   *;               DQnMAN*8  -- 9b magnitude
                   *;               Bn        -- 16b TC (Q14)
                   *;               SUMn      -- 16b TC (Q1)
                   *;
                   *;*************************************************************
                   *;
                   *;     -1        -1        -1        -1        -1        -1
                   *; DQ  z         z         z         z         z         z
                   *; o--->---o--->---o--->---o--->---o--->---o--->---o
                   *;         ¦         ¦         ¦         ¦         ¦         ¦
                   *;         ¦         ¦         ¦         ¦         ¦         ¦
                   *;       vB1(k)  vB2(k)  vB3(k)  vB4(k)  vB5(k)  vB6(k)
                   *;         ¦         ¦         ¦         ¦         ¦         ¦
                   *;         ¦         ¦         ¦         ¦         ¦         ¦
                   *;         o--->---o--->---o--->---o--->---o--->---o--->---o
                   *;        WB1      WB2      WB3      WB4     WB5      WB6     SEZ
                   *;
                   *;*************************************************************
                   *;
01B3 2E0F   SIGDIF LAC    B6,14    ; compute B6*DQ5
01B4 F800          CALL   FLOAT    ; ret/w mantissa in TEMP1; exp in ac
01B5 04DE
01B6 001A          ADD    DQ5EXP
```

```
B0056 01B7 5022          SACL   SUM1
B0057 01B8 3822          LAR    0,SUM1   ; exp of product offset by table add
B0058 01B9 6A60          LT     DQ5MAN   ; scaled up by 2**3
B0059 01BA 277D          LAC    THREE,7  ; multiply fudge factor
B0060 01BB 6D21          MPY    TEMP1
B0061 01BC 6C80          LTA    *,0      ; mult mant, add 48, fetch shift fac
B0062 01BD 796E          AND    KFF80
B0063 01BE 5021          SACL   TEMP1
B0064 01BF 6D21          MPY    TEMP1    ; apply shift factor = f(exp)
B0065 01C0 7F8E          PAC
B0066 01C1 FA00          BLZ    RS1      ; exp >= 26
      01C2 04B6
B0067 01C3 5922          SACH   SUM1,1   ; exp <  26
B0068 01C4 660F   CHK1   ZALS   B6       ; check sign of product
B0069 01C5 785A          XOR    SDQ6
B0070 01C6 7948          AND    K32768
B0071 01C7 FF00          BZ     POS1
      01C8 01CC
B0072 01C9 7F89   NEG1   ZAC             ; negate if necessary
B0073 01CA 1022          SUB    SUM1
B0074 01CB 5022          SACL   SUM1
B0075 01CC 2E0E   POS1   LAC    B5,14    ; compute B5*DQ4
B0076 01CD F800          CALL   FLOAT    ; ret/w mantissa in TEMP1; exp in ac
      01CE 04DE
B0077 01CF 0019          ADD    DQ4EXP
B0078 01D0 6919          DMOV   DQ4EXP
B0079 01D1 5023          SACL   SUM2
B0080 01D2 3823          LAR    0,SUM2   ; exp of product offset by table add
B0081 01D3 6B5F          LTD    DQ4MAN   ; scaled up by 2**3
B0082 01D4 277D          LAC    THREE,7  ; multiply fudge factor
B0083 01D5 6D21          MPY    TEMP1
B0084 01D6 6C80          LTA    *,0      ; mult mant, add 48, fetch shift fac
B0085 01D7 796E          AND    KFF80
B0086 01D8 5021          SACL   TEMP1
B0087 01D9 6D21          MPY    TEMP1    ; apply shift factor = f(exp)
B0088 01DA 7F8E          PAC
B0089 01DB FA00          BLZ    RS2      ; exp >= 26
      01DC 04BB
B0090 01DD 5923          SACH   SUM2,1   ; exp < 26
B0091 01DE 660E   CHK2   ZALS   B5       ; check sign of product
B0092 01DF 7859          XOR    SDQ5
B0093 01E0 7948          AND    K32768
B0094 01E1 FF00          BZ     POS2
      01E2 01E6
B0095 01E3 7F89   NEG2   ZAC             ; negate if necessary
B0096 01E4 1023          SUB    SUM2
B0097 01E5 5023          SACL   SUM2
B0098 01E6 2E0D   POS2   LAC    B4,14    ; compute B4*DQ3
B0099 01E7 F800          CALL   FLOAT    ; ret/w mantissa in TEMP1; exp in ac
      01E8 04DE
B0100 01E9 0018          ADD    DQ3EXP
B0101 01EA 6918          DMOV   DQ3EXP
B0102 01EB 5025          SACL   SUM3
B0103 01EC 3825          LAR    0,SUM3   ; exp of product offset by table add
B0104 01ED 6B5E          LTD    DQ3MAN   ; scaled up by 2**3
B0105 01EE 277D          LAC    THREE,7  ; multiply fudge factor
B0106 01EF 6D21          MPY    TEMP1
```

```
B0107 01F0 6C80          LTA    *,0      ; mult mant, add 48, fetch shift fac
B0108 01F1 796E          AND    KFF80
B0109 01F2 5021          SACL   TEMP1
B0110 01F3 6D21          MPY    TEMP1    ; apply shift factor = f(exp)
B0111 01F4 7F8E          PAC
B0112 01F5 FA00          BLZ    RS3      ; exp >= 26
      01F6 04C0
B0113 01F7 5925          SACH   SUM3,1   ; exp < 26
B0114 01F8 660D   CHK3   ZALS   B4       ; check sign of product
B0115 01F9 7858          XOR    SDQ4
B0116 01FA 7948          AND    K32768
B0117 01FB FF00          BZ     POS3
      01FC 0200
B0118 01FD 7F89   NEG3   ZAC             ; negate if necessary
B0119 01FE 1025          SUB    SUM3
B0120 01FF 5025          SACL   SUM3
B0121 0200 2E0C   POS3   LAC    B3,14    ; compute B3*DQ2
B0122 0201 F800          CALL   FLOAT    ; ret/w mantissa in TEMP1; exp in ac
      0202 04DE
B0123 0203 0017          ADD    DQ2EXP
B0124 0204 6917          DMOV   DQ2EXP
B0125 0205 501E          SACL   SUM4
B0126 0206 381E          LAR    0,SUM4   ; exp of product offset by table add
B0127 0207 6B5D          LTD    DQ2MAN   ; scaled up by 2**3
B0128 0208 277D          LAC    THREE,7  ; multiply fudge factor
B0129 0209 6D21          MPY    TEMP1
B0130 020A 6C80          LTA    *,0      ; mult mant, add 48, fetch shift fac
B0131 020B 796E          AND    KFF80
B0132 020C 5021          SACL   TEMP1
B0133 020D 6D21          MPY    TEMP1    ; apply shift factor = f(exp)
B0134 020E 7F8E          PAC
B0135 020F FA00          BLZ    RS4      ; exp >= 26
      0210 04C5
B0136 0211 591E          SACH   SUM4,1   ; exp < 26
B0137 0212 660C   CHK4   ZALS   B3       ; check sign of product
B0138 0213 7857          XOR    SDQ3
B0139 0214 7948          AND    K32768
B0140 0215 FF00          BZ     POS4
      0216 021A
B0141 0217 7F89   NEG4   ZAC             ; negate if necessary
B0142 0218 101E          SUB    SUM4
B0143 0219 501E          SACL   SUM4
B0144 021A 2E0B   POS4   LAC    B2,14    ; compute B2*DQ1
B0145 021B F800          CALL   FLOAT    ; ret/w mantissa in TEMP1; exp in ac
      021C 04DE
B0146 021D 0016          ADD    DQ1EXP
B0147 021E 6916          DMOV   DQ1EXP
B0148 021F 501F          SACL   SUM5
B0149 0220 381F          LAR    0,SUM5   ; exp of product offset by table add
B0150 0221 6B5C          LTD    DQ1MAN   ; scaled up by 2**3
B0151 0222 277D          LAC    THREE,7  ; multiply fudge factor
B0152 0223 6D21          MPY    TEMP1
B0153 0224 6C80          LTA    *,0      ; mult mant, add 48, fetch shift fac
B0154 0225 796E          AND    KFF80
B0155 0226 5021          SACL   TEMP1
B0156 0227 6D21          MPY    TEMP1    ; apply shift factor = f(exp)
B0157 0228 7F8E          PAC
```

```
B0158 0229 FA00          BLZ     RS5      ; exp >= 26
      022A 04CA
B0159 022B 591F          SACH    SUM5,1   ; exp < 26
B0160 022C 660B  CHK5    ZALS    B2       ; check sign of product
B0161 022D 7856          XOR     SDQ2
B0162 022E 7948          AND     K32768
B0163 022F FF00          BZ      POS5
      0230 0234
B0164 0231 7F89  NEG5    ZAC              ; negate if necessary
B0165 0232 101F          SUB     SUM5
B0166 0233 501F          SACL    SUM5
B0167 0234 2E0A  POS5    LAC     B1,14    ; compute B1*DQ
B0168 0235 F800          CALL    FLOAT    ; ret/w mantissa in TEMP1; exp in accum
      0236 04DE
B0169 0237 0015          ADD     DQEXP
B0170 0238 6915          DMOV    DQEXP
B0171 0239 5020          SACL    SUM6
B0172 023A 3820          LAR     0,SUM6   ; exp of product offset by table addr
B0173 023B 6B5B          LTD     DQMAN    ; scaled up by 2**3
B0174 023C 277D          LAC     THREE,7  ; multiply fudge factor
B0175 023D 6D21          MPY     TEMP1
B0176 023E 6C80          LTA     *,0      ; mult mant, add 48, fetch shift factor
B0177 023F 796E          AND     KFF80
B0178 0240 5021          SACL    TEMP1
B0179 0241 6D21          MPY     TEMP1    ; apply shift factor = f(exp)
B0180 0242 7F8E          PAC
B0181 0243 FA00          BLZ     RS6      ; exp >= 26
      0244 04CF
B0182 0245 5920          SACH    SUM6,1   ; exp < 26
B0183 0246 660A  CHK6    ZALS    B1       ; check sign of product
B0184 0247 7855          XOR     SDQ1
B0185 0248 7948          AND     K32768
B0186 0249 FF00          BZ      POS6
      024A 024E
B0187 024B 7F89  NEG6    ZAC              ; negate if necessary
B0188 024C 1020          SUB     SUM6
B0189 024D 5020          SACL    SUM6
B0190      024E  POS6    EQU     $
B0191            *;
B0192            *;**********************************************************
B0193            *;
B0194            *; compute SE -- signal estimate
B0195            *;
B0196            *;  SE = A1(k-1)*SR(k-1) + A2(k-1)*SR(k-1) + SEZ(k)
B0197            *;
B0198            *;     Multiplies are dONE in floating pt
B0199            *;       SR's are stored in f.p. notation
B0200            *;       A's are floated each pass
B0201            *;
B0202            *;**********************************************************
B0203            *; FLOATING POINT MULTIPLY (FMULT)
B0204            *;
B0205            *;     INPUT:  RECONSTRUCTED SIGNAL    -- SRn (SRnEXP/SRnMAN)
B0206            *;             PREDICTOR COEFFICIENTS -- An
B0207            *;
B0208            *;     OUTPUT: FILTER TAP OUTPUTS      -- WAn (SUMn+6)
B0209            *;
```

```
B0210            *;     NOTATION: SRnEXP    -- 4b + offset
B0211            *;               SRnMAN*8  -- 9b magnitude
B0212            *;               An        -- 16b TC (Q14)
B0213            *;               SUMn+6    -- 16b TC (Q1)
B0214            *;
B0215            *;**********************************************************
B0216            *;
B0217            *;                -1        -1
B0218            *;            SR  z         z
B0219            *;            o--->---o--->---o
B0220            *;                    |       |
B0221            *;                    |       |
B0222            *;                  vA1(k)  vA2(k)
B0223            *;                    |       |
B0224            *;                    |       |
B0225            *;            o---<---o---<---o---<---o
B0226            *;            SE     WA1     WA2     SEZ
B0227            *;
B0228            *;**********************************************************
B0229            *;
B0230 024E 2E12  GETSE   LAC     A2,14    ; compute A2*SR1
B0231 024F F800          CALL    FLOAT    ; ret/w mantissa in TEMP1; exp in accum
      0250 04DE
B0232 0251 001D          ADD     SR1EXP
B0233 0252 5027          SACL    SUM7
B0234 0253 3827          LAR     0,SUM7   ; exp of product offset by table addr
B0235 0254 6A53          LT      SR1MAN   ; scaled up by 2**3
B0236 0255 277D          LAC     THREE,7  ; multiply fudge factor
B0237 0256 6D21          MPY     TEMP1
B0238 0257 6C80          LTA     *,0      ; mult mant, add 48, fetch shift factor
B0239 0258 796E          AND     KFF80
B0240 0259 5021          SACL    TEMP1
B0241 025A 6D21          MPY     TEMP1    ; apply shift factor = f(exp)
B0242 025B 7F8E          PAC
B0243 025C FA00          BLZ     RS11     ; exp >= 26
      025D 04D4
B0244 025E 5927          SACH    SUM7,1   ; exp < 26
B0245 025F 6612  CHK11   ZALS    A2       ; check sign of product
B0246 0260 7814          XOR     SR1
B0247 0261 7948          AND     K32768
B0248 0262 FF00          BZ      POS11
      0263 0267
B0249 0264 7F89  NEG11   ZAC              ; negate if necessary
B0250 0265 1027          SUB     SUM7
B0251 0266 5027          SACL    SUM7
B0252 0267 2E11  POS11   LAC     A1,14    ; compute A1*SR
B0253 0268 F800          CALL    FLOAT    ; ret/w mantissa in TEMP1; exp in accum
      0269 04DE
B0254 026A 001C          ADD     SREXP
B0255 026B 691C          DMOV    SREXP
B0256 026C 5028          SACL    SUM8
B0257 026D 3828          LAR     0,SUM8   ; exp of product offset by table addr
B0258 026E 6B52          LTD     SRMAN    ; scaled up by 2**3
B0259 026F 277D          LAC     THREE,7  ; multiply fudge factor
B0260 0270 6D21          MPY     TEMP1
B0261 0271 6C80          LTA     *,0      ; mult mant, add 48, fetch shift factor
B0262 0272 796E          AND     KFF80
```

```
0273 5021         SACL    TEMP1
0274 6D21         MPY     TEMP1    ; apply shift factor = f(exp)
0275 7F8E         PAC
0276 FA00         BLZ     RS21     ; exp >= 26
0277 04D9
0278 5928         SACH    SUM8,1   ; exp < 26
0279 6913  CHK21  DMOV    SR
027A 6613         ZALS    SR       ; check sign of product
027B 7811         XOR     A1
027C 7948         AND     K32768
027D FF00         BZ      POS21
027E 0282
027F 7F89  NEG21  ZAC              ; negate if necessary
0280 1028         SUB     SUM8
0281 5028         SACL    SUM8
     0282  POS21  EQU     $
           *;
           *;*******************************************************
           *; ACCUMULATE FILTER TAP OUTPUTS (ACCUM)
           *;
           *;      INPUT:  FILTER TAP OUTPUTS -- WAn & WBn (SUMm)
           *;
           *;      OUTPUT: PARTIAL SUM OF ZEROES FILTER -- SEZ
           *;              SIGNAL ESTIMATE              -- SE
           *;
           *;      NOTATION: SUMm -- 16b TC (Q1)
           *;                SEZ  -- 15b TC (Q0) [sign extended]
           *;                SE   -- 15b TC (Q0) [sign extended]
           *;
           *;*******************************************************
           *;
0282 2F20         LAC     SUM6,15 ; accumulate products
0283 0F1F         ADD     SUM5,15
0284 0F1E         ADD     SUM4,15
0285 0F25         ADD     SUM3,15
0286 0F23         ADD     SUM2,15
0287 0F22         ADD     SUM1,15
0288 5904         SACH    SEZ,1
0289 0F27         ADD     SUM7,15
028A 0F28         ADD     SUM8,15
028B 5903         SACH    SE,1
028C 2F03         LAC     SE,15
028D 5803         SACH    SE
           *;
           *;*******************************************************
           *; limit speed control parameter: AL <= 1.0
           *;
           *;      AL = 1           if APP >  1
           *;      AL = APP         if APP <= 1
           *;
           *;      INPUT:  UNLIMITED SPEED CONTROL -- AP (APP)
           *;
           *;      OUTPUT: LIMITED SPEED CONTROL -- AL
           *;
           *;      NOTATION:  APP -- unsigned 10b (Q8)
           *;                 AL  -- unsigned  7b (Q6)
           *;
```

```
           *;*******************************************************
           *;
028E 2C4C  LIMA   LAC     ONE,12
028F 5006         SACL    AL
0290 2005         LAC     APP                  ; check if APP >=1
0291 184C         SUB     ONE,8
0292 FD00         BGEZ    MIX                  ;APP >= 1
0293 0297
0294 2405         LAC     APP,4
0295 7970         AND     MFFC0
0296 5006         SACL    AL                   ;APP  < 1
           *;
           *;*******************************************************
           *; MIX
           *; form linear combination of fast and slow scale factors
           *;
           *;     Y(k) = (1-AL(k))*YL(k-1) + AL(k)*YU(k-1)
           *;
           *;     INPUT:  SLOW QUANTIZER SCALE FACTOR -- YL (YLL/YLH)
           *;             FAST QUANTIZER SCALE FACTOR -- YU
           *;             LIMITED SPEED CONTROL       -- AL
           *;
           *;     OUTPUT: QUANTIZER SCALE FACTOR          -- Y
           *;             RESCALED QUANTIZER SCALE FACTOR -- YOVER4
           *;
           *;     NOTATION: YL     -- 19b unsigned (Q15)
           *;                         stored as:
           *;                           low 15b -- YLL
           *;                           hi   4b -- YLH
           *;               YU     -- 13b unsigned (Q9)
           *;               AL     -- 7b unsigned  (Q6)
           *;               Y      -- 13b unsigned (Q9)
           *;               YOVER4 -- 11b unsigned (Q7)
           *;
           *;*******************************************************
           *;
0297 2A4A  MIX    LAC     YLL,10               ; shift yl right by 6
0298 5823         SACH    TEMP3
0299 2949         LAC     YLH,9
029A 0023         ADD     TEMP3
029B 5023         SACL    TEMP3                ; YL>>6
029C 204E         LAC     YU
029D 1023         SUB     TEMP3                ; YU-(YL>>6)
029E 5021         SACL    TEMP1
029F 6A06         LT      AL
02A0 6D21         MPY     TEMP1
02A1 7F8E         PAC                          ; AL*(YU-(YL>>6))
02A2 FD00         BGEZ    NONNEG               ; negative truncation
02A3 02A5
02A4 0072         ADD     M4095
02A5 0C23  NONNEG ADD     TEMP3,12
02A6 5C09         SACH    Y,4
02A7 2E09         LAC     Y,14                 ; compute and save y>>2
02A8 5829         SACH    YOVER4
02A9 7F8D         RET                          ; ret from SIGDIF
```

```
                      COPY    AQUAN.ASM
                *;
                *;************************************************************
                *; DIFFERENCE SIGNAL COMPUTATION
                *;
                *;      INPUT:  LINEAR PCM SAMPLE -- SL (SAMPLE)
                *;              SIGNAL ESTIMATE   -- SE
                *;
                *;      OUTPUT: DIFFERENCE SIGNAL -- D (accumulator)
                *;
                *;      NOTATION:  SL -- 14b TC (Q0) [sign extended]
                *;                 SE -- 15b TC (Q0) [sign extended]
                *;                 D  -- 16b TC (Q0)
                *;
                *;************************************************************
                *;
                *;                  SL   +-------+
                *;             ------->|       |   D
                *;                  SE   | SUBTA |------->
                *;             ------->|       |
                *;                       +-------+
                *;
                *;************************************************************
                *;
02AA 2026       AQUAN   LAC     SAMPLE          ; compute difference sig
02AB 1003               SUB     SE
                *;
                *;************************************************************
                *;  ADAPTIVE QUANTIZER
                *;
                *; Implements the following modules (per CCITT spec):
                *;
                *;      LOG     -- computes log of difference signal
                *;      SUBTB   -- scales log by subtracting Y
                *;      QUAN    -- computes 4b output
                *;
                *;************************************************************
                *;
                *;      INPUT:  DIFFERENCED PCM SAMPLE -- D (accumulator)
                *;              QUANTIZER SCALE FACTOR -- Y (YOVER4)
                *;
                *;      OUTPUT: ADPCM OUTPUT SAMPLE    -- I
                *;
                *;      NOTATION:  D      -- 16b TC (Q0)
                *;                 YOVER4 -- 11b SM (Q7) POSITIVE VALUE ONLY
                *;                 I      -- 4b SM (Q0)
                *;
                *;************************************************************
                *;
                *;                          DS
                *;           +---------------------------+
                *;           |                           |
                *;           |                           V
                *;   D    +--+----+  DL   +-------+ DLN +--------+   I
                *; ------>|  LOG  |------>| SUBTB |----->|  QUAN  |------>
                *;        +-------+       +-------+      +--------+
                *;                            ^
```

```
                *;                                  | Y
                *;
                *;************************************************************
                *;
                *; first get log of difference signal -- express
                *; as unsigned 11b number (4b exp/7b mantissa)
                *;
                *; First order log approximation: log2 (1+x) = x.
                *;
02AC 5824               SACH    TEMP4     ; -1 if neg; 0 if positive (DS)
02AD 7F88               ABS
02AE 5021               SACL    TEMP1
02AF 184C       GETEXP  SUB     ONE,8     ; binary search to get exponent
02B0 FD00               BGEZ    C8TO14
02B1 02E7
02B2 046D       C0TO7   ADD     M15,4     ; TEMP1-16       exp = 0-7
02B3 FD00               BGEZ    C4TO7
02B4 02CE
02B5 027D       C0TO3   ADD     THREE,2   ; TEMP1-4        exp = 0-3
02B6 FD00               BGEZ    C2TO3
02B7 02C3
02B8 014C       C0TO1   ADD     ONE,1     ; TEMP1-2        exp = 0-1
02B9 FD00               BGEZ    EXP1
02BA 02BF
02BB 7000       EXP0    LARK    0,0       ; exp = 0
02BC 2721               LAC     TEMP1,7
02BD F900               B       GETMAN    ; save exponent and get mantissa
02BE 0321
02BF 7001       EXP1    LARK    0,1       ; exp = 1
02C0 2621               LAC     TEMP1,6
02C1 F900               B       GETMAN
02C2 0321
02C3 124C       C2TO3   SUB     ONE,2     ; TEMP1-8       exp = 2-3
02C4 FD00       .       BGEZ    EXP3
02C5 02CA
02C6 7002       EXP2    LARK    0,2       ; exp = 2
02C7 2521               LAC     TEMP1,5
02C8 F900               B       GETMAN
02C9 0321
02CA 7003       EXP3    LARK    0,3       ; exp = 3
02CB 2421               LAC     TEMP1,4
02CC F900               B       GETMAN
02CD 0321
02CE 147D       C4TO7   SUB     THREE,4   ; TEMP1-64       exp = 4-7
02CF FD00               BGEZ    C6TO7
02D0 02DC
02D1 054C       C4TO5   ADD     ONE,5     ; TEMP1-32       exp = 4-5
02D2 FD00               BGEZ    EXP5
02D3 02D8
02D4 7004       EXP4    LARK    0,4       ; exp = 4
02D5 2321               LAC     TEMP1,3
02D6 F900               B       GETMAN
02D7 0321
02D8 7005       EXP5    LARK    0,5       ; exp = 5
02D9 2221               LAC     TEMP1,2
02DA F900               B       GETMAN
02DB 0321
```

```
02DC 164C  C6TO7   SUB    ONE,6    ; TEMP1-128      exp = 6-7
02DD FD00          BGEZ   EXP7
02DE 02E3
02DF 7006  EXP6    LARK   0,6      ; exp = 6
02E0 2121          LAC    TEMP1,1
02E1 F900          B      GETMAN
02E2 0321
02E3 7007  EXP7    LARK   0,7      ; exp = 7
02E4 2021          LAC    TEMP1
02E5 F900          B      GETMAN
02E6 0321
02E7 186D  C8TO14  SUB    M15,8    ; TEMP1-4096     exp = 8-14
02E8 FD00          BGEZ   CCTOE
02E9 030B
02EA 0A7D  C8TO11  ADD    THREE,10 ;TEMP1-1024      exp = 8-11
02EB FD00          BGEZ   CATOB
02EC 02FC
02ED 094C  C8TO9   ADD    ONE,9    ; TEMP1-512      exp = 8-9
02EE FD00          BGEZ   EXP9
02EF 02F6
02F0 7008  EXP8    LARK   0,8      ; exp = 8
02F1 2F21          LAC    TEMP1,15
02F2 5821          SACH   TEMP1
02F3 2021          LAC    TEMP1
02F4 F900          B      GETMAN
02F5 0321
02F6 7009  EXP9    LARK   0,9      ; exp = 9
02F7 2E21          LAC    TEMP1,14
02F8 5821          SACH   TEMP1
02F9 2021          LAC    TEMP1
02FA F900          B      GETMAN
02FB 0321
02FC 1A4C  CATOB   SUB    ONE,10   ; TEMP1-2048     exp = 10-11
02FD FD00          BGEZ   EXP11
02FE 0305
02FF 700A  EXP10   LARK   0,10     ; exp = 10
0300 2D21          LAC    TEMP1,13
0301 5821          SACH   TEMP1
0302 2021          LAC    TEMP1
0303 F900          B      GETMAN
0304 0321
0305 700B  EXP11   LARK   0,11     ; exp = 11
0306 2C21          LAC    TEMP1,12
0307 5821          SACH   TEMP1
0308 2021          LAC    TEMP1
0309 F900          B      GETMAN
030A 0321
030B 1C7D  CCTOE   SUB    THREE,12 ; TEMP1-16384    exp = 12-14
030C FD00          BGEZ   EXP14
030D 031D
030E 0D4C  CCTOD   ADD    ONE,13   ; TEMP1-8192     exp = 13-14
030F FD00          BGEZ   EXP13
0310 0317
0311 700C  EXP12   LARK   0,12     ; exp = 12
0312 2B21          LAC    TEMP1,11
0313 5821          SACH   TEMP1
0314 2021          LAC    TEMP1
```

```
0315 F900          B      GETMAN
0316 0321
0317 700D  EXP13   LARK   0,13     ; exp = 13
0318 2A21          LAC    TEMP1,10
0319 5821          SACH   TEMP1
031A 2021          LAC    TEMP1
031B F900          B      GETMAN
031C 0321
031D 700E  EXP14   LARK   0,14     ; exp = 14
031E 2921          LAC    TEMP1,9
031F 5821          SACH   TEMP1
0320 2021          LAC    TEMP1
0321 796F  GETMAN  AND    M127
0322 3021          SAR    0,TEMP1
0323 0721          ADD    TEMP1,7 ; DL  4e...7m (sign=SGN(D))
                   *;
                   *;-----------------------------------------------
                   *;
                   *; scale LOG D by subtraction (Y>>2 is in YOVER4)
                   *;
                   *;-----------------------------------------------
                   *;
0324 0B4C  SUBTB   ADD    ONE,11   ; offset by 2K
0325 1029          SUB    YOVER4
                   *;
                   *;-----------------------------------------------
                   *; 16 LEVEL quantizer
                   *;
                   *; Table values defined in CCITT spec p67
                   *; Implemented table is offset by 2048
                   *;-----------------------------------------------
                   *;
     07F9  ITAB1   EQU    2041     ; bottom of level 1
     087B  ITAB2   EQU    2171     ; bottom of level 2
     08CA  ITAB3   EQU    2250     ; bottom of level 3
     0905  ITAB4   EQU    2309     ; bottom of level 4
     0936  ITAB5   EQU    2358     ; bottom of level 5
     0964  ITAB6   EQU    2404     ; bottom of level 6
     0995  ITAB7   EQU    2453     ; bottom of level 7
                   *;
0326 107C  QUAN    SUB    K2309    ; TEMP2-2309
0327 FD00          BGEZ   CI4TO7
0328 033E
0329 007B  CI0TO3  ADD    K138     ; TEMP2-2171    I = 0-3
032A FD00          BGEZ   CI2TO3
032B 0335
032C 007A  CI0TO1  ADD    K130     ; TEMP2-2041    I = 0-1
032D FD00          BGEZ   IEQ1
032E 0332
032F 7E00  IEQ0    LACK   0
0330 F900          B      GETIM
0331 0351
0332 7E01  IEQ1    LACK   1
0333 F900          B      GETIM
0334 0351
0335 1078  CI2TO3  SUB    K79      ; TEMP2-2250    I = 2-3
0336 FD00          BGEZ   IEQ3
```

```
0337 033B
0338 7E02  IEQ2    LACK    2
0339 F900          B       GETIM
033A 0351
033B 7E03  IEQ3    LACK    3
033C F900          B       GETIM
033D 0351
033E 1079  CI4TO7  SUB     K95     ; TEMP2-2404     I = 4-7
033F FD00          BGEZ    CI6TO7
0340 034A
0341 0064  CI5TO6  ADD     K46     ; TEMP2-2358     I = 5-6
0342 FD00          BGEZ    IEQ5
0343 0347
0344 7E04  IEQ4    LACK    4
0345 F900          B       GETIM
0346 0351
0347 7E05  IEQ5    LACK    5
0348 F900          B       GETIM
0349 0351
034A 1065  CI6TO7  SUB     K49     ; TEMP2-2453     I = 6-7
034B FD00          BGEZ    IEQ7
034C 0350
034D 7E06  IEQ6    LACK    6
034E F900          B       GETIM
034F 0351
0350 7E07  IEQ7    LACK    7
0351 5002  GETIM   SACL    IM      ; accumulator = |I|
0352 7824          XOR     TEMP4   ; add sign bit and flip if necessary
0353 796D          AND     M15     ; mask for final four-bit value
0354 7F8D  QDONE   RET             ; return from AQUAN
```

```
                 COPY    PRDICT.ASM
          *;
          *;*****************************************************************
          *; ADAPTATION/PREDICTION
          *;
          *;  Implements the following modules per CCITT spec:
          *;
          *;    Inverse Adaptive Quantizer
          *;     RECONST          -- reconstructs D from I
          *;     ADDA             -- adds back scale factor
          *;     ANTILOG          -- log to lin conversion to get DQ
          *;     FLOAT A          -- float DQ
          *;    Scale Factor Adaptation
          *;     FUNCTW           -- map I to log scale factor
          *;     FILTD            -- update fast scale factor
          *;     LIMB             -- limit scale factor
          *;     FILTE            -- update slow scale factor
          *;    Adaptation Speed Control
          *;     FUNCTF           -- map I to F function
          *;     FILTA            -- update short term ave of F
          *;     FILTB            -- update long term ave of F
          *;     SUBTC            -- determ speed control update
          *;                              technique
          *;     FILTC            -- update speed control
          *;    Adaptive Predictor
          *;     ADDB             -- compute reconstructed signal
          *;     FLOAT A          -- float SR
          *;     ADDC             -- compute sign of PK
          *;     UPA2             -- update A2 coeff of 2nd order pred
          *;     LIMC             -- limit A2
          *;     UPA1             -- update A1 coeff of 2nd order pred
          *;     LIMD             -- limit A1
          *;     UPB              -- update coeffs of 6th order pred
          *;     XOR              -- compute sign of DQ*DQn
          *;
          *;     NOTE: DELAY A/B/C implicit in timing of MIX/LIMA
          *;           and computation of SEZ/SE
          *;
          *;*****************************************************************
          *;
          *; First convert quantized difference back to log domain.
          *; This is done by table look-up.  Also use ADPCM magnitude
          *; to look-up the scale-factor multipliers WI and rate-of-
          *; change weighting function FI.
          *;
0355 2002 PRDICT LAC     IM
0356 006A        ADD     INQTAB  ; reconst table
0357 6721        TBLR    TEMP1   ; DQLN
0358 034C        ADD     ONE,3   ; WI table address and offset
0359 6769        TBLR    WI      ; lookup WI
035A 034C        ADD     ONE,3   ; FI table address and offset
035B 6768        TBLR    FI      ; lookup FI
          *;
          *;*****************************************************************
          *; INVERSE ADAPTIVE QUANTIZER
          *;
          *;     INPUT:  ADPCM INPUT SAMPLE      -- I (IM->TEMP1)
```

```
D0057                *;              QUANTIZER SCALE FACTOR -- Y (YOVER4)
D0058                *;
D0059                *;      OUTPUT: QUANTIZED DIFFERENCE SIGNAL -- DQ
D0060                *;
D0061                *;      NOTATION:  I        -- 4b SM (Q0)
D0062                *;                 IM       -- 3b magnitude (Q0)
D0063                *;                 DQLN(TEMP1) -- 12b TC (Q7) [sign extended]
D0064                *;                 YOVER4 -- 11b SM (Q7) POSITIVE VALUE ONLY
D0065                *;                 DQ       -- 15b TC (Q0) [sign extended]
D0066                *;                 DQMAN*8-- 9b magnitude
D0067                *;                 DQEXP    -- 4b magnitude
D0068                *;
D0069                *;*******************************************************
D0070                *;
D0071                *;                               DQS
D0072                *;           +------------------------------+
D0073                *;           |                              |
D0074                *;           |                              V
D0075                *;   I    +---+---+ DQLN  +--------+  DQL  +---------+  DQ
D0076                *; ------>|RECONST|------>|  ADDA  |------>|ANTILOG |------>
D0077                *;        +-------+       +--------+       +---------+
D0078                *;                            ^
D0079                *;                            | Y
D0080                *;
D0081                *;*******************************************************
D0082                *;
D0083                *; add back scale factor
D0084                *;
D0085 035C 2521  ADDA    LAC     TEMP1,5
D0086 035D 0529          ADD     YOVER4,5
D0087                *;
D0088                *; now covert to linear domain
D0089                *;
D0090 035E 0C4C  ALOG    ADD     ONE,12           ; inc exponent for floated value
D0091 035F 5C15          SACH    DQEXP,4          ; save exponent + sign ext
D0092 0360 7972          AND     M4095            ; isolate mantissa * 2**5
D0093 0361 0C4C          ADD     ONE,12           ; Alog x = 1 + x
D0094 0362 505B          SACL    DQMAN            ; DQMAN = 0001 XXXX XXX0 0000
D0095 0363 2015          LAC     DQEXP
D0096 0364 0066          ADD     SHIFT            ; add table ptr
D0097 0365 6723          TBLR    TEMP3            ; get multiplier
D0098 0366 6A23          LT      TEMP3
D0099 0367 044C          ADD     ONE,4            ; offset to mask table
D0100 0368 6723          TBLR    TEMP3            ; mask for dqman
D0101 0369 6D5B          MPY     DQMAN            ; adjust mantissa
D0102 036A 7F8E          PAC
D0103 036B 5C10          SACH    DQ,4
D0104 036C 2B4C  ADDSGN  LAC     ONE,11           ; +2048 represents +sign
D0105 036D 5054          SACL    SDQ
D0106 036E 2001          LAC     I                ; check sign
D0107 036F 134C          SUB     ONE,3
D0108 0370 FA00          BLZ     FLTDQ
      0371 0377
D0109 0372 7F89          ZAC                      ; I carried negative sign
D0110 0373 1010          SUB     DQ
D0111 0374 5010          SACL    DQ
D0112 0375 2B4B          LAC     MINUS,11         ; -2048 represents -sign
```

```
D0113 0376 5054          SACL    SDQ
D0114                *;
D0115                *;*******************************************************
D0116                *; FLOAT DQ -- convert 2's comp number to floating
D0117                *;
D0118                *;      INPUT:   DQ
D0119                *;
D0120                *;      OUTPUT: 4b exponent in DQEXP (saved from log value)
D0121                *;              6b mantissa*8 in DQMAN (adjusted from log)
D0122                *;              sign preserved in DQ
D0123                *;
D0124                *;*******************************************************
D0125                *;
D0126 0377 255B  FLTDQ   LAC     DQMAN,5 ; 00000000 0000001X XXXXXX00 00000000
D0127 0378 5C5B          SACH    DQMAN,4 ; DQMAN = 0000 0000 001X XXXX
D0128 0379 235B          LAC     DQMAN,3 ; DQMAN * 2**3
D0129 037A 7923          AND     TEMP3
D0130 037B 505B          SACL    DQMAN
D0131                *;
D0132                *;*******************************************************
D0133                *; QUANTIZER SCALE FACTOR ADAPTATION
D0134                *;
D0135                *;      INPUT:   ADPCM SAMPLE   -- I
D0136                *;
D0137                *;      OUTPUT: FAST QUANTIZER SCALE FACTOR -- YU
D0138                *;              SLOW QUANTIZER SCALE FACTOR -- YL (YLL/YLH)
D0139                *;
D0140                *;      NOTATION: I   -- 4b SM (Q0)
D0141                *;                YU -- 13b unsigned (Q9)
D0142                *;                YL -- 19b unsigned (Q15)
D0143                *;                   stored as:
D0144                *;                     low 15b -- YLL
D0145                *;                     hi   4b -- YLH
D0146                *;
D0147                *;*******************************************************
D0148                *;
D0149                *; WI  +-------+YUT +------+YUP +--------+       YU
D0150                *; --->| FILTD |--->| LIMB |-+->| DELAYB |---------------->
D0151                *;     +-------+    +------+ | +--------+
D0152                *;        ^                  |
D0153                *;        | Y                |  +-------+YLP +--------+ YL
D0154                *;        |                  +->| FILTE |--->| DELAYC |-+->
D0155                *;                              +-------+    +--------+ |
D0156                *;                                  ^                  |
D0157                *;                                  +------------------+
D0158                *;
D0159                *;*******************************************************
D0160                *; Update fast adaptation scale factor
D0161                *;
D0162                *;   YU(k) = (1-2**-5)*Y(k) + (2**-5)*W(I(k))
D0163                *;
D0164                *;      INPUT:  QUANTIZER SCALE FACTOR  -- Y
D0165                *;              SCALE FACTOR MULTIPLIER -- WI
D0166                *;
D0167                *;      OUTPUT: FAST QUANTIZER SCALE FACTOR -- YU
D0168                *;
D0169                *;      NOTATION:  WI -- 12b TC (Q4) [sign extended]
```

```
                *;               Y  -- 13b unsigned (Q9)
                *;               YU -- 13b unsigned (Q9)
                *;
                *;*****************************************************
                *;
037C 2C09  FILTD  LAC     Y,12            ; Y      (Q21)
037D 1709         SUB     Y,7             ; Y/32   (Q21)
037E 0C69         ADD     WI,12           ; WI/32  (Q21)
037F 5C4E         SACH    YU,4            ; YU     (Q9)
                *;
                *; limit quant scale factor 1.06 <= YU <= 10.0
                *;
0380 1C6B  LIMB   SUB     K544,12         ; check lo threshold
0381 FD00         BGEZ    CHKHI
0382 0386
0383 206B         LAC     K544
0384 F900         B       STRLIM          ; go store limited value
0385 038A
0386 1C61  CHKHI  SUB     K4576,12        ; check hi threshold
0387 FB00         BLEZ    FILTE           ; within limits--continue
0388 038B
0389 206C         LAC     K5120
038A 504E  STRLIM SACL    YU
                *;
                *;*****************************************************
                *; Update slow adaptation scale factor
                *;
                *;    YL(k) = (1-2**-6)*YL(k-1) + 2**-6 * YU(k)
                *;
                *;       INPUT:  SLOW QUANTIZER SCALE FACTOR -- YL (YLL/YLH)
                *;               FAST QUANTIZER SCALE FACTOR -- YU
                *;
                *;       OUTPUT: SLOW QUANTIZER SCALE FACTOR -- YL (YLL/YLH)
                *;
                *;       NOTATION:  YU -- 13b unsigned (Q9)
                *;                  YL -- 19b unsigned (Q15)
                *;                     stored as:
                *;                        low 15b -- YLL
                *;                        hi   4b -- YLH
                *;
                *;*****************************************************
                *;
038B 2649  FILTE  LAC     YLH,6        ; shift yl left by 6
038C 5021         SACL    TEMP1
038D 2F21         LAC     TEMP1,15     ; YL       (Q21)
038E 064A         ADD     YLL,6
038F 1F49         SUB     YLH,15       ; YL/64    (Q21)
0390 104A         SUB     YLL
0391 064E         ADD     YU,6         ; YU/64    (Q21)
0392 5921         SACH    TEMP1,1
0393 7974         AND     M32767
0394 5022         SACL    TEMP2        ; result = yl (shifted left by 6)
0395 2A22         LAC     TEMP2,10     ; shift result right 6 --> 4.Q15
0396 5822         SACH    TEMP2
0397 2921         LAC     TEMP1,9      ; YL       (Q15)
0398 0022         ADD     TEMP2
0399 5949         SACH    YLH,1
```

```
039A 7974         AND     M32767
039B 504A         SACL    YLL
                *;
                *;*****************************************************
                *; ADAPTATION SPEED CONTROL
                *;
                *;     INPUT:  ADPCM SAMPLE  -- I
                *;
                *;     OUTPUT: UNLIMITED SPEED CONTROL -- AP (APP)
                *;
                *;     NOTATION: I    -- 4b SM (Q0)
                *;               APP -- 10b unsigned (Q8)
                *;
                *;*****************************************************
                *;
                *;                      ¦ Y
                *;                      V
                *; FI   +-------+DMSP+-------+ AX +-------+APP +--------+ AP
                *; -+->¦ FILTA ¦-+->¦ SUBTC ¦--->¦ FILTC ¦--->¦ DELAYA ¦-+->
                *;  ¦  +-------+ ¦  +-------+    +-------+    +--------+ ¦
                *;  ¦     ^      V      ^           ^                    ¦
                *;  ¦  DMS¦ +--------+  ¦           +--------------------+
                *;  ¦     +-¦ DELAYA ¦  ¦
                *;  ¦       +--------+  ¦
                *;  ¦                   ¦
                *;  ¦  +-------+  DMLP  ¦
                *;  +->¦ FILTB ¦--+-----+
                *;     +-------+  ¦
                *;        ^       V
                *;     DML¦ +--------+
                *;        +-¦ DELAYA ¦
                *;          +--------+
                *;
                *;*****************************************************
                *; update short term average of FI
                *;
                *;   DMS(k) = (1-2**-5)*DMS(k-1) + 2**-5 * FI(k)
                *;
                *;     INPUT:  SHORT TERM AVERAGE       -- DMS
                *;             RATE-OF-CHANGE FUNCTION -- FI
                *;
                *;     OUTPUT: SHORT TERM AVERAGE -- DMS
                *;
                *;     NOTATION:  DMS -- 12b unsigned (Q9)
                *;                FI  -- 7b unsigned (Q4)
                *;
                *;*****************************************************
                *;
039C 2F68  FILTA  LAC     FI,15           ; FI/32  (Q24)
039D 0F07         ADD     DMS,15          ; DMS    (Q24)
039E 1A07         SUB     DMS,10          ; DMS/32 (Q24)
039F 5907         SACH    DMS,1
                *;
                *;*****************************************************
                *; update long term average of FI
                *;
                *;   DML(k) = (1-2**-7)*DML(k-1) + 2**-7 * FI(k)
                *;
                *;      INPUT:  LONG TERM AVERAGE         -- DML
                *;              RATE-OF-CHANGE FUNCTION -- FI
                *;
                *;      OUTPUT: LONG TERM AVERAGE -- DML
                *;
                *;      NOTATION:  DML -- 14b unsigned (Q11)
                *;                 FI  -- 7b unsigned (Q4)
                *;
                *;****************************************************
                *;
03A0 2F68  FILTB  LAC     FI,15            ; FI/128    (Q26)
03A1 0F08         ADD     DML,15           ; DML       (Q26)
03A2 1808         SUB     DML,8            ; DML/128   (Q26)
03A3 5908         SACH    DML,1
                *;
                *;----------------------------------------------------
                *; Compute mag of diff of short and long term functions of
                *; quantizer output sequence and perform threshold
                *; comparison to compute speed control parameter--low-pass
                *; result.
                *;
                *;   APP(k) = (1-2**-4)*APP(k-1) + 2**-3 , if Y < 3 or
                *;                               if |DMS-DML| > 2**-3 * DML
                *;   else
                *;
                *;   APP(k) = (1-2**-4)*APP(k-1)
                *;
                *;     INPUT:  SHORT TERM AVERAGE        -- DMS
                *;             LONG TERM AVERAGE         -- DML
                *;             UNLIMITED SPEED CONTROL -- APP
                *;             QUANTIZER SCALE FACTOR   -- Y
                *;
                *;     OUTPUT: UNLIMITED SPEED CONTROL -- APP
                *;
                *;     NOTATION:  APP -- 10b unsigned (Q8)
                *;                Y   -- 13b unsigned (Q9)
                *;                DMS -- 12b unsigned (Q9)
                *;                DML -- 14b unsigned (Q11)
                *;
                *;****************************************************
                *;
03A4 6505  FILTC  ZALH    APP              ; APP              (Q24)
03A5 1C05         SUB     APP,12           ; APP/16           (Q24)
03A6 5805         SACH    ARP              ; (1-2**-4)*APP    (Q8)
03A7 2009         LAC     Y
03A8 197D         SUB     THREE,9          ; 3         (Q9)
03A9 FA00         BLZ     ADD18
03AA 03B3
03AB 2D08         LAC     DML,13           ; DML/8     (Q27)
03AC 5823         SACH    TEMP3            ; DML/8     (Q11)
03AD 2207         LAC     DMS,2            ; DMS       (Q11)
03AE 1008         SUB     DML              ; DMS-DML
03AF 7F88         ABS
03B0 1023         SUB     TEMP3            ; |DMS-DML|-DML/8
03B1 FA00         BLZ     APRED
03B2 03B6
```

```
03B3 2005  ADD18  LAC     APP              ; APP      (Q8)
03B4 054C         ADD     ONE,5
03B5 5005         SACL    APP              ; + 1/8    (Q8)
                *;
                *;****************************************************
                *; ADAPTIVE PREDICTOR
                *;
                *;****************************************************
                *;
     03B6  APRED  EQU     $
                *;
                *;****************************************************
                *; compute coeff of 6th order predictor
                *;
                *; Bi(k) = (1-2**-8)*Bi(k-1) + 2**-7*SGN[DQ(k)]*SGN[DQ(k-i)]
                *;     for i = 1...6
                *;     and Bi is implicitly limited to +/- 2
                *;
                *;     NOTATION:        Bn   -- 16b TC (Q14)
                *;                      SDQn -- +2048 if sign positive
                *;                              -2048 if sign negative
                *;
                *;****************************************************
                *;
03B6 6A5A  GETB6  LT      SDQ6
03B7 280F         LAC     B6,8     ; B6 * 2**-8 TRUNCATED
03B8 5821         SACH    TEMP1
03B9 2F0F         LAC     B6,15    ; Q29
03BA 1F21         SUB     TEMP1,15
03BB 6D54         MPY     SDQ      ; SGN(SDQ)*SGN(SDQ6) * 2**-7  (Q29)
03BC 6B59         LTD     SDQ5
03BD 590F         SACH    B6,1     ; Q14
03BE 280E  GETB5  LAC     B5,8     ; B5 * 2**-8 TRUNCATED
03BF 5821         SACH    TEMP1
03C0 2F0E         LAC     B5,15    ; Q29
03C1 1F21         SUB     TEMP1,15
03C2 6D54         MPY     SDQ
03C3 6B58         LTD     SDQ4
03C4 590E         SACH    B5,1     ; Q14
03C5 280D  GETB4  LAC     B4,8     ; B4 * 2**-8 TRUNCATED
03C6 5821         SACH    TEMP1
03C7 2F0D         LAC     B4,15    ; Q29
03C8 1F21         SUB     TEMP1,15
03C9 6D54         MPY     SDQ
03CA 6B57         LTD     SDQ3
03CB 590D         SACH    B4,1     ; Q14
03CC 280C  GETB3  LAC     B3,8     ; B3 * 2**-8 TRUNCATED
03CD 5821         SACH    TEMP1
03CE 2F0C         LAC     B3,15    ; Q29
03CF 1F21         SUB     TEMP1,15
03D0 6D54         MPY     SDQ
03D1 6B56         LTD     SDQ2
03D2 590C         SACH    B3,1     ; Q14
03D3 280B  GETB2  LAC     B2,8     ; B2 * 2**-8 TRUNCATED
03D4 5821         SACH    TEMP1
03D5 2F0B         LAC     B2,15    ; Q29
03D6 1F21         SUB     TEMP1,15
```

```
03D7 6D54          MPY     SDQ
03D8 6B55          LTD     SDQ1
03D9 590B          SACH    B2,1     ; Q14
03DA 280A  GETB1   LAC     B1,8     ; B1 * 2**-8 TRUNCATED
03DB 5821          SACH    TEMP1
03DC 2F0A          LAC     B1,15    ; Q29
03DD 1F21          SUB     TEMP1,15
03DE 6D54          MPY     SDQ
03DF 6B54          LTD     SDQ
03E0 590A          SACH    B1,1     ; Q14
                   *;
                   *;*************************************************************
                   *; To update coefficients of 2nd order predictor,
                   *; First get sign of sum of SEZ and DQ
                   *;
                   *;      NOTATION: if SEZ+DQ >= 0 then PK0 = 512
                   *;                  else              PK0 = -512
                   *;
                   *;*************************************************************
                   *;
03E1 6950  ADDC    DMOV    PK1                  ; PK1==>PK2
03E2 694F          DMOV    PK0                  ; PK0==>PK1
03E3 2004          LAC     SEZ
03E4 0110          ADD     DQ,1
03E5 5821          SACH    TEMP1        ; FFFF or 0000
03E6 2A21          LAC     TEMP1,10     ; FC00 or 0000
03E7 094C          ADD     ONE,9        ; FE00 or 0200 ¦ -512 or +512
03E8 504F          SACL    PK0
03E9 6A4F  SUMGTO LT       PK0
                   *;
                   *;*************************************************************
                   *;  now calculate 1/2 * f[A1(k-1)]
                   *;
                   *;        = 2*A1        if ¦A1¦  <= 1/2
                   *;        = SGN(A1)     if ¦A1¦  >  1/2
                   *;
                   *;*************************************************************
                   *;
03EA 2111  GETF    LAC     A1,1                 ; 2*A1
03EB 5023          SACL    TEMP3
03EC FA00          BLZ     GETF2
03ED 03F4
03EE 1E4C  GETF1   SUB     ONE,14               ; is ¦A1¦ < 1/2
03EF FA00          BLZ     GETA1
03F0 03FA
03F1 2062          LAC     K16382               ; approx 1
03F2 F900          B       DONEF
03F3 03F9
03F4 7F88  GETF2   ABS
03F5 1E4C          SUB     ONE,14               ; is ¦A1¦ < 1/2
03F6 FA00          BLZ     GETA1
03F7 03FA
03F8 2063          LAC     M16382               ; approx -1
03F9 5023  DONEF   SACL    TEMP3
                   *;
                   *;*************************************************************
                   *; Compute A1 coeff of 2nd order predictor
```

```
                   *;
                   *;   A1(k) = (1-2**-8)*A1(k-1)
                   *;                          + (3*2**-8)*SGN[p(k)]*SGN[p(k-1)]
                   *;
                   *;     NOTATION:  A1  -- 16b TC (Q14)
                   *;                PKn -- +512 if SGN[p(k)] = 1
                   *;                       -512 if SGN[p(k)] = -1
                   *;
                   *;*************************************************************
                   *;
03FA 2811  GETA1   LAC     A1,8                 ; A1*2**-8 TRUNCATED
03FB 5822          SACH    TEMP2
03FC 2C11          LAC     A1,12                ; Q26
03FD 1C22          SUB     TEMP2,12
03FE 6D50          MPY     PK1                  ; SGN[p(k-1)*SGN[p(k)]
03FF 7F8F          APAC
0400 7F8F          APAC
0401 7F8F          APAC                         ; +3*SGN[p(k-1)*SGN[p(k)]
0402 5C11          SACH    A1,4                 ; store as Q14
0403 7F8E          PAC                          ; save sign
                   *;
                   *;*************************************************************
                   *; Compute A2 coeff of 2nd order predictor
                   *;
                   *;   A2(k) = (1-2**-7)*A2(k-1)
                   *;         + (2**-7)*{SGN[p(k)]*SGN[(p(k-2)]
                   *;                   - f[A1(k-1)]*SGN[p(k)]*SGN[p(k-1)]}
                   *;
                   *;     NOTATION:  A2        -- 16b TC (Q14)
                   *;                F(),TEMP3 -- 16b TC (Q14)
                   *;                PKn.      -- +512 if SGN[p(k)] = 1
                   *;                             -512 if SGN[p(k)] = -1
                   *;
                   *;*************************************************************
                   *;
0404 FD00  GETA2   BGEZ    SUBF                 ; if sign + --> subtract F
0405 0409
0406 7F89          ZAC                          ;  else negate F and subtract
0407 1023          SUB     TEMP3
0408 5023          SACL    TEMP3
                   *;
0409 2912  SUBF    LAC     A2,9                 ; A2*2**-7 TRUNCATED
040A 581E          SACH    SUM4
040B 6D51          MPY     PK2                  ; SGN[p(k-2)]*SGN[p(k)]
040C 7F8E          PAC
040D 7F8F          APAC                         ; 2*2**-8*above  (Q26)
040E 1623          SUB     TEMP3,6              ; 2*TEMP3*2**-7  (Q26)
040F 5C23          SACH    TEMP3,4              ; Q14
0410 2012          LAC     A2
0411 101E          SUB     SUM4                 ; leak factor
0412 0023          ADD     TEMP3
0413 5012          SACL    A2                   ; Q14
                   *
0414 5821          SACH    TEMP1                ; save sign to make +/- .75
                   *;
                   *; limit A2 to +/- .75 and prevent overflow
                   *;
```

```
0415 7F88  LIMC   ABS
0416 1C7D         SUB    THREE,12         ; |value| must be < .75
0417 FB00         BLEZ   LIMD
0418 041D
0419 2C7D         LAC    THREE,12         ; .75  (Q14)
041A 7821         XOR    TEMP1            ; 1's complement if negative
041B 1021         SUB    TEMP1            ; 2's complement if negative
041C 5012  DONEC  SACL   A2               ; Q14
           *;
           *; limit A1(k) to +/- [1-2**-4 - A2(k)]
           *;
041D 2A6D  LIMD   LAC    M15,10           ; 1-2**-4      (Q14)
041E 1012         SUB    A2
041F 5021         SACL   TEMP1            ; 1-2**-4-A2P  (Q14)
0420 2011         LAC    A1
0421 5824         SACH   TEMP4            ; save sign to make +/- LIMIT
0422 7F88         ABS
0423 1021         SUB    TEMP1
0424 FB00         BLEZ   FLTSR            ; A1 <= LIMIT
0425 042A
0426 2021  A1LIM  LAC    TEMP1            ; ABS value of LIMIT
0427 7824         XOR    TEMP4            ; 1's complement if negative
0428 1024         SUB    TEMP4            ; 2's complement if negative
0429 5011         SACL   A1               ; Q14
           *;
           *;**********************************************************
           *; COMPUTE RECONSTRUCTED SIGNAL
           *;
           *;     INPUT:  QUANTIZED DIFFERENCE SIGNAL -- DQ
           *;             SIGNAL ESTIMATE             -- SE
           *;
           *;     OUTPUT: RECONSTRUCTED SIGNAL -- SR
           *;
           *;     NOTATION:  DQ -- 15b TC (Q0) [sign extended]
           *;                SE -- 15b TC (Q0) [sign extended]
           *;                SR -- 16b TC (Q0)
           *;
           *;**********************************************************
           *; FLOAT SR -- convert 2's comp number to floating
           *;
           *;     INPUT:  accumulator
           *;
           *;     OUTPUT: --4b exponent left in SREXP
           *;             --6b mantissa*8 left in SRMAN
           *;             --sign preserved in SR
           *;
           *;**********************************************************
           *;
042A 2010  FLTSR  LAC    DQ       ; compute reconstructed signal
042B 0003         ADD    SE
042C 5013         SACL   SR
042D 7F88         ABS             ; convert to floating point notation
042E 5052         SACL   SRMAN
042F 174C         SUB    ONE,7    ; binary search to get exponent
0430 FD00         BGEZ   D8TOF
0431 0469
0432 036D  D0TO7  ADD    M15,3    ; TEMP1-8 -- exp = 0-7
```

```
0433 FD00         BGEZ   D4TO7
0434 044A
0435 017D  D0TO3  ADD    THREE,1 ; TEMP1-2 -- exp = 0-3
0436 FD00         BGEZ   D2TO3
0437 043D
0438 2052  D0TO1  LAC    SRMAN   ; exp = 0-1
0439 501C  EXX01  SACL   SREXP
043A 284C         LAC    ONE,8
043B 5052         SACL   SRMAN
043C 7F8D         RET
043D 114C  D2TO3  SUB    ONE,1   ; TEMP1-4 -- exp = 2-3
043E FD00         BGEZ   EXX3
043F 0445
0440 2752  EXX2   LAC    SRMAN,7 ; exp=2
0441 5052         SACL   SRMAN
0442 7E02         LACK   2
0443 501C         SACL   SREXP
0444 7F8D         RET
0445 2652  EXX3   LAC    SRMAN,6 ; exp=3
0446 5052         SACL   SRMAN
0447 7E03         LACK   3
0448 501C         SACL   SREXP
0449 7F8D         RET
044A 137D  D4TO7  SUB    THREE,3 ; TEMP1-32 -- exp = 4-7
044B FD00         BGEZ   D6TO7
044C 045A
044D 044C  D4TO5  ADD    ONE,4   ; TEMP1-16 -- exp = 4-5
044E FD00         BGEZ   EXX5
044F 0455
0450 2552  EXX4   LAC    SRMAN,5 ; exp=4
0451 5052         SACL   SRMAN
0452 7E04         LACK   4
0453 501C         SACL   SREXP
0454 7F8D         RET
0455 2452  EXX5   LAC    SRMAN,4 ; exp=5
0456 5052         SACL   SRMAN
0457 7E05         LACK   5
0458 501C         SACL   SREXP
0459 7F8D         RET
045A 154C  D6TO7  SUB    ONE,5   ; TEMP1-64 -- exp = 6-7
045B FD00         BGEZ   EXX7
045C 0462
045D 2352  EXX6   LAC    SRMAN,3 ; exp=6
045E 5052         SACL   SRMAN
045F 7E06         LACK   6
0460 501C         SACL   SREXP
0461 7F8D         RET
0462 2F52  EXX7   LAC    SRMAN,15
0463 5852         SACH   SRMAN
0464 2352         LAC    SRMAN,3
0465 5052         SACL   SRMAN
0466 7E07         LACK   7
0467 501C         SACL   SREXP
0468 7F8D         RET
0469 1177  D8TOF  SUB    K960,1  ; TEMP1-2048 -- exp = 8-15
046A FD00         BGEZ   DCTOF
046B 0491
```

```
D0606 046C 097D  D8TOB  ADD    THREE,9 ; TEMP1-512 -- exp = 8-11
D0607 046D FD00         BGEZ   DATOB
      046E 0480
D0608 046F 084C  D8TO9  ADD    ONE,8   ; TEMP1-256 -- exp = 8-9
D0609 0470 FD00         BGEZ   EXX9
      0471 0479
D0610 0472 2E52  EXX8   LAC    SRMAN,14 ; exp=8
D0611 0473 5852         SACH   SRMAN
D0612 0474 2352         LAC    SRMAN,3
D0613 0475 5052         SACL   SRMAN
D0614 0476 7E08         LACK   8
D0615 0477 501C         SACL   SREXP
D0616 0478 7F8D         RET
D0617 0479 2D52  EXX9   LAC    SRMAN,13 ; exp=9
D0618 047A 5852         SACH   SRMAN
D0619 047B 2352         LAC    SRMAN,3
D0620 047C 5052         SACL   SRMAN
D0621 047D 7E09         LACK   9
D0622 047E 501C         SACL   SREXP
D0623 047F 7F8D         RET
D0624 0480 194C  DATOB  SUB    ONE,9    ; TEMP1-1024 -- exp=10-11
D0625 0481 FD00         BGEZ   EXX11
      0482 048A
D0626 0483 2C52  EXX10  LAC    SRMAN,12 ; exp=10
D0627 0484 5852         SACH   SRMAN
D0628 0485 2352         LAC    SRMAN,3
D0629 0486 5052         SACL   SRMAN
D0630 0487 7E0A         LACK   10
D0631 0488 501C         SACL   SREXP
D0632 0489 7F8D         RET
D0633 048A 2B52  EXX11  LAC    SRMAN,11 ; exp=11
D0634 048B 5852         SACH   SRMAN
D0635 048C 2352         LAC    SRMAN,3
D0636 048D 5052         SACL   SRMAN
D0637 048E 7E0B         LACK   11
D0638 048F 501C         SACL   SREXP
D0639 0490 7F8D         RET
D0640 0491 1B7D  DCTOF  SUB    THREE,11 ; TEMP1-8192 -- exp=12-15
D0641 0492 FD00         BGEZ   DETOF
      0493 04A5
D0642 0494 0C4C  DCTOD  ADD    ONE,12   ; TEMP1-4096 -- exp=12-13
D0643 0495 FD00         BGEZ   EXX13
      0496 049E
D0644 0497 2A52  EXX12  LAC    SRMAN,10 ; exp=12
D0645 0498 5852         SACH   SRMAN
D0646 0499 2352         LAC    SRMAN,3
D0647 049A 5052         SACL   SRMAN
D0648 049B 7E0C         LACK   12
D0649 049C 501C         SACL   SREXP
D0650 049D 7F8D         RET
D0651 049E 2952  EXX13  LAC    SRMAN,9  ; exp=13
D0652 049F 5852         SACH   SRMAN
D0653 04A0 2352         LAC    SRMAN,3
D0654 04A1 5052         SACL   SRMAN
D0655 04A2 7E0D         LACK   13
D0656 04A3 501C         SACL   SREXP
D0657 04A4 7F8D         RET
```

```
D0658 04A5 1D4C  DETOF  SUB    ONE,13   ; TEMP1-16384 -- exp=14-15
D0659 04A6 FD00         BGEZ   EXX15
      04A7 04AF
D0660 04A8 2852  EXX14  LAC    SRMAN,8  ; exp=14
D0661 04A9 5852         SACH   SRMAN
D0662 04AA 2352         LAC    SRMAN,3
D0663 04AB 5052         SACL   SRMAN
D0664 04AC 7E0E         LACK   14
D0665 04AD 501C         SACL   SREXP
D0666 04AE 7F8D         RET
D0667 04AF 2752  EXX15  LAC    SRMAN,7  ; exp=15
D0668 04B0 5852         SACH   SRMAN
D0669 04B1 2352         LAC    SRMAN,3
D0670 04B2 5052         SACL   SRMAN
D0671 04B3 7E0F         LACK   15
D0672 04B4 501C         SACL   SREXP
D0673 04B5 7F8D         RET
```

```
                        COPY    UTILITY.ASM
                  *;
                  *;  code to do left shifts for SEZ/SE calculations
                  *;
04B6 7F88  RS1    ABS                     ; make positive before mask
04B7 7974         AND     M32767  ; keep lower 15 bits
04B8 5022         SACL    SUM1    ; save result
04B9 F900         B       CHK1    ; return
04BA 01C4
04BB 7F88  RS2    ABS
04BC 7974         AND     M32767
04BD 5023         SACL    SUM2
04BE F900         B       CHK2
04BF 01DE
04C0 7F88  RS3    ABS
04C1 7974         AND     M32767
04C2 5025         SACL    SUM3
04C3 F900         B       CHK3
04C4 01F8
04C5 7F88  RS4    ABS
04C6 7974         AND     M32767
04C7 501E         SACL    SUM4
04C8 F900         B       CHK4
04C9 0212
04CA 7F88  RS5    ABS
04CB 7974         AND     M32767
04CC 501F         SACL    SUM5
04CD F900         B       CHK5
04CE 022C
04CF 7F88  RS6    ABS
04D0 7974         AND     M32767
04D1 5020         SACL    SUM6
04D2 F900         B       CHK6
04D3 0246
04D4 7F88  RS11   ABS
04D5 7974         AND     M32767
04D6 5027         SACL    SUM7
04D7 F900         B       CHK11
04D8 025F
04D9 7F88  RS21   ABS
04DA 7974         AND     M32767
04DB 5028         SACL    SUM8
04DC F900         B       CHK21
04DD 0279
```

```
                  *;
                  *;-----------------------------------------------------------
                  *; FLOAT SUBROUTINE--convert 2's comp number to floating
                  *;                 INPUT: accumulator
                  *;                 OUTPUT:
                  *;                         --4b exponent left in accum
                  *;                         --6b mantissa left in TEMP1
                  *;                         --sign preserved in original number
                  *;
                  *;-----------------------------------------------------------
      002A FLTSFT EQU     42         ; address of shift multipliers
                  *;
                  *;
04DE 5821  FLOAT  SACH    TEMP1
04DF 2021         LAC     TEMP1
04E0 7F88         ABS
04E1 5021         SACL    TEMP1
04E2 164C         SUB     ONE,6      ; binary search to get expONEnt
04E3 FD00         BGEZ    E7TOD
04E4 0511
04E5 0076  E0TO7  ADD     K56        ; TEMP1-8 -- exp = 0-6
04E6 FD00         BGEZ    E4TO6
04E7 0501
04E8 017D  E0TO3  ADD     THREE,1 ; TEMP1-2 -- exp = 0-3
04E9 FD00         BGEZ    E2TO3
04EA 04F6
04EB 2021  E0TO1  LAC     TEMP1      ; exp = 0-1
04EC FE00         BNZ     E1
04ED 04F2
04EE 254C  E0     LAC     ONE,5      ; exp=0
04EF 5021         SACL    TEMP1
04F0 7E2A         LACK    FLTSFT+0
04F1 7F8D         RET
04F2 2521  E1     LAC     TEMP1,5 ; exp=1
04F3 5021         SACL    TEMP1
04F4 7E2B         LACK    FLTSFT+1
04F5 7F8D         RET
04F6 114C  E2TO3  SUB     ONE,1      ; TEMP1-4 -- exp = 2-3
04F7 FD00         BGEZ    E3
04F8 04FD
04F9 2421  E2     LAC     TEMP1,4 ; exp=2
04FA 5021         SACL    TEMP1
04FB 7E2C         LACK    FLTSFT+2
04FC 7F8D         RET
04FD 2321  E3     LAC     TEMP1,3 ; exp=3
04FE 5021         SACL    TEMP1
04FF 7E2D         LACK    FLTSFT+3
0500 7F8D         RET
0501 134C  E4TO6  SUB     ONE,3      ; TEMP1-16 -- exp = 4-6
0502 FD00         BGEZ    E5TO6
0503 0508
0504 2221  E4     LAC     TEMP1,2 ; exp=4
0505 5021         SACL    TEMP1
0506 7E2E         LACK    FLTSFT+4
0507 7F8D         RET
0508 144C  E5TO6  SUB     ONE,4      ; TEMP1-32 -- exp = 5-6
0509 FD00         BGEZ    E6
```

```
      050A 050F
E0088 050B 2121  E5     LAC     TEMP1,1 ; exp=5
E0089 050C 5021         SACL    TEMP1
E0090 050D 7E2F         LACK    FLTSFT+5
E0091 050E 7F8D         RET
E0092 050F 7E30  E6     LACK    FLTSFT+6         ; exp=6
E0093 0510 7F8D         RET
E0094 0511 1077  E7TOD  SUB     K960    ; TEMP1-1024 -- exp = 7-13
E0095 0512 FD00         BGEZ    EBTOD
      0513 052D
E0096 0514 087D  E7TOA  ADD     THREE,8 ; TEMP1-256  -- exp = 7-10
E0097 0515 FD00         BGEZ    E9TOA
      0516 0522
E0098 0517 074C  E7TO8  ADD     ONE,7   ; TEMP1-128  -- exp = 7-8
E0099 0518 FD00         BGEZ    E8
      0519 051E
E0100 051A 2F21  E7     LAC     TEMP1,15 ; exp=7
E0101 051B 5821         SACH    TEMP1
E0102 051C 7E31         LACK    FLTSFT+7
E0103 051D 7F8D         RET
E0104 051E 2E21  E8     LAC     TEMP1,14 ; exp=8
E0105 051F 5821         SACH    TEMP1
E0106 0520 7E32         LACK    FLTSFT+8
E0107 0521 7F8D         RET
E0108 0522 184C  E9TOA  SUB     ONE,8   ; TEMP1-512  -- exp = 9-10
E0109 0523 FD00         BGEZ    E10
      0524 0529
E0110 0525 2D21  E9     LAC     TEMP1,13 ; exp=9
E0111 0526 5821         SACH    TEMP1
E0112 0527 7E33         LACK    FLTSFT+9
E0113 0528 7F8D         RET
E0114 0529 2C21  E10    LAC     TEMP1,12 ; exp=10
E0115 052A 5821         SACH    TEMP1
E0116 052B 7E34         LACK    FLTSFT+10
E0117 052C 7F8D         RET
E0118 052D 1A7D  EBTOD  SUB     THREE,10         ; TEMP1-4096 -- exp=11-13
E0119 052E FD00         BGEZ    EDTOE
      052F 053B
E0120 0530 0B4C  EBTOC  ADD     ONE,11  ; TEMP1-2048 -- exp=11-12
E0121 0531 FD00         BGEZ    E12
      0532 0537
E0122 0533 2B21  E11    LAC     TEMP1,11 ; exp=11
E0123 0534 5821         SACH    TEMP1
E0124 0535 7E35         LACK    FLTSFT+11
E0125 0536 7F8D         RET
E0126 0537 2A21  E12    LAC     TEMP1,10 ; exp=12
E0127 0538 5821         SACH    TEMP1
E0128 0539 7E36         LACK    FLTSFT+12
E0129 053A 7F8D         RET
E0130 053B 1C4C  EDTOE  SUB     ONE,12  ; TEMP1-8192 -- exp=0
E0131 053C FD00         BGEZ    E0
      053D 04EE
E0132 053E 2921  E13    LAC     TEMP1,9  ; exp=13
E0133 053F 5821         SACH    TEMP1
E0134 0540 7E37         LACK    FLTSFT+13
E0135 0541 7F8D         RET
```

```
 0006                      COPY    INIT.ASM
F0001                *;*****************************************************
F0002                *; SYSTEM INITIALIZATION
F0003                *;*****************************************************
F0004      004A NOCONS EQU     74
F0005      0036 PTCONS EQU     54
F0006                *;
F0007 0542 7F81 RESET  DINT                          ; Disable interrupts
F0008 0543 6E00        LDPK    0                     ; Initialize data page
F0009                *;
F0010 0544 7035 SETPAC LARK    0,53
F0011 0545 6880        LARP    0
F0012 0546 7F89        ZAC                           ; Zero iram
F0013 0547 5080 ZRAMA  SACL    *,0,0
F0014 0548 F400        BANZ    ZRAMA
      0549 0547
F0015                *;
F0016 054A 7E01        LACK    .1
F0017 054B 504C        SACL    ONE
F0018 054C 6A4C        LT      ONE
F0019 054D 859B        MPYK    CONS
F0020 054E 7F8E        PAC                           ; ROM ADDR
F0021 054F 7136        LARK    1,PTCONS              ; RAM ADDR
F0022 0550 7049        LARK    0,NOCONS-1
F0023 0551 6881 NXCONS LARP    1
F0024 0552 67A0        TBLR    *+,0
F0025 0553 004C        ADD     ONE
F0026 0554 F400        BANZ    NXCONS
      0555 0551
F0027                *;
F0028 0556 4021        IN      TEMP1,CTL
F0029 0557 2021        LAC     TEMP1
F0030 0558 FA00        BLZ     ALAW
      0559 055E
F0031 055A FF00 MULAW  BZ      MULAWX
      055B 0020
F0032 055C F900        B       MULAWR
      055D 00F9
F0033 055E 7974 ALAW   AND     M32767
F0034 055F FF00        BZ      ALAWX
      0560 004E
F0035 0561 F900        B       ALAWR
      0562 0184
F0036                *;
```

```
 0007                  COPY    ROMRAM.ASM
G0001            *;
G0002            *;*****************************************************
G0003            *;     ROM
G0004            *;*****************************************************
G0005 0563       ROMLOC BSS     0
G0006            *; SHIFT MULT TABLE
G0007 0563       SHFT   BSS     0
G0008 0563 0000         DATA    0         ; log dq to flt dq exponent adjustment
G0009 0564 0001         DATA    1
G0010 0565 0002         DATA    2
G0011 0566 0004         DATA    4
G0012 0567 0008         DATA    8
G0013 0568 0010         DATA    16
G0014 0569 0020         DATA    32
G0015 056A 0040         DATA    64
G0016 056B 0080         DATA    128
G0017 056C 0100         DATA    256
G0018 056D 0200         DATA    512
G0019 056E 0400         DATA    1024
G0020 056F 0800         DATA    2048
G0021 0570 1000         DATA    4096
G0022 0571 2000         DATA    8192
G0023 0572 4000         DATA    16384
G0024            *; DQMAN MASK TABLE
G0025 0573 FE00         DATA    65024
G0026 0574 FF00         DATA    65280
G0027 0575 FF80         DATA    65408
G0028 0576 FFC0         DATA    65472
G0029 0577 FFE0         DATA    65504
G0030 0578 FFF0         DATA    65520
G0031 0579 FFF8         DATA    65528
G0032 057A FFF8         DATA    65528
G0033 057B FFF8         DATA    65528
G0034 057C FFF8         DATA    65528
G0035 057D FFF8         DATA    65528
G0036 057E FFF8         DATA    65528
G0037 057F FFF8         DATA    65528
G0038 0580 FFF8         DATA    65528
G0039 0581 FFF8         DATA    65528
G0040 0582 FFF8         DATA    65528
G0041            *; INVERSE QUANTIZING TABLE
G0042 0583       IQTAB  BSS     0
G0043 0583 FF79         DATA    65401
G0044 0584 0044         DATA    68
G0045 0585 00A5         DATA    165
G0046 0586 00E8         DATA    232
G0047 0587 011D         DATA    285
G0048 0588 014C         DATA    332
G0049 0589 0179         DATA    377
G0050 058A 01AC         DATA    428
G0051            *; WI TABLE
G0052 058B       WTABLE BSS     0
G0053 058B FFF4         DATA    65524
G0054 058C 0004         DATA    4
G0055 058D 001B         DATA    27
G0056 058E 0032         DATA    50
```

```
G0057 058F 0062         DATA    98
G0058 0590 00B8         DATA    184
G0059 0591 0154         DATA    340
G0060 0592 0454         DATA    1108
G0061            *; FI TABLE
G0062 0593       FITABL BSS     0
G0063 0593              BSS     0
G0064 0593 0000         DATA    0
G0065 0594 0000         DATA    0
G0066 0595 0000         DATA    0
G0067 0596 0010         DATA    16
G0068 0597 0010         DATA    16
G0069 0598 0010         DATA    16
G0070 0599 0030         DATA    48
G0071 059A 0070         DATA    112
G0072            *;
G0073            *; misc constants to be initialized
G0074            *;
G0075 059B       CONS   BSS     0
G0076 059B 0002         DATA    2    ; 12th entry of shift table (1st 11 all 0)
G0077 059C 0004         DATA    4
G0078 059D 0008         DATA    8
G0079 059E 0010         DATA    16
G0080 059F 0020         DATA    32
G0081 05A0 0040         DATA    64
G0082 05A1 0080         DATA    128
G0083 05A2 0100         DATA    256
G0084 05A3 0200         DATA    512
G0085 05A4 0400         DATA    1024
G0086 05A5 0800         DATA    2048
G0087 05A6 1000         DATA    4096
G0088 05A7 2000         DATA    8192
G0089 05A8 4000         DATA    16384
G0090 05A9 FFFF         DATA    -1
G0091 05AA FFFE         DATA    -2
G0092 05AB FFFC         DATA    -4
G0093 05AC 00FF         DATA    255      ;M255
G0094 05AD 8000         DATA    32768    ;K32768
G0095 05AE 0001         DATA    1        ;YLH
G0096 05AF 0800         DATA    2048     ;YLL
G0097 05B0 FFFF         DATA    -1
G0098 05B1 0001         DATA    1
G0099 05B2 0021         DATA    33       ; BIAS
G0100 05B3 0220         DATA    544      ;YU
G0101 05B4 0200         DATA    512      ;PK0
G0102 05B5 0200         DATA    512      ;PK1
G0103 05B6 0200         DATA    512      ;PK2
G0104 05B7 0100         DATA    256      ;SRMAN
G0105 05B8 0100         DATA    256      ;SR1MAN
G0106 05B9 0800         DATA    2048     ;SDQ
G0107 05BA 0800         DATA    2048     ;SDQ1
G0108 05BB 0800         DATA    2048     ;SDQ2
G0109 05BC 0800         DATA    2048     ;SDQ3
G0110 05BD 0800         DATA    2048     ;SDQ4
G0111 05BE 0800         DATA    2048     ;SDQ5
G0112 05BF 0800         DATA    2048     ;SDQ6
G0113 05C0 0100         DATA    256      ;DQMAN
```

```
G0114 05C1 0100          DATA     256       ;DQ1MAN
G0115 05C2 0100          DATA     256       ;DQ2MAN
G0116 05C3 0100          DATA     256       ;DQ3MAN
G0117 05C4 0100          DATA     256       ;DQ4MAN
G0118 05C5 0100          DATA     256       ;DQ5MAN
G0119 05C6 11E0          DATA     4576      ;K4576
G0120 05C7 3FFE          DATA     16382     ;K16382
G0121 05C8 C002          DATA     -16382    ;M16382
G0122 05C9 002E          DATA     46        ;K46
G0123 05CA 0031          DATA     49        ;K49
G0124 05CB 0563          DATA     SHFT      ;SHIFT
G0125 05CC 003F          DATA     63        ;K63
G0126 05CD 0000          DATA     0         ;FI
G0127 05CE 0000          DATA     0         ;WI
G0128 05CF 0583          DATA     IQTAB     ;INQTAB
G0129 05D0 0220          DATA     544       ;K544
G0130 05D1 1400          DATA     5120      ;K5120
G0131 05D2 000F          DATA     15        ;M15
G0132 05D3 FF80          DATA     -128      ;KFF80
G0133 05D4 007F          DATA     127       ;K127
G0134 05D5 FFC0          DATA     -64       ;MFFC0
G0135 05D6 1080          DATA     4224      ; BIAS*2**7
G0136 05D7 0FFF          DATA     4095      ;M4095
G0137 05D8 0000          DATA     0         ;spare
G0138 05D9 7FFF          DATA     32767     ;M32767
G0139 05DA FF00          DATA     -256      ;KFF00
G0140 05DB 0038          DATA     56        ;K56
G0141 05DC 03C0          DATA     960       ;K960
G0142 05DD 004F          DATA     79        ;K79
G0143 05DE 005F          DATA     95        ;K95
G0144 05DF 0082          DATA     130       ;K130
G0145 05E0 008A          DATA     138       ;K138
G0146 05E1 0905          DATA     2309      ;K2309
G0147 05E2 0003          DATA     3         ;THREE
G0148 05E3 0000          DATA     0         ;spare
G0149 05E4 0080          DATA     128       ;M0080
```

```
G0151                *;*************************************************************
G0152                *;       RAM
G0153                *;*************************************************************
G0154 05E5           RAMLOC BSS     0
G0155 0000                  DORG    0
G0156                *;
G0157                *;  RAM Location # 000
G0158 0000 0000             DATA    0         ; spare
G0159                *;
G0160                *;  RAM Location # 001
G0161 0001 0000      I      DATA    0         ; 32Kb output
G0162                *;
G0163                *;  RAM Location # 002
G0164 0002 0000      IM     DATA    0         ; 8-level version of I
G0165                *;
G0166                *;  RAM Location # 003
G0167 0003 0000      SE     DATA    0         ; signal estimate
G0168                *;
G0169                *;  RAM Location # 004
G0170 0004 0000      SEZ    DATA    0         ; partial signal estimate
G0171                *;
G0172                *;  RAM Location # 005
G0173 0005 0000      APP    DATA    0         ; unlimited speed control parm
G0174                *;
G0175                *;  RAM Location # 006
G0176 0006 0000      AL     DATA    0         ; limited speed control parm
G0177                *;
G0178                *;  RAM Location # 007
G0179 0007 0000      DMS    DATA    0         ; short term average of F
G0180                *;
G0181                *;  RAM Location # 008
G0182 0008 0000      DML    DATA    0         ; long term average of F
G0183                *;
G0184                *;  RAM Location # 009
G0185 0009 0000      Y      DATA    0         ; quantizer scale factor
G0186                *;
G0187                *;  RAM Location # 010
G0188 000A 0000      B1     DATA    0         ; 6th order predictor coefficient
G0189                *;
G0190                *;  RAM Location # 011
G0191 000B 0000      B2     DATA    0         ; 6th order predictor coefficient
G0192                *;
G0193                *;  RAM Location # 012
G0194 000C 0000      B3     DATA    0         ; 6th order predictor coefficient
G0195                *;
G0196                *;  RAM Location # 013
G0197 000D 0000      B4     DATA    0         ; 6th order predictor coefficient
G0198                *;
G0199                *;  RAM Location # 014
G0200 000E 0000      B5     DATA    0         ; 6th order predictor coefficient
G0201                *;
G0202                *;  RAM Location # 015
G0203 000F 0000      B6     DATA    0         ; 6th order predictor coefficient
G0204                *;
G0205                *;  RAM Location # 016
G0206 0010 0000      DQ     DATA    0         ; quantized diff signal
G0207                *;
```

```
G0208                *;  RAM Location # 017
G0209 0011 0000  A1      DATA     0        ; coefficients of 2nd order predictor
G0210                *;
G0211                *;  RAM Location # 018
G0212 0012 0000  A2      DATA     0        ; coefficients of 2nd order predictor
G0213                *;
G0214                *;  RAM Location # 019
G0215 0013 0000  SR      DATA     0        ; reconstructed signal frame k
G0216                *;
G0217                *;  RAM Location # 020
G0218 0014 0000  SR1     DATA     0        ; reconstructed signal frame k
G0219                *;
G0220                *;  RAM Location # 021
G0221 0015 0000  DQEXP_ DATA     0        ; exponent of DQ
G0222                *;
G0223                *;  RAM Location # 022
G0224 0016 0000  DQ1EXP DATA     0        ; exponent of DQ1
G0225                *;
G0226                *;  RAM Location # 023
G0227 0017 0000  DQ2EXP DATA     0        ; exp of DQ2
G0228                *;
G0229                *;  RAM Location # 024
G0230 0018 0000  DQ3EXP DATA     0        ; exp of DQ3
G0231                *;
G0232                *;  RAM Location # 025
G0233 0019 0000  DQ4EXP DATA     0        ; exp of DQ4
G0234                *;
G0235                *;  RAM Location # 026
G0236 001A 0000  DQ5EXP DATA     0        ; exp of DQ5
G0237                *;
G0238                *;  RAM Location # 027
G0239 001B 0000  SCRACH DATA     0        ; scrach variable
G0240                *;
G0241                *;  RAM Location # 028
G0242 001C 0000  SREXP  DATA     0        ; exp of SR
G0243                *;
G0244                *;  RAM Location # 029
G0245 001D 0000  SR1EXP DATA     0        ; exp of SR1
G0246                *;
G0247                *;  RAM Location # 030
G0248 001E 0000  SUM4    DATA     0        ; temp
G0249                *;
G0250                *;  RAM Location # 031
G0251 001F 0000  SUM5    DATA     0        ; temp
G0252                *;
G0253                *;  RAM Location # 032
G0254 0020 0000  SUM6    DATA     0        ; temp
G0255                *;
G0256                *;  RAM Location # 033
G0257 0021 0000  TEMP1   DATA     0        ; temp
G0258                *;
G0259                *;  RAM Location # 034
G0260 0022       SUM1    BSS      0        ; temp
G0261 0022 0000  TEMP2   DATA     0        ; temp
G0262                *;
G0263                *;  RAM Location # 035
G0264 0023       SUM2    BSS      0        ; temp
```

```
G0265 0023 0000  TEMP3   DATA     0        ; temp
G0266                *;
G0267                *;  RAM Location # 036
G0268 0024 0000  TEMP4   DATA     0        ; temp
G0269                *;
G0270                *;  RAM Location # 037
G0271 0025 0000  SUM3    DATA     0        ; temp
G0272                *;
G0273                *;  RAM Location # 038
G0274 0026 0000  SAMPLE DATA     0        ; Linear sample
G0275                *;
G0276                *;  RAM Location # 039
G0277 0027 0000  SUM7    DATA     0        ; temp storage of SR1*A1 tap
G0278                *;
G0279                *;  RAM Location # 040
G0280 0028 0000  SUM8    DATA     0        ; temp storage of SR2*A2 tap
G0281                *;
G0282                *;  RAM Location # 041
G0283 0029 0000  YOVER4 DATA     0        ; Y>>2
G0284                *;
G0285                *;  RAM Location # 042
G0286 002A 0000          DATA     0        ; first location of shift table
G0287                *;
G0288                *;  RAM Location # 043
G0289 002B 0000          DATA     0
G0290                *;
G0291                *;  RAM Location # 044
G0292 002C 0000          DATA     0
G0293                *;
G0294                *;  RAM Location # 045
G0295 002D 0000          DATA     0
G0296                *;
G0297                *;  RAM Location # 046
G0298 002E 0000          DATA     0
G0299                *;
G0300                *;  RAM Location # 047
G0301 002F 0000          DATA     0
G0302                *;
G0303                *;  RAM Location # 048
G0304 0030 0000          DATA     0
G0305                *;
G0306                *;  RAM Location # 049
G0307 0031 0000          DATA     0
G0308                *;
G0309                *;  RAM Location # 050
G0310 0032 0000          DATA     0
G0311                *;
G0312                *;  RAM Location # 051
G0313 0033 0000          DATA     0
G0314                *;
G0315                *;  RAM Location # 052
G0316 0034 0000          DATA     0
G0317                *;
G0318                *;  RAM Location # 053
G0319 0035 0000          DATA     0
G0320                *;
G0321                *;  RAM Location # 054
```

```
G0322 0036 0000         DATA    0
G0323             *;
G0324             *;  RAM Location # 055
G0325 0037 0000         DATA    0
G0326             *;
G0327             *;  RAM Location # 056
G0328 0038 0000   EIGHT DATA    0
G0329             *;
G0330             *;  RAM Location # 057
G0331 0039 0000         DATA    0
G0332             *;
G0333             *;  RAM Location # 058
G0334 003A 0000         DATA    0
G0335             *;
G0336             *;  RAM Location # 059
G0337 003B 0000         DATA    0
G0338             *;
G0339             *;  RAM Location # 060
G0340 003C 0000         DATA    0
G0341             *;
G0342             *;  RAM Location # 061
G0343 003D 0000         DATA    0
G0344             *;
G0345             *;  RAM Location # 062
G0346 003E 0000         DATA    0
G0347             *;
G0348             *;  RAM Location # 063
G0349 003F 0000         DATA    0
G0350             *;
G0351             *;  RAM Location # 064
G0352 0040 0000         DATA    0
G0353             *;
G0354             *;  RAM Location # 065
G0355 0041 0000         DATA    0
G0356             *;
G0357             *;  RAM Location # 066
G0358 0042 0000         DATA    0
G0359             *;
G0360             *;  RAM Location # 067
G0361 0043 0000         DATA    0
G0362             *;
G0363             *;  RAM Location # 068
G0364 0044 0000         DATA    0
G0365             *;
G0366             *;  RAM Location # 069
G0367 0045 0000         DATA    0
G0368             *;
G0369             *;  RAM Location # 070
G0370 0046 0000         DATA    0        ; last loc of table (42-70)
G0371             *;
G0372             *;  RAM Location # 071
G0373 0047 0000   M255  DATA    0
G0374             *;
G0375             *;  RAM Location # 072
G0376 0048 0000   K32768 DATA   0        ; sign bit
G0377             *;
G0378             *;  RAM Location # 073
```

```
G0379 0049 0000   YLH    DATA    0      ; fast quant scale factor (hi word)
G0380             *;
G0381             *;  RAM Location # 074
G0382 004A 0000   YLL    DATA    0      ; slow quant scale factor (lo word)
G0383             *;
G0384             *;  RAM Location # 075
G0385 004B 0000   MINUS  DATA    0      ; -1
G0386             *;
G0387             *;  RAM Location # 076
G0388 004C 0000   ONE    DATA    0      ; 1
G0389             *;
G0390             *;  RAM Location # 077
G0391 004D 0000   BIAS   DATA    0      ; constant for mulaw conversions
G0392             *;
G0393             *;  RAM Location # 078
G0394 004E 0000   YU     DATA    0      ; fast quant scale factor
G0395             *;
G0396             *;  RAM Location # 079
G0397 004F 0000   PK0    DATA    0      ; sign of p(k)
G0398             *;
G0399             *;  RAM Location # 080
G0400 0050 0000   PK1    DATA    0      ; sign of p(k-1)
G0401             *;
G0402             *;  RAM Location # 081
G0403 0051 0000   PK2    DATA    0      ; sign of p(k-2)
G0404             *;
G0405             *;  RAM Location # 082
G0406 0052 0000   SRMAN  DATA    0      ; mantissa of SR
G0407             *;
G0408             *;  RAM Location # 083
G0409 0053 0000   SR1MAN DATA    0      ; mantissa of SR1
G0410             *;
G0411             *;  RAM Location # 084
G0412 0054 0000   SDQ    DATA    0      ; sign DQ(k)
G0413             *;
G0414             *;  RAM Location # 085
G0415 0055 0000   SDQ1   DATA    0      ; sign DQ(k-1)
G0416             *;
G0417             *;  RAM Location # 086
G0418 0056 0000   SDQ2   DATA    0      ; sign DQ(k-2)
G0419             *;
G0420             *;  RAM Location # 087
G0421 0057 0000   SDQ3   DATA    0      ; sign DQ(k-3)
G0422             *;
G0423             *;  RAM Location # 088
G0424 0058 0000   SDQ4   DATA    0      ; sign DQ(k-4)
G0425             *;
G0426             *;  RAM Location # 089
G0427 0059 0000   SDQ5   DATA    0      ; sign DQ(k-5)
G0428             *;
G0429             *;  RAM Location # 090
G0430 005A 0000   SDQ6   DATA    0      ; sign DQ(k-6)
G0431             *;
G0432             *;  RAM Location # 091
G0433 005B 0000   DQMAN  DATA    0      ; mantissa of DQ
G0434             *;
G0435             *;  RAM Location # 092
```

```
G0436 005C 0000  DQ1MAN DATA    0        ; mantissa of DQ1
G0437                  *;
G0438                  *;  RAM Location # 093
G0439 005D 0000  DQ2MAN DATA    0        ; mantissa of DQ2
G0440                  *;
G0441                  *;  RAM Location # 094
G0442 005E 0000  DQ3MAN DATA    0        ; mantissa of DQ3
G0443                  *;
G0444                  *;  RAM Location # 095
G0445 005F 0000  DQ4MAN DATA    0        ; mantissa of DQ4
G0446                  *;
G0447                  *;  RAM Location # 096
G0448 0060 0000  DQ5MAN DATA    0        ; mantissa of DQ5
G0449                  *;
G0450                  *;  RAM Location # 097
G0451 0061 0000  K4576  DATA    0        ; 4576
G0452                  *;
G0453                  *;  RAM Location # 098
G0454 0062 0000  K16382 DATA    0        ; +16382
G0455                  *;
G0456                  *;  RAM Location # 099
G0457 0063 0000  M16382 DATA    0        ; -16382
G0458                  *;
G0459                  *;  RAM Location # 100
G0460 0064 0000  K46    DATA    0        ; 46
G0461                  *;
G0462                  *;  RAM Location # 101
G0463 0065 0000  K49    DATA    0        ; 49
G0464                  *;
G0465                  *;  RAM Location # 102
G0466 0066 0000  SHIFT  DATA    0        ; SHIFT table address
G0467                  *;
G0468                  *;  RAM Location # 103
G0469 0067 0000  K63    DATA    0        ; 63
G0470                  *;
G0471                  *;  RAM Location # 104
G0472 0068 0000  FI     DATA    0        ; FI value
G0473                  *;
G0474                  *;  RAM Location # 105
G0475 0069 0000  WI     DATA    0        ; WI value
G0476                  *;
G0477                  *;  RAM Location # 106
G0478 006A 0000  INQTAB DATA    0        ; Inverse quan table address
G0479                  *;
G0480                  *;  RAM Location # 107
G0481 006B 0000  K544   DATA    0        ; 544
G0482                  *;
G0483                  *;  RAM Location # 108
G0484 006C 0000  K5120  DATA    0        ; 5120
G0485                  *;
G0486                  *;  RAM Location # 109
G0487 006D 0000  M15    DATA    0        ; 15
G0488                  *;
G0489                  *;  RAM Location # 110
G0490 006E 0000  KFF80  DATA    0        ; >FF80
G0491                  *;
G0492                  *;  RAM Location # 111
```

```
G0493 006F 0000  M127   DATA    0        ; 127
G0494                  *;
G0495                  *;  RAM Location # 112
G0496 0070 0000  MFFC0  DATA    0        ; -64
G0497                  *;
G0498                  *;  RAM Location # 113
G0499 0071 0000  BIASA  DATA    0        ; 33*128
G0500                  *;
G0501                  *;  RAM Location # 114
G0502 0072 0000  M4095  DATA    0        ; 4095
G0503                  *;
G0504                  *;  RAM Location # 115
G0505 0073 0000         DATA    0        ; spare
G0506                  *;
G0507                  *;  RAM Location # 116
G0508 0074 0000  M32767 DATA    0        ; 32767
G0509                  *;
G0510                  *;  RAM Location # 117
G0511 0075 0000  KFF00  DATA    0        ; >FF00
G0512                  *;
G0513                  *;  RAM Location # 118
G0514 0076 0000  K56    DATA    0        ; 56
G0515                  *;
G0516                  *;  RAM Location # 119
G0517 0077 0000  K960   DATA    0        ; 960
G0518                  *;
G0519                  *;  RAM Location # 120
G0520 0078 0000  K79    DATA    0        ; constants used for quantizing table
G0521                  *;
G0522                  *;  RAM Location # 121
G0523 0079 0000  K95    DATA    0        ; constants used for quantizing table
G0524                  *;
G0525                  *;  RAM Location # 122
G0526 007A 0000  K130   DATA    0        ; constants used for quantizing table
G0527                  *;
G0528                  *;  RAM Location # 123
G0529 007B 0000  K138   DATA    0        ; constants used for quantizing table
G0530                  *;
G0531                  *;  RAM Location # 124
G0532 007C 0000  K2309  DATA    0        ; constants used for quantizing table
G0533                  *;
G0534                  *;  RAM Location # 125
G0535 007D 0000  THREE  DATA    0        ; 3
G0536                  *;
G0537                  *;  RAM Location # 126
G0538 007E 0000         DATA    0        ; spare
G0539                  *;
G0540                  *;  RAM Location # 127
G0541 007F 0000  M0080  DATA    0        ; alaw mask
G0542                  *;
NO ERRORS, NO WARNINGS
```

```
CCITT       32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107     16:36:03  03-20-85
LABEL     VALUE   DEFN   REFERENCES                                   PAGE 0055

A1        0011    G0209  B0252 B0270 D0431 D0456 D0458 D0464 D0515 D0523
A1LIM     0426    D0520
A2        0012    G0212  B0230 B0245 D0486 D0493 D0496 D0508 D0513
ADC       0001    A0031  A0056 A0126
ADD18     03B3    D0336  D0328
ADDA      035C    D0085
ADDC      03E1    D0413
ADDONE    01A8    A0566  A0562
ADDSGN    036C    D0104
AL        0006    G0176  B0321 B0327 B0361
ALAW      055E    F0033  F0030
ALAWR     0184    A0511  A0512 F0035
ALAWX     004E    A0172  A0173 F0034
ALOG      035E    D0090
ANOMLE    019F    A0559  A0557
ANOMLY    01AF    A0572  A0564
APP       0005    G0173  B0322 B0325 D0323 D0324 D0325 D0336 D0338
APRED     03B6    D0345  D0335
AQUAN     02AA    C0024  A0099 A0168 A0351 A0504
B1        000A    G0188  B0167 B0183 D0396 D0398 D0402
B2        000B    G0191  B0144 B0160 D0389 D0391 D0395
B3        000C    G0194  B0121 B0137 D0382 D0384 D0388
B4        000D    G0197  B0098 B0114 D0375 D0377 D0381
B5        000E    G0200  B0075 B0091 D0368 D0370 D0374
B6        000F    G0203  B0053 B0068 D0361 D0363 D0367
BIAS      004D    G0391  A0082 A0089 A0253 A0264 A0272 A0282 A0290 A0302 A0310
                         A0320 A0334 A0346 A0426 A0436 A0444 A0456 A0464 A0474
                         A0488
BIASA     0071    G0499  A0180 A0184 A0188 A0192 A0196 A0200 A0204
C0TO1     02B8    C0075
C0TO3     02B5    C0073
C0TO7     02B2    C0071
C2TO3     02C3    C0083  C0074
C4TO5     02D1    C0093
C4TO7     02CE    C0091  C0072
C6TO7     02DC    C0101  C0092
C8TO11    02EA    C0111
C8TO14    02E7    C0109  C0070
C8TO9     02ED    C0113
CATOB     02FC    C0125  C0112
CCITT     0002    A0033  A0101 A0170 A0210 A0364
CCTOD     030E    C0139
CCTOE     030B    C0137  C0110
CHK1      01C4    B0068  E0007
CHK11     025F    B0245  E0031
CHK2      01DE    B0091  E0011
CHK21     0279    B0268  E0035
CHK3      01F8    B0114  E0015
CHK4      0212    B0137  E0019
CHK5      022C    B0160  E0023
CHK6      0246    B0183  E0027
CHKHI     0386    D0186  D0183
C10TO1    032C    C0187
C10TO3    0329    C0185
C12TO3    0335    C0193  C0186
C14TO7    033E    C0199  C0184
C15TO6    0341    C0201
```

```
CCITT       32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107     16:36:03  03-20-85
LABEL     VALUE   DEFN   REFERENCES                                   PAGE 0056

C16TO7    034A    C0207  C0200
CLNUP     00EA    A0342  A0330
CLNUPA    0176    A0496  A0484
CMPRSA    010B    A0405
CMPRSU    007F    A0251
CONS      059B    G0075  F0019
CTL       0000    A0034  F0028
D0TO1     0438    D0559
D0TO3     0435    D0557
D0TO7     0432    D0555
D2TO3     043D    D0564  D0558
D4TO5     044D    D0578
D4TO7     044A    D0576  D0556
D6TO7     045A    D0590  D0577
D8TO9     046F    D0608
D8TOB     046C    D0606
D8TOF     0469    D0604  D0554
DAC       0001    A0032  A0357 A0510
DATOB     0480    D0624  D0607
DCTOD     0494    D0642
DCTOF     0491    D0640  D0605
DETOF     04A5    D0658  D0641
DML       0008    G0182  D0293 D0294 D0295 D0329 D0332
DMS       0007    G0179  D0273 D0274 D0275 D0331
DO32KA    0106    A0376  A0369
DO32KU    007A    A0222  A0215
DONEC     041C    D0508
DONEF     03F9    D0442  D0437
DQ        0010    G0206  D0103 D0110 D0111 D0416 D0548
DQ1EXP    0016    G0224  B0146 B0147
DQ1MAN    005C    G0436  B0150
DQ2EXP    0017    G0227  B0123 B0124
DQ2MAN    005D    G0439  B0127
DQ3EXP    0018    G0230  B0100 B0101
DQ3MAN    005E    G0442  B0104
DQ4EXP    0019    G0233  B0077 B0078
DQ4MAN    005F    G0445  B0081
DQ5EXP    001A    G0236  B0055
DQ5MAN    0060    G0448  B0058
DQEXP     0015    G0221  B0169 B0170 D0091 D0095
DQMAN     005B    G0433  B0173 D0094 D0101 D0126 D0127 D0128 D0130
E0        04EE    E0062  E0131
E0TO1     04EB    E0060
E0TO3     04E8    E0058
E0TO7     04E5    E0056
E1        04F2    E0066  E0061
E10       0529    E0114  E0109
E11       0533    E0122
E12       0537    E0126  E0121
E13       053E    E0132
E2        04F9    E0072
E2TO3     04F6    E0070  E0059
E3        04FD    E0076  E0071
E4        0504    E0082
E4TO6     0501    E0080  E0057
E5        050B    E0088
E5TO6     0508    E0086  E0081
```

```
CCITT        32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107     16:36:03  03-20-85
LABEL     VALUE   DEFN   REFERENCES                                    PAGE 0057

E6        050F    E0092  E0087
E7        051A    E0100
E7TO8     0517    E0098
E7TOA     0514    E0096
E7TOD     0511    E0094  E0055
E8        051E    E0104  E0099
E9        0525    E0110
E9TOA     0522    E0108  E0097
EBTOC     0530    E0120
EBTOD     052D    E0118  E0095
EDTOE     053B    E0130  E0119
EIGHT     0038    G0328  A0541
EXP0      02BB    C0077
EXP1      02BF    C0080  C0076
EXP10     02FF    C0127
EXP11     0305    C0132  C0126
EXP12     0311    C0141
EXP13     0317    C0146  C0140
EXP14     031D    C0151  C0138
EXP2      02C6    C0085
EXP3      02CA    C0088  C0084
EXP4      02D4    C0095
EXP5      02D8    C0098  C0094
EXP6      02DF    C0103
EXP7      02E3    C0106  C0102
EXP8      02F0    C0115
EXP9      02F6    C0120  C0114
EXPNDA    0035    A0146
EXPNDU    0005    A0076
EXX01     0439    DQ560
EXX10     0483    D0626
EXX11     048A    D0633  D0625
EXX12     0497    D0644
EXX13     049E    D0651  D0643
EXX14     04A8    D0660
EXX15     04AF    D0667  D0659
EXX2      0440    D0566
EXX3      0445    D0571  D0565
EXX4      0450    D0580
EXX5      0455    D0585  D0579
EXX6      045D    D0592
EXX7      0462    D0597  D0591
EXX8      0472    D0610
EXX9      0479    D0617  D0609
FI        0068    G0472  D0051 D0272 D0292
FILTA     039C    D0272
FILTB     03A0    D0292
FILTC     03A4    D0323
FILTD     037C    D0175
FILTE     038B    D0209  D0187
FINI      00E6    A0338  A0268 A0276 A0286 A0294 A0306 A0314 A0324
FINISH    0172    A0492  A0422 A0430 A0440 A0448 A0460 A0468 A0478
FITABL    0593    G0062
FLOAT     04DE    E0050  B0054 B0076 B0099 B0122 B0145 B0168 B0231 B0253
FLTDQ     0377    D0126  D0108
FLTSFT    002A    E0047  E0064 E0068 E0074 E0078 E0084 E0090 E0092 E0102 E0106
                         E0112 E0116 E0124 E0128 E0134
```

```
CCITT        32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107     16:36:03  03-20-85
LABEL     VALUE   DEFN   REFERENCES                                    PAGE 0058

FLTSR     042A    D0548  D0519
GETA1     03FA    D0456  D0435 D0440
GETA2     0404    D0481
GETB1     03DA    D0396
GETB2     03D3    D0389
GETB3     03CC    D0382
GETB4     03C5    D0375
GETB5     03BE    D0368
GETB6     03B6    D0360
GETEXP    02AF    C0069
GETF      03EA    D0431
GETF1     03EE    D0434
GETF2     03F4    D0438  D0433
GETI      0018    A0098
GETIM     0351    C0212  C0190 C0192 C0196 C0198 C0204 C0206 C0210
GETMAN    0321    C0155  C0079 C0082 C0087 C0090 C0097 C0100 C0105 C0108 C0119
                         C0124 C0131 C0136 C0145 C0150
GETSE     024E    B0230
I         0001    G0161  A0100 A0101 A0169 A0170 A0210 A0212 A0364 A0366 A0544
                         D0106
IDEQIM    01B1    A0574  A0547
IDGTIM    01A0    A0560  A0546
IDLTIM    0191    A0548
IEQ0      032F    C0189
IEQ1      0332    C0191  C0188
IEQ2      0338    C0195
IEQ3      033B    C0197  C0194
IEQ4      0344    C0203
IEQ5      0347    C0205  C0202
IEQ6      034D    C0209
IEQ7      0350    C0211  C0208
IM        0002    G0164  A0213 A0216 A0218 A0367 A0370 A0372 C0212 D0045
INQTAB    006A    G0478  D0046
INTRPT    0002    A0050  A0050
IQTAB     0583    G0042  G0128
ITAB1     07F9    C0175
ITAB2     087B    C0176
ITAB3     08CA    C0177
ITAB4     0905    C0178
ITAB5     0936    C0179
ITAB6     0964    C0180
ITAB7     0995    C0181
K130      007A    G0526  C0187
K138      007B    G0529  C0185
K16382    0062    G0454  D0436
K2309     007C    G0532  C0183
K32768    0048    G0376  A0147 B0070 B0093 B0116 B0139 B0162 B0185 B0247 B0271
K4576     0061    G0451  D0186
K46       0064    G0460  C0201
K49       0065    G0463  C0207
K5120     006C    G0484  D0188
K544      006B    G0481  D0182 D0184
K56       0076    G0514  E0056
K63       0067    G0469  A0327 A0481
K79       0078    G0520  C0193
K95       0079    G0523  C0199
K960      0077    G0517  D0604 E0094
```

```
CCITT       32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107     16:36:03  03-20-85
LABEL      VALUE   DEFN   REFERENCES                                   PAGE 0059

KFF00      0075   G0511   A0077
KFF80      006E   G0490   B0062 B0085 B0108 B0131 B0154 B0177 B0239 B0262
LIMA       028E   B0320
LIMB       0380   D0182
LIMC       0415   D0502
LIMD       041D   D0512   D0504
M0080      007F   G0541   A0508
M127       006F   G0493   A0549 A0558 A0563 A0566 C0155
M15        006D   G0487   A0217 A0261 A0269 A0279 A0287 A0299 A0307 A0317 A0331
                          A0371 A0415 A0423 A0433 A0441 A0453 A0461 A0471 A0485
                          C0071 C0109 C0214 D0512 D0555
M16382     0063   G0457   D0441
M255       0047   G0373   A0343 A0355 A0497
M32767     0074   G0508   A0079 A0149 D0217 D0224 E0005 E0009 E0013 E0017 E0021
                          E0025 E0029 E0033 F0033
M4095      0072   G0502   A0081 A0151 B0365 D0092
MAXNEG     01AD   A0570   A0567
MAXPOS     0199   A0554   A0551
MFFC0      0070   G0496   B0326
MINUS      004B   G0385   D0112
MIX        0297   B0353   B0324
MULAW      055A   F0031
MULAWR     00F9   A0358   A0359 F0032
MULAWX     0020   A0103   A0104 F0031
NEG1       01C9   B0072
NEG11      0264   B0249
NEG2       01E3   B0095
NEG21      027F   B0273
NEG3       01FD   B0118
NEG4       0217   B0141
NEG5       0231   B0164
NEG6       024B   B0187
NOCONS     004A   F0004   F0022
NONNEG     02A5   B0366   B0364
NORMAL     00DF   A0331   A0326
NORMLA     016B   A0485   A0480
NXCONS     0551   F0023   F0026
ONE        004C   G0388   A0176 A0214 A0255 A0259 A0277 A0297 A0315 A0325 A0368
                          A0409 A0413 A0418 A0431 A0451 A0469 A0479 A0543 A0552
                          A0556 A0561 A0568 B0320 B0323 C0069 C0075 C0083 C0093
                          C0101 C0113 C0125 C0139 C0165 D0048 D0050 D0090 D0093
                          D0099 D0104 D0107 D0337 D0419 D0434 D0439 D0553 D0561
                          D0564 D0578 D0590 D0608 D0624 D0642 D0658 E0054 E0062
                          E0070 E0080 E0086 E0098 E0108 E0120 E0130 F0017 F0018
                          F0025
PK0        004F   G0397   D0414 D0420 D0421
PK1        0050   G0400   D0413 D0460
PK2        0051   G0403   D0488
POS1       01CC   B0075   B0071
POS11      0267   B0252   B0248
POS2       01E6   B0098   B0094
POS21      0282   B0276   B0272
POS3       0200   B0121   B0117
POS4       021A   B0144   B0140
POS5       0234   B0167   B0163
POS6       024E   B0190   B0186
PRDICT     0355   D0045   A0102 A0171 A0223 A0377
```

```
CCITT       32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107     16:36:03  03-20-85
LABEL      VALUE   DEFN   REFERENCES                                   PAGE 0060

PTCONS     0036   F0005   F0021
QDONE      0354   C0215
QUAN       0326   C0183
RAMLOC     05E5   G0154
RCVA       00FD   A0364   A0511
RCVMU      0071   A0210   A0358
RESET      0542   F0007   A0043
ROMLOC     0563   G0005
RS1        04B6   E0004   B0066
RS11       04D4   E0028   B0243
RS2        04BB   E0008   B0089
RS21       04D9   E0032   B0266
RS3        04C0   E0012   B0112
RS4        04C5   E0016   B0135
RS5        04CA   E0020   B0158
RS6        04CF   E0024   B0181
SAMPLE     0026   G0274   A0090 A0091 A0094 A0158 A0159 A0162 A0265 A0273 A0283
                          A0291 A0303 A0311 A0321 A0328 A0335 A0345 A0349 A0419
                          A0427 A0437 A0445 A0457 A0465 A0475 A0482 A0489 A0499
                          A0502 C0024
SATCH      00DA   A0327
SATCHA     0166   A0481
SBASE      0024   A0106   A0086
SBASEA     0052   A0175   A0155
SCAL0A     0118   A0415
SCAL1A     0121   A0423   A0414
SCAL2A     012D   A0433
SCAL3A     0136   A0441   A0432
SCAL4A     0145   A0453
SCAL5A     014E   A0461   A0452
SCAL6A     015A   A0471
SCAL7A     0163   A0479   A0470
SCALE0     008C   A0261
SCALE1     0095   A0269   A0260
SCALE2     00A1   A0279
SCALE3     00AA   A0287   A0278
SCALE4     00B9   A0299
SCALE5     00C2   A0307   A0298
SCALE6     00CE   A0317
SCALE7     00D7   A0325   A0316
SCL021     0089   A0259
SCL023     0086   A0257
SCL0T1     0115   A0413
SCL0T3     0112   A0411
SCL223     009E   A0277   A0258
SCL2T3     012A   A0431   A0412
SCL425     00B6   A0297
SCL427     00B3   A0295   A0256
SCL4T5     0142   A0451
SCL4T7     013F   A0449   A0410
SCL627     00CB   A0315   A0296
SCL6T7     0157   A0469   A0450
```

```
CCITT       32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107    16:36:03  03-20-85
LABEL     VALUE  DEFN   REFERENCES                                   PAGE 0061

SCRACH    001B   G0239  A0056 A0076 A0083 A0106 A0108 A0110 A0112 A0114 A0116
                        A0118 A0120 A0126 A0146 A0152 A0175 A0179 A0183 A0187
                        A0191 A0195 A0199 A0203 A0254 A0262 A0263 A0266 A0270
                        A0271 A0274 A0280 A0281 A0284 A0288 A0289 A0292 A0300
                        A0301 A0304 A0308 A0309 A0312 A0318 A0319 A0322 A0332
                        A0333 A0336 A0338 A0339 A0344 A0356 A0357 A0408 A0416
                        A0417 A0420 A0424 A0425 A0428 A0434 A0435 A0438 A0442
                        A0443 A0446 A0454 A0455 A0458 A0462 A0463 A0466 A0472
                        A0473 A0476 A0486 A0487 A0490 A0492 A0493 A0498 A0509
                        A0510 A0548 A0560 A0574
SDQ       0054   G0412  D0105 D0113 D0365 D0372 D0379 D0386 D0393 D0400 D0401
SDQ1      0055   G0415  B0184 D0394
SDQ2      0056   G0418  B0161 D0387
SDQ3      0057   G0421  B0138 D0380
SDQ4      0058   G0424  B0115 D0373
SDQ5      0059   G0427  B0092 D0366
SDQ6      005A   G0430  B0069 D0360
SE        0003   G0167  B0301 B0302 B0303 C0025 D0549
SETPAC    0544   F0010
SEZ       0004   G0170  B0298 D0415
SHFT      0563   G0007  G0124
SHIFT     0066   G0466  D0096
SIGDIF    01B3   B0053  A0098 A0167 A0222 A0376
SR        0013   G0215  A0247 A0401 B0268 B0269 D0550
SR1       0014   G0218  B0246
SR1EXP    001D   G0245  B0232
SR1MAN    0053   G0409  B0235
SREXP     001C   G0242  B0254 B0255 D0560 D0569 D0574 D0583 D0588 D0595 D0602
                        D0615 D0622 D0631 D0638 D0649 D0656 D0665 D0672
SRMAN     0052   G0406  B0258 D0552 D0559 D0562 D0566 D0567 D0571 D0572 D0580
                        D0581 D0585 D0586 D0592 D0593 D0597 D0598 D0599 D0600
                        D0610 D0611 D0612 D0613 D0617 D0618 D0619 D0620 D0626
                        D0627 D0628 D0629 D0633 D0634 D0635 D0636 D0644 D0645
                        D0646 D0647 D0651 D0652 D0653 D0654 D0660 D0661 D0662
                        D0663 D0667 D0668 D0669 D0670
STRLIM    038A   D0189  D0185
SUBF      0409   D0486  D0481
SUBONE    019B   A0556  A0550
SUBTB     0324   C0165
SUM1      0022   G0260  B0056 B0057 B0067 B0073 B0074 B0297 E0006
SUM2      0023   G0264  B0079 B0080 B0090 B0096 B0097 B0296 E0010
SUM3      0025   G0271  B0102 B0103 B0113 B0119 B0120 B0295 E0014
SUM4      001E   G0248  B0125 B0126 B0136 B0142 B0143 B0294 D0487 D0494 E0018
SUM5      001F   G0251  B0148 B0149 B0159 B0165 B0166 B0293 E0022
SUM6      0020   G0254  B0171 B0172 B0182 B0188 B0189 B0292 E0026
SUM7      0027   G0277  B0233 B0234 B0244 B0250 B0251 B0299 E0030
SUM8      0028   G0280  B0256 B0257 B0267 B0274 B0275 B0300 E0034
SUMGTO    03E9   D0421
SYNC      0188   A0541  A0353 A0506
```

```
CCITT       32010 FAMILY MACRO ASSEMBLER   PC2.1 84.107    16:36:03  03-20-85
LABEL     VALUE  DEFN   REFERENCES                                   PAGE 0062

TEMP1     0021   G0257  A0093 A0148 A0153 A0154 A0160 A0161 A0340 A0341 A0494
                        A0495 A0542 A0545 B0060 B0063 B0064 B0083 B0086 B0087
                        B0106 B0109 B0110 B0129 B0132 B0133 B0152 B0155 B0156
                        B0175 B0178 B0179 B0237 B0240 B0241 B0260 B0263 B0264
                        B0360 B0362 C0068 C0078 C0081 C0086 C0089 C0096 C0099
                        C0104 C0107 C0116 C0117 C0118 C0121 C0122 C0123 C0128
                        C0129 C0130 C0133 C0134 C0135 C0142 C0143 C0144 C0147
                        C0148 C0149 C0152 C0153 C0154 C0156 C0157 D0047 D0085
                        D0210 D0211 D0216 D0221 D0362 D0364 D0369 D0371 D0376
                        D0378 D0383 D0385 D0390 D0392 D0397 D0399 D0417 D0418
                        D0498 D0506 D0507 D0514 D0518 D0520 E0050 E0051 E0053
                        E0060 E0063 E0066 E0067 E0072 E0073 E0076 E0077 E0082
                        E0083 E0088 E0089 E0100 E0101 E0104 E0105 E0110 E0111
                        E0114 E0115 E0122 E0123 E0126 E0127 E0132 E0133 F0028
                        F0029
TEMP2     0022   G0261  A0080 A0087 A0150 A0156 D0218 D0219 D0220 D0222 D0457
                        D0459
TEMP3     0023   G0265  B0354 B0356 B0357 B0359 B0366 D0097 D0098 D0100 D0129
                        D0330 D0334 D0432 D0442 D0483 D0484 D0491 D0492 D0495
TEMP4     0024   G0268  A0251 A0342 A0347 A0348 A0405 A0407 A0496 A0500 A0501
                        C0066 C0213 D0516 D0521 D0522
THREE     007D   G0535  A0257 A0295 A0411 A0449 B0059 B0082 B0105 B0128 B0151
                        B0174 B0236 B0259 C0073 C0091 C0111 C0137 D0327 D0503
                        D0505 D0557 D0576 D0606 D0640 E0058 E0096 E0118
WI        0069   G0475  D0049 D0177
WTABLE    058B   G0052
XMTA      0034   A0126  A0172
XMTMU     0004   A0056  A0103
Y         0009   G0185  B0367 B0368 D0175 D0176 D0326
YLH       0049   G0379  B0355 D0209 D0213 D0223
YLL       004A   G0382  B0353 D0212 D0214 D0225
YOVER4    0029   G0283  B0369 C0166 D0086
YU        004E   G0394  B0358 D0178 D0189 D0215
ZRAMA     0547   F0013  F0014
```