AN2834
Rev. 1, 9/2004

# Packet Telephony Automatic Level Control on the StarCore SC140 Core

by Lúcio F. C. Pessoa and Mao Zeng

This application note presents an automatic level control (ALC) design and implementation that fully complies with the ITU Recommendation G.169. The proposed design takes into account all the current ALC requirements and provides a clear mathematical formulation of the problem for a successful implementation on fixed-point architectures. The algorithm was efficiently implemented and extensively tested on a Freescale StarCore™-based MSC8101 DSP. The current implementation requires less than 0.3 million cycles per second (MCPS). The tests were based on standard ALC testing equipment and included both objective and subjective assessment. No apparent speech quality degradation was identified during real-time tests and no interference was observed on signaling tones for dual-tone multi-frequency (DTMF), text telephone (TTY), and facsimile communications. The tests demonstrate that the proposed ALC device is adequate for standard ALC digital telephony applications and can be used as a foundation for future extensions of the current ALC specification.

**CONTENTS**

# 1 Basics of Automatic Level Control

ALC, sometimes called automatic gain control (AGC), is a well-known application in communication systems with a given input signal conditioned to produce an output signal as constant in amplitude as possible, while supporting a wide gain range and controlled gain-reduction and gain-recovery characteristics. A nonlinear feedback loop controls the signal level by adjusting a linear gain in either the receive or transmit paths of the system. For example, an early hardware implementation includes a variable resistor, voltage amplifier, and feedback loop for adjusting a resistor control amplifier [1]. In most systems, the target signal level is a user-defined constant value.

ALC has become an important application in digital telephony systems using network-based equipment, where a restricted gain or loss is introduced in the transmit path to maintain the transmit signal level at a predetermined value. In this context, ALC is part of a broader class of voice quality enhancement (VQE) devices [2], which may include network echo cancellation [3], noise reduction, and other related signal enhancement processing blocks. When ALC is used in these systems, the overall speech quality should be kept within acceptable mean opinion score (MOS) levels [4], [5].

The International Telecommunication Union (ITU) developed the G.169 Recommendation [6] for network ALC devices to ensure network stability and minimize degradation of speech or voice-band signaling tones when the ALC device is in the communication network. However, the ITU does not specify a standard ALC algorithm, planning rules, or target levels for ALC devices. Furthermore, some critical deployment aspects, as well as possible extensions on either transmit or receive path directions, are still under development. The main G.169 requirements are as follows, and specific G.169 tests are defined to ensure that these three requirements are not violated:

- The ALC should not enhance the background noise.
- It should not track the level of echo signals.
- It should not distort signaling tones, such as DTMF, voice band data, and facsimile transmission.

An ALC device can be modeled mathematically as a linear gain function $g(n)$ that multiplies an input signal $x(n)$ and generates an output signal $y(n) = g(n)\,x(n)$. The input and output signal levels are estimated by filtering $|x(n)|^k$ and $|y(n)|^k$, k = 1, 2, . . . , with a low-pass filter. For simplicity, a standard single-pole low-pass filter with a Z transform $H(z) = (1 - a)/(1 - az^{-1})$, $0 < a < 1$ is employed to estimate signal levels. The analysis in the following derivation uses a second-order model (that is, $k = 2$). The input and output signal levels are defined in **Equation 1**.

**Equation 1**

$$P_x(n) = aP_x(n-1) + (1-a)x^2(n)$$

$$P_y(n) = aP_y(n-1) + (1-a)y^2(n)$$

An ALC device can be designed by minimizing the mean squared error signal $e(n) = P_{ref} - P_y(n)$, where $P_{ref}$ is a reference (target) signal level for $y(n)$. The gradient of the instantaneous squared error signal with respect to the gain $g(n)$ is given in **Equation 2**.

**Equation 2**

$$\frac{\partial e^2(n)}{\partial g(n)} = -2e(n)\frac{\partial P_y(n)}{\partial g(n)} \approx -4g(n)e(n)P_x(n)$$

Where we use the approximation $P_y(n) \approx g^2(n)P_x(n)$. The resulting LMS adaptation algorithm is defined in **Equation 3**.

**Equation 3**

$$g(n) = g(n-1)[1 + \mu P_x(n)(P_{ref} - P_y(n-1))]$$

where $\mu > 0$ is a step-size controlling the rate of change of the gain signal $g(n)$. A small adaptation step-size is typically required to avoid a level clipping effect on the signal. However, a very small step-size leads to undesirable adaptation speed, so there is a trade-off in choosing the step-size for gain adaptation. The optimal step-size depends on the predefined target signal level range of ALC devices. If $P_{ref} > P_y(n-1) \rightarrow g(n) > g(n-1)$, and if $P_{ref} < P_y(n-1) \rightarrow g(n) < g(n-1)$. Furthermore, as the input signal level $P_x(n)$ decreases, the gain adaptation tends to stop. The resulting gain adaptation sequence is:

1. Estimate the input signal level $P_x(n)$.
2. Update the gain signal $g(n)$.
3. Generate the output signal $y(n) = g(n)\,x(n)$.
4. Estimate the output signal level $P_y(n)$.
5. Repeat steps 1–4 for new samples of $x(n)$.

Depending on the bandwidth of the low-pass filter used to estimate signal levels, the ALC gain adaptation algorithm in **Equation 3** may generate a gain bias. However, this bias is negligible when narrowband filters are used, which is the preferred approach here.

If $x(n)$ is a wide-sense stationary (WSS) signal, the ALC adaptive algorithm, using the signal level estimation as per **Equation 1**, is expected to converge to an optimal gain, as shown in **Equation 4**:

**Equation 4**

$$g_{opt}(a) = \sqrt{P_{ref}\frac{E\{P_x(n)\}}{E\{P_x^2(n)\}}} = \sqrt{\frac{P_{ref}(1+a)E\{x^2(n)\}}{2aE\{P_x(n-1)x^2(n)\} + (1-a)E\{x^4(n)\}}}$$

where $E\{\cdot\}$ denotes statistical expectation. If the gain adaptation is implemented using a narrowband low-pass filter, that is, one that uses a control parameter $a$ close to one, the resulting optimal gain can be approximated in **Equation 5**.

**Equation 5**

$$\bar{g} = \sqrt{\frac{P_{ref}}{E\{x^2(n)\}}}$$

On the other hand, if the gain adaptation is implemented using a wideband filter, that is, one that uses a control parameter $a$ very close to zero, the resulting optimal gain becomes the one shown in **Equation 6**.

**Equation 6**

$$g_{opt}(0) = \sqrt{P_{ref}\frac{E\{x^2(n)\}}{E\{x^4(n)\}}}$$

Therefore, the expected bias of the output signal level for WSS signals is given in **Equation 7**.

**Equation 7**

$$g_b(a) = \frac{g_{opt}(a)}{\bar{g}} = \sqrt{\frac{(1+a)E\{x^2(n)\}^2}{2aE\{P_x(n-1)x^2(n)\} + (1-a)E\{x^4(n)\}}}$$

For example, the bias for a white Gaussian noise with zero mean and unit variance is as follows, which clearly converges to one for values of $a$ close to one.

$$g_b(a) = \sqrt{(1+a)/(3-a)}$$

# 2 Architecture

One major G.169 requirement to constrain the gain adaptation algorithm is that the ALC should not enhance the background noise. In the current design, the signal is considered background noise if $P_x(n) \le P_{min}$. Therefore, the background noise requirement can be met by stopping the gain adaptation and disabling gain amplification in the presence of pure background noise. That is, if the gain $g(n)$ is larger than one, temporarily set it to one; otherwise, keep it unchanged.

In the G.169 terminology, $x(n)$ is the send-in signal; the receive-in signal, denoted by $z(n)$, is the source of echo after it is reflected by a hybrid circuit [3]. To detect the presence of echo, the power level $P_z(n)$ of the receive-in signal $z(n)$ is estimated using the same single-pole low-pass filter that estimates $P_x(n)$ and $P_y(n))$ and compared with $P_{min}$. To take into account possible delay in the echo path, if $P_z(n) > P_{min}$ at least once in the latest $N$ samples, the current design assumes that echo is present. Therefore, the G.169 requirement that the ALC should not track the level of echo signals is observed if the gain adaptation is stopped when echo is detected. Both the background noise and the level tracking conditions can be handled at the same time using the following control signals:

**Equation 8**

$$e_x(n) = U(P_x(n) - P_{min})$$

$$e_z(n) = U\left(\sum_{k=n-N}^{n} U(P_z(k) - P_{min})\right)$$

where U (·) denotes the standard step function, which is equal to one if its argument is positive and to zero otherwise. Based on these control signals, $e_x(n) = 1$ indicates that the send-in signal is strong enough to be gain adjusted by the ALC device; $e_z(n) = 1$ indicates the presence of echo.

Since the ALC must be implemented in a fixed-point DSP, special signal conditioning must be defined. To keep the dynamic gain within predefined bounds, the total ALC gain $g_t(n)$ is decoupled into a static gain $g_s$ and a dynamic gain $g(n)$, such that $g_t(n) = g_s\, g(n)$. This step can also be viewed as a format-converter in a fixed-point implementation. During the adaptation, lower and upper bounds $g_{min}$ and $g_{max}$ are imposed on the dynamic gain. A unit total gain is obtained when $g(n) = G_{max} = 1 / g_s$. Finally, lower and upper bounds $y_{min}$ and $y_{max}$ are also imposed on the output signal from the ALC.

The gain adaptation sequence after the G.169 requirements are met is as follows:

1. Estimate the send-in and receive-in signal levels: $P_x(n)$ and $P_z(n)$.
2. Compute the control signals: $e_x(n)$ and $e_z(n)$.
3. Update the gain signal: $g_1(n) = g(n-1)[1 + \mu P_x(n)e_x(n)(P_{ref} - P_y(n-1))]$.
4. Constrain the gain signal: $g_2(n) = min\{max\{g_1(n), g_{min}\}, g_{max}\}$.
5. Define an intermediate gain signal: $g_3(n) = min\{g_2(n), G_{max}\}$.
6. Use the control signal $e_x(n)$: $g_4(n) = [g_2(n) - g_3(n)]e_x(n) + g_3(n)$.

7. Use the control signal $e_z(n)$: $g(n) = [g(n-1) - g_4(n)]e_z(n) + g_4(n)$.
8. Compute the total gain signal: $g_t(n) = g_s g(n)$.
9. Generate the output signal: $y(n) = min\{max\{g_t(n)x(n), y_{min}\}y_{max}\}$.
10. Estimate the ALC output signal level: $P_y(n)$.
11. Repeat steps 1–10 for new samples of $x(n)$.

**Figure 1** shows the resulting top-level structure of the ALC device. The send-in $x(n)$ and receive-in $z(n)$ signals are processed and generate a send-out $y(n)$ signal. The receive-in signal is employed to adjust the ALC functionality in the presence of echo. The total ALC gain signal $g_t(n) = g_s\, g(n)$ is the linear gain adjusting the send-in signal.

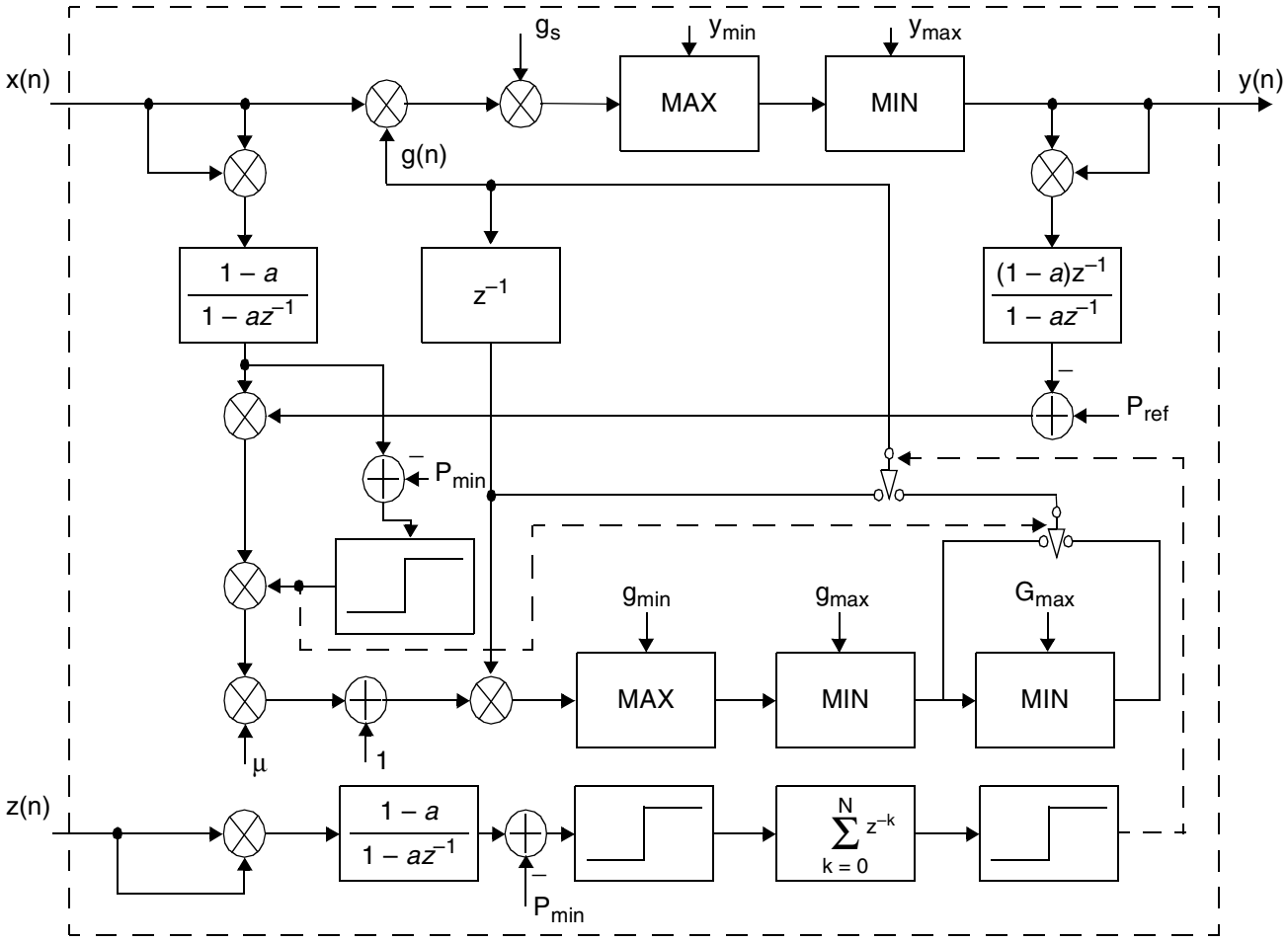


**Figure 1.** Top-Level Architecture of the Proposed ALC Device

Typical ALC control parameters for a 16-bit fixed-point implementation are:

- $g_s = 2^7$
- $G_{max} = 1 / g_s = 2^{-7}$
- $g_{min} = 2^{-15}$
- $g_{max} = G_{max} \times 10^{10/20} \sim 810 \times 2^{-15}$ (that is, 10 dB)
- $y_{min} = -1$

- $y_{max} = 1 - 2^{-15}$
- $a = 1 - 2^{-8}$
- $P_{min} = 10^{-(20 + 6.1824)/10} \sim 79 \times 2^{-15}$ (that is, –20 dBm0)
- $\mu = 8096 \times 2^{-15}$
- $N = 79$

# 3 Automatic Level Control on StarCore

This section describes an implementation of the ALC architecture presented in **Section 2** on a Freescale StarCore-based MSC8101 DSP. Extensive real-time tests demonstrated that the ALC software implementation requires less than 0.3 million cycles per second (MCPS), on average. A flexible application programming interface (API) dynamically enables/disables the ALC and dynamically changes the reference signal level $P_{ref}$. The code was integrated into a real-time framework using the MSC8101ADS board [7] and validated for G.169 compliance using the DSPG ECT-1 ALC tester equipment, as detailed in **Section 3.2**. **Figure 2** shows the basic connectivity for real-time objective testing. The ALC is integrated with an echo canceller (ECAN), and both are configured to process TDM data provided by a T1 connection (A) and controlled via the RS-232 serial communication interface.

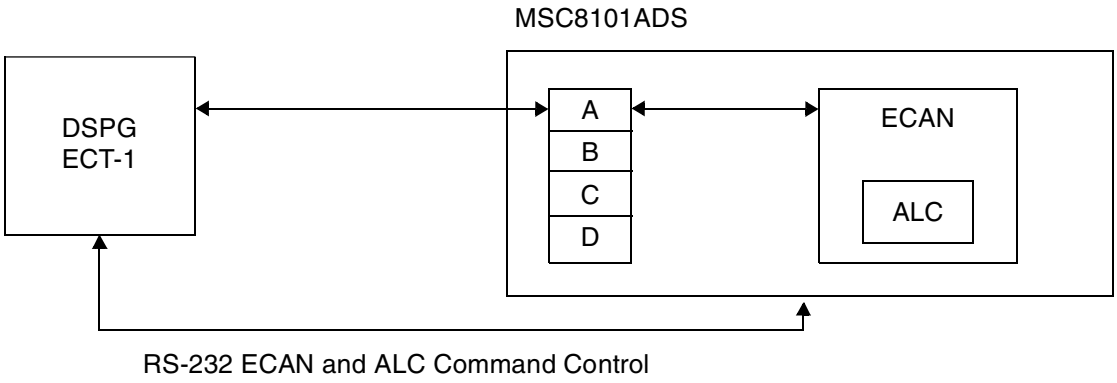


**Figure 2.** Connection Between MSC8101ADS and DSPG Tester

In additional real-time tests, the MSC8101ADS board was connected to telephones, facsimile machines, and TTY phones via a T1 connection. No degradation was observed on DTMF detection and TTY/facsimile communication. Subjective tests also revealed no apparent distortions.

The StarCore SC140 core is a flexible programmable DSP core that enables the emergence of computationally-intensive communications applications by providing exceptional performance, low power consumption, efficient compatibility, and compact code density. This core efficiently deploys a variable-length execution set (VLES) model that achieves maximum parallelism by allowing two address generation units and four data arithmetic logic units to execute multiple instructions in a single clock cycle.[1] This section briefly describes the fixed-point implementation of the ALC device on the SC140 core, focusing on two important issues in the fixed-point implementation: limited precision and computational efficiency.

1. For details, refer to the *SC140 DSP Core Reference Manual*, which is available at the web site listed on the back cover of this document.

## 3.1 Estimation of Signal Levels

The ALC device requires level estimations of three signals:

- $S_{in}$: Send-in signal $x(n)$.
- $S_{out}$: Send out signal $y(n)$.
- $R_{in}$: Receive-in signal $z(n)$.

The filtering process for the estimation is defined in **Section 1**. With 16-bit input data to be processed by the low-pass estimation filter, 32-bit data format is used for each signal level to preserve precision. To achieve computational efficiency on the SC140 DSP core, the low-pass filtering process is modified in the implementation as shown in **Equation 9**.

**Equation 9**

$$P_s(n) = P_s(n-1) - b(P_s(n-1) - s^2(n))$$

where $s(n)$ denotes the underlying signal whose level is being estimated. Furthermore, the smoothing factor $b = 1 - a$ is set to $2^{-k}$ so that shift operations can be used to avoid double-precision multiplication.

## 3.2 Gain Adaptation

The trade-off between data precision and computational efficiency is a crucial design issue. To increase computational efficiency, double-precision multiplication must be limited. However, fewer precision bits for variables may impair the adaptation capability of the device and even cause temporal clipping distortions. Therefore, we used a Q9.23 data format for the internal computations of the gain $g(n)$ to provide enough precision for both the integer and fractional parts. Meanwhile, 16-bit format is used for other variables to reduce the number of double-precision multiplication operations.

## 3.3 Auxiliary Functions

In initialization or control function calls, calculation of the function $10^{x/10}$ is used extensively to convert the system parameters from decibel to a fixed-point binary expression. The integer ($A_i$) and fractional ($A_f$) parts of this function can be calculated as follows:

**Equation 10**

$$A_i = 2^{(\lfloor ax \rfloor + 1)}$$

$$A_f = 2^{(ax - \lfloor ax \rfloor - 1)}$$

where $\lfloor \cdot \rfloor$ is the flooring function, which corresponds to the largest integer smaller than or equal to its argument, and $\alpha = log_2(10)/10 \approx 10885 \times 2^{-15}$. Then a fourth-order polynomial approximation of the following function is employed to evaluate the fractional bits, which prove to be sufficient for achieving 16-bit precision.

$$2^y \approx \sum_{k=0}^{4} c^k y^{4-k}$$

**Table 1** lists the coefficients of the fourth-order polynomial. The MMSE coefficients are optimized for values of $y$ between –1 and 0 (notice that the exponent of $A_f$ is in the range from –1 to 0). The coefficients of the standard Taylor expansion ($c_k = (ln2)^k / k!$) are included for comparison. The computation of the fourth-order polynomial requires three cycles on the SC140 DSP core:

- Cycle 1: $A = y \times y$, $B = c_1 y + c_2$, $C = c_3 y + c_4$
- Cycle 2: $D = A \times B + C$, $E = A \times A$
- Cycle 3: $F = c_0 E + D$

**Table 1.** Coefficients of the Polynomial Approximation of $2^y$

| Coefficients X $2^{15}$ | Taylor Coefficients | MMSE Approximation |
|---|---|---|
| $c_0$ | 315 | 224 |
| $c_1$ | 1819 | 1743 |
| $c_2$ | 7872 | 7844 |
| $c_3$ | 22713 | 22709 |
| $c_4$ | 32767 | 32767 |

**Figure 3** illustrates the typical dynamic behavior of the ALC StarCore DSP implementation.

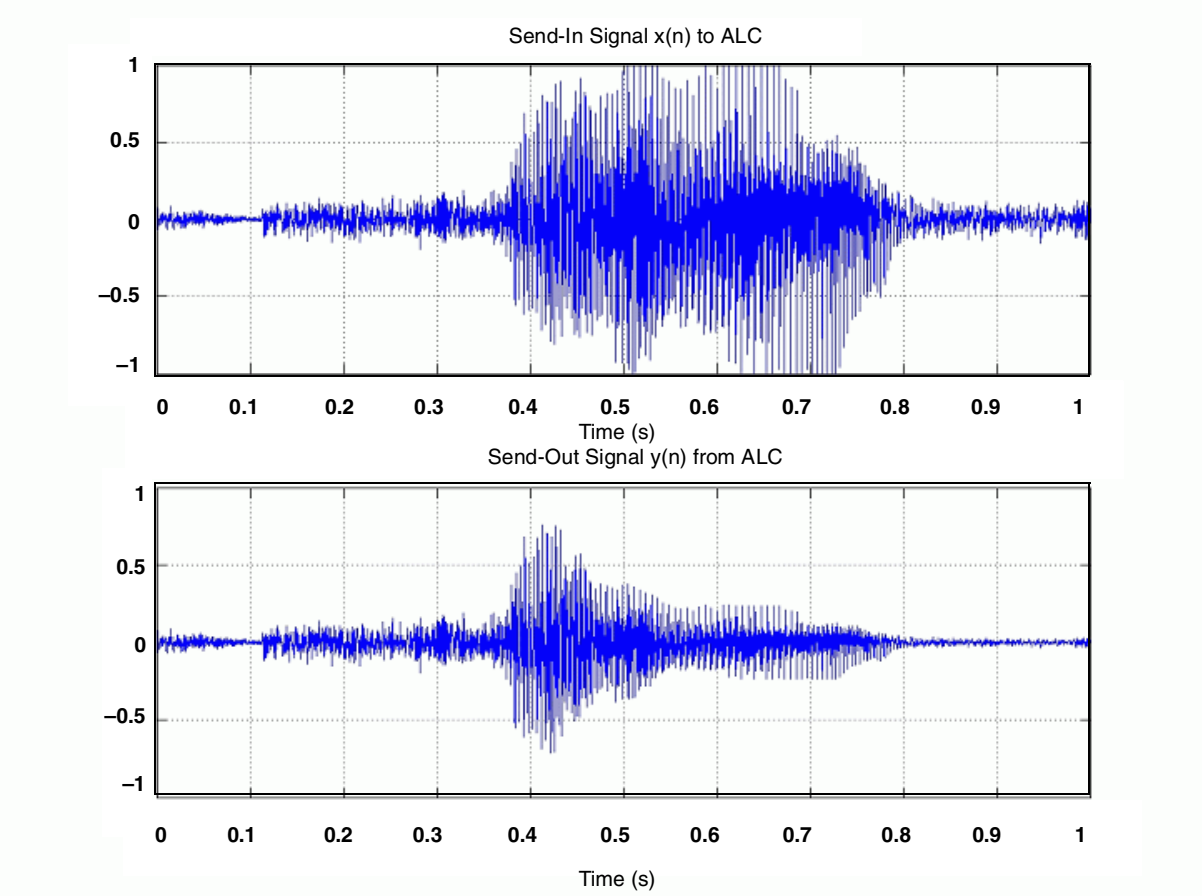


**Figure 3.** Typical Performance of the Proposed ALC Device

**Figure 4** shows the dynamic signal levels and total gain $g_t(n)$ corresponding to the signals illustrated in **Figure 3**, where the target reference level is set to $P_{ref} = -16$ dBm0.

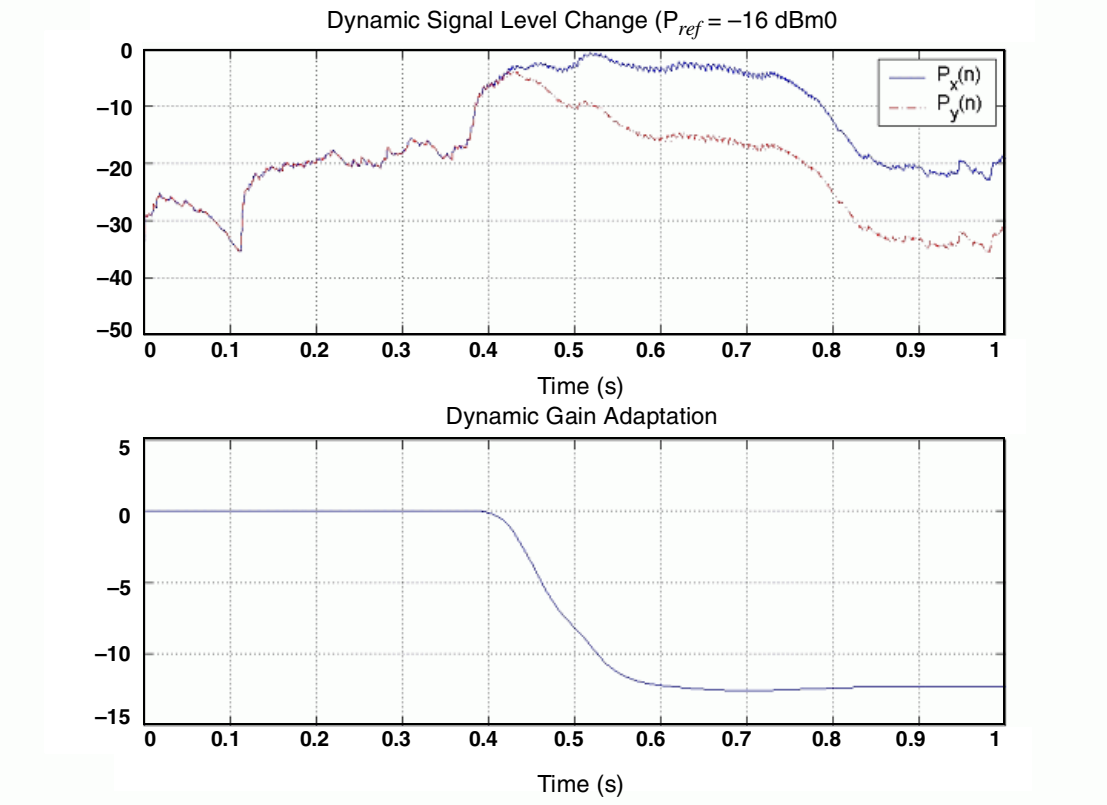


**Figure 4.** Dynamic Signal Levels and Total Gain

# 4 G.169 Compliance Test Results

All the objective tests reported in **Table 2** were performed on the DSPG ECT-1 ALC tester equipment with a T1 connection. The test coverage follows the ITU-T G.169 specification [6]. Because the ALC device is designed as a step-gain device, the Test 7.4.1 method 1 is not applicable. For this test, method 2 was used and it demonstrated the expected performance; that is, no degradation occurred after the ALC device was inserted into the system. Also, Tests 7.5.1 and 7.5.2 are not included here because they are not implemented in the ECT-1 ALC tester equipment.

**Table 2.** G.169 Test Coverage

| DSPG Test ID | G.169 Test Name |
|---|---|
| Test 7.1.2A | Stability test with echo cancellation. Derive initial gain. |
| Test 7.1.2B | Stability test with echo cancellation. Maximum gain in 1 second. |
| Test 7.1.2C | Stability test with echo cancellation. Maximum gain. |
| Test 7.2B | Echo robustness test with echo cancellation. |
| Test 7.3A | Noise tolerance test. Robustness to circuit noise. |
| Test 7.3B | Noise tolerance test. Robustness to environment noise. |

**Table 3** summarizes the test results, along with the corresponding test controls and test conditions. A sinusoid wave with a frequency of 1004 Hz was selected as the test signal. The tests indicate that selecting different signal types does not affect the results.

**Table 3.** G.169 Test Results Summary

| Number | Test | Test Conditions | | | | | Number of Tests | Number of Passes | Passing Rate (%) |
|---|---|---|---|---|---|---|---|---|---|
| | | ECAN | Target Level | ($S_{gen}$ in dBm0; $A_{echo}$ in dB; $T_d$ in ms) | | | | | |
| 1 | 7.1.2A | ON | –13 dBm0 | –20: 2: 0; | —; | — | 11 | 11 | 100 |
| 2 | 7.1.2B | ON | –13 dBm0 | –29: –5: –39; | —; | — | 9 | 9 | 100 |
| 3 | 7.1.2C | ON | –13 dBm0 | default; | —; | — | 18 | 18 | 100 |
| 4 | 7.2B | ON | –13 dBm0 | –2.0; | 6:6:30; | 0:10:60 | 35 | 35 | 100 |
| 5 | 7.3A | OFF | –13 dBm0 | –5.0; | —; | — | 5 | 5 | 100 |
| 6 | 7.3B | OFF | –13 dBm0 | –5.0; | —; | — | 5 | 5 | 100 |
| NOTE: X:Y:Z to be read as X for the lower/upper bound, Y for the incremental size, and Z for the upper/lower bound. | | | | | | | | | |

# 5 Conclusion

ALC is an important application in packet telephony systems for monitoring and adjusting the transmit signal level to improve voice quality over the communications network. The ITU-T G.169-compliant ALC design described in this application note takes into account all the current ALC requirements and provides a clear mathematical formulation of the problem for a successful implementation on fixed-point architectures. The ALC design was efficiently implemented and extensively tested in real-time on the Freescale StarCore-based MSC8101 DSP, requiring less than 0.3 million cycles per second (MCPS) on average. The ALC device was validated in conjunction with a carrier-class network echo canceller. The tests were based on standard ALC testing equipment and included both objective and subjective assessments. No apparent speech quality degradation was identified during real-time tests, and no interference was observed on such signaling tones as DTMF, TTY, and facsimile communication, which demonstrates that the proposed ALC device is adequate for standard ALC digital telephony applications and can be used as a foundation for future extensions of the ALC specification.

# 6 References

[1] G. A. Hellwarth and G. D Jones, “Automatic Conditioning of Speech Signals,” *IEEE Transactions on Audio and Electroacoustics*, vol. AU-16, no. 2, June 1968, pp. 169–179.

[2] ITU-T Recommendation G.160, “Voice Enhancement Devices,” *SG15, TD13* (WP 2/15) draft 9.1., 2003.

[3] R. A. Dyba, P. P. He, and L. F. C. Pessoa, *Network Echo Cancellers and Freescale Solutions using the StarCore SC140 Core*, Freescale Application Note AN2598/D, 2004.

[4] I. Cotanis, “Speech in the VQE Device Environment,” *IEEE Conference on Wireless Communications and Networking*, 2003, pp. 1102–1106.

[5] ITU-T Recommendation P.82, *Method for Evaluation of Service from the Standpoint of Speech Transmission Quality*, 1988.

[6] ITU-T Recommendation G.169, *Automatic Level Control Devices*, 1999.

[7] *MSC8101 Application Development System (MSC8101ADS) User’s Manual*, Metrowerks, Inc., 2001.

**NOTES:**

## *How to Reach Us:*

**Home Page:**
www.freescale.com

**E-mail:**
support@freescale.com

**USA/Europe or Locations not listed:**
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

**Europe, Middle East, and Africa:**
Freescale Halbleiter Deutschland GMBH
Technical Information Center
Schatzbogen 7
81829 München, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

**Japan:**
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu. Minato-ku
Tokyo 106-8573, Japan
0120 191014 or +81-3-3440-3569
support.japan@freescale.com

**Asia/Pacific:**
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T. Hong Kong
+800 2666 8080

***For Literature Requests Only:***
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com



AN2834
Rev. 1
9/2004