An All-Digital Automatic Gain Control

APPLICATION REPORT: SPRA081

Al Lovrich
Raj Chirayil
Digital Signal Processor Products
Semiconductor Group
Texas Instruments

Digital Signal Processing Solutions

TEXAS INSTRUMENTS

IMPORTANT NOTICE

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

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

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

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

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

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

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Copyright © 1997, Texas Instruments Incorporated

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

An All-Digital Automatic Gain Control

Abstract

This report describes an all-digital implementation of an AGC on a TMS320C17 DSP. The AGC is designed specifically for modem applications.

- The first section provides an overview of modem receiver structure and implementation.
- The second section discusses the AGC block diagram and the motivation for using an AGC in a modem receiver.
- The third section covers the AGC hardware and software implementation aspects on a TMS320C17 DSP.
- Appendix A provides QAM Signal Energy data.
- Appendix B gives an overview on Fractional Number Representation.

Product Support

World Wide Web

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

Email

For technical issues or clarification on switching products, please send a detailed email to dsph@ti.com. Questions receive prompt attention and are usually answered within one business day.

One of the basic structural blocks of a modem receiver is the Automatic Gain Control (AGC). The AGC is an adaptive system that operates over a wide dynamic range while maintaining the output signal at a constant level. This is necessary for the proper operation of the carrier recovery and clock recovery algorithms of the modem receiver.

This application report describes an all-digital implementation of an AGC on a TMS320C17 Digital Signal Processor (DSP). The AGC is designed specifically for modem applications. The structure of this application report is as follows:

- The first section provides an overview of modem receiver structure and implementation.
- Section two discusses the AGC block diagram and the motivation for using an AGC in a modem receiver.
- The last section covers the AGC hardware and software implementation aspects on a TMS320C17 DSP.

Introduction

A modem (MOdulator/DEModulator) is a device that modulates baseband signals at the transmitter and demodulates the received data at the receiver. To achieve full-duplex operation, frequency division multiplexing is employed, in which both modems simultaneously transmit and receive information over a single channel by dividing the telephone bandwidth into separate frequency bands: one for transmit with a carrier frequency of 1200 Hz and one for receive with a carrier frequency of 2400 Hz. A modem receiver consists of several functional blocks, which include answer/originate bandpass filters, AGC, demodulator, adaptive equalizer, clock recovery, carrier recovery, decision block, decoder, and descrambler.

In this report, we are concerned with the implementation of a DSP-based AGC for a V.22 bis modem product[1]. One of the basic structural blocks of a modem receiver is the AGC. The AGC is an adaptive system that operates over a wide dynamic range while maintaining the output signal at a constant level. The AGC is needed because several modules within the receiver use amplitude thresholds to make their decisions. These threshold levels must remain constant over the entire dynamic range of input signals, typically from –9 dbm to –43 dBm[2]. This is achieved through use of a software AGC, which multiplies the input signal with a gain factor, depending on the actual received signal level.

Modem Transmitter

The CCITT V.22 bis standard is a 2400-bps modem that uses Quadrature Amplitude Modulation (QAM) technique to transmit and receive data through the communications channel. This section presents an overview of QAM systems and the equations governing their operations.

In Quadrature Amplitude Modulation, the information is encoded as phase changes of the transmitted carrier and amplitude variations. With R denoting the amplitude and ϕ the phase change, the transmitted signal s(n) is mathematically represented as

$$s\ (n)\ =\ R\ \cos\ (\omega_c\ +\ \phi\ ) \tag{1}$$

where ω_c is the carrier frequency. Simplifying (1) and substituting $I_n = R\cos(\phi)$ and $Q_n = -R\sin(\phi)$ into it results in (2); this is used to describe QAM modulation systems.

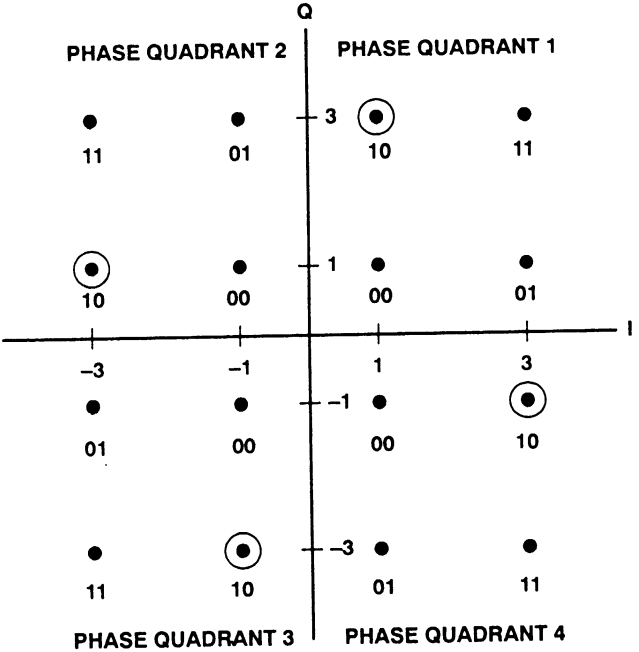
$$s(n) = I_n \cos(\omega_n) + Q_n \sin(\omega_n) \tag{2}$$

Transmission of a baseband sequence {In,Qn} is called *quadrature* transmission, with two carriers in phase quadrature to one another ($\cos \omega_c t$ and $\sin \omega_c t$) transmitted simultaneously over the same communications channel. Figure 1 shows a two-dimensional diagram of the signals of form (2) with the horizontal axis corresponding to the *in-phase* signal (I_n) and the vertical axis representing the *quadrature* signal (Q_n). These signal points are referred to as a 16-symbol QAM-signal constellation.

Each value of the $\{I_n,Q_n\}$ corresponds to one signaling element transmitted. The number of signaling elements per second is referred to as the baud rate. The *baud rate* is set by the CCITT V.22 bis recommendation to 600. By encoding four incoming bits (*quadbits*) in a single baud, transmission of 2400 bps is accomplished.

The encoding of the incoming data stream $d_s(n)$ into values of the sequence $\{I_n,Q_n\}$ is accomplished by the encoder. The encoder maps the first two bits of a quadbit as a phase quadrant change relative to the quadrant occupied by the preceding signal element. The last two bits of the quadbit define one of four signaling elements associated with the new quadrant[3].

Figure 1. V.22 bis Signal Constellation

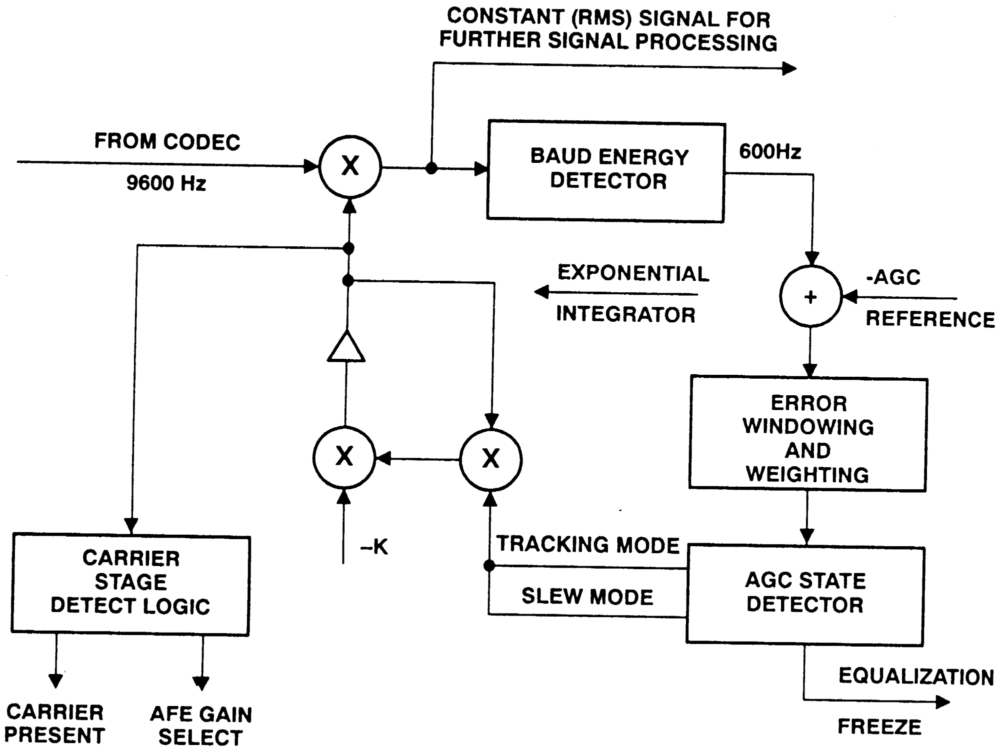


The AGC Algorithm

The AGC circuit is a closed-loop regulating system that maintains the output level of an amplifier at a constant level, even though the input signal may vary substantially. The AGC modeling and design techniques based on linear system design have been studied in detail[4]. The global stability of AGC loops assures the designer that the overall loop will stay stable under considerable weaker conditions if the proper design rules are followed[5].

Figure 2 is a block diagram of the modem automatic gain control. The AGC algorithm is partitioned into tasks performed once per sampling interval, and tasks performed once per baud interval. The sampling rate for the overall system is the designer's choice as long as it satisfies the Nyquist's criterion. A widely used sampling rate for the communications channel is 8 kHz. In the system in Figure 2, the sampling rate is chosen to be an integer multiple of the baud rate. Therefore, a sampling rate of 9.6 kHz is selected. This value is divisible by the master crystal frequency of 18.432 MHz.

Figure 2. Modem AGC Block Diagram



Baud Energy Detector

In Figure 2, every incoming linearized PCM sample is multiplied by the AGC gain factor. The result is available to the modem reciever for further signal processing. It is also used to update the baud energy detector. The energy of a baud interval is computed according to

$$E = \sum x_n{}^2 \tag{3}$$

where x_n represents the incoming samples. The accumulated baud energy is then compared against a reference level, which depends on the modulation scheme. This comparison is necessary to compute the AGC loop error signal. It is this error that the AGC is trying to minimize.

The QAM transmitted signal shown in (2) can be rewritten, taking waveform shaping into account as follows

$$s\ (t) = \sum I_n\ g(t-nT)\ \cos\ \omega_c\ t + \sum Q_n\ g(t-nT)\ \sin\ \omega_c\ t \tag{4}$$

where $\omega_c = 2\pi f_c$, where f_c = carrier frequency
$g(t)$ = shaping waveform
T = sampling interval
I_n, Q_n = data symbols

AGC Reference Energy

The signal energy for a particular constellation point (I_n,Q_n) is given by (see Appendix A)

$$E_n = 1/2\ (I_n{}^2 + Q_n{}^2) \tag{5}$$

The energy reference level is chosen to be

$$E_{ref} = E\ \{\ E_n\ \} \tag{6}$$

where E{ } denotes the expectation operation. The V.22 bis modem standard requires the transmitter to scramble the incoming digital sequence from the DTE and descramble the decoded data in the receiver[2,3]. The use of scrambler in the modem transmitter effectively randomizes the data and avoids data-dependent patterns in the transmitted sequence. This allows the constellation point sequences to be modeled as a random sequence, with each point having an equal probability of occurrence of $E\{(I_n,Q_n)\} = 1/N$. Therefore, (6) can be written as

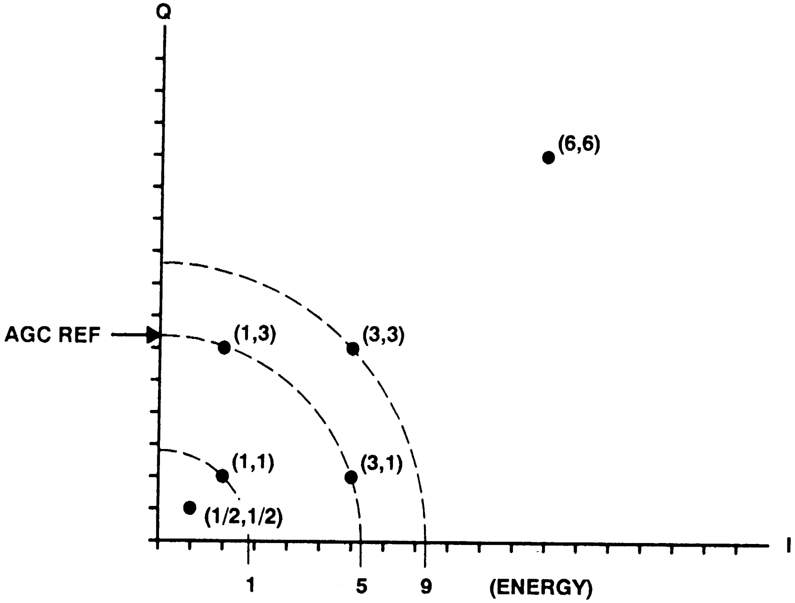
$$E_{ref} = \sum_{n=1}^{N} 1/N\ (\ E_n\ ) \tag{7}$$

Figure 3 shows a portion of the signal constellation diagram of a V.22 bis modem.

Applying (7) to all 16 constellation points results in

$$\begin{aligned} E_{ref} &= 1/16 \left\{ 4 \left[(\ 1^2 + 3^2\ ) + 1/2\ (\ 1^2 + 1^2\ ) + 1/2\ (3^2 + 3^2\ ) \right] \right\} \\ &= 1/16 \left\{ 4 \left[(\ 10\ ) + (\ 1\ ) + (\ 9\ ) \right] \right\} \\ &= 5 \end{aligned} \tag{8}$$

Figure 3. Signal Energy Constellation Diagram



In Figure 3, constellation points (3,3) and (1,1) with respective energy contents of 9 and 1 lie outside the reference level of 5. A window function is then necessary so that the AGC does not treat these energy variations around the nominal energy as distortions induced by the communication channel.

Therefore, the AGC should apply corrections when the incoming signal level is outside the interval (1,9)(see Figure 3). Such implementation, however, neglects the effects of intersymbol interference (ISI). ISI arises in systems whenever pulses are transmitted in a band-limited channel. In such channels, pulses tend not to die out immediately, and the tail from one pulse interferes with the next pulse. ISI-related effects are more easily shown when constant amplitude modulation techniques, such as DPSK, are considered. In a DPSK modem receiver, the received signal exhibits gain variations, that are entirely due to ISI. Since the modem equalizer compensates for ISI, the AGC should not act upon ISI-related signal-level variations, because this would introduce noise into the modem receiver and degrade the overall performance.

The received signal r(t) at the input of the receiver is the convolution of the channel impulse response h(t) with the transmitted symbols x_j in

$$r(t) = \sum_j x_j h(t - jT) + \mu(t) \tag{9}$$

where $\mu(t)$ is the additive white Gaussian noise. For the effects of ISI to be seen, the received signal must be sampled at the instant t_0+kT with t_0 incorporating the sampler phase and delay effects.

$$r(t_0 + kT) = x_k h(t_0) + \sum_{j \neq k} x_j h(t_0 + kT - jT) + \mu(t_0 + kt) \tag{10}$$

The first term of the right-hand side of (10) is the desired signal and is used to determine the transmitted symbol, while the middle term is ISI, which arises from the neighboring symbols [6]. With x_k, a constant amplitude sequence, the middle term in (10) results in received signal amplitude variations. Thus, the AGC design must incorporate an energy window around the energy reference level as defined by x_k's.

DSP Implementation

Hardware

This section describes the hardware requirements of the modem. The modem hardware consists of the following functional blocks:

1) Host Interface
2) DSP
3) Controller
4) Controller-DSP Interface
5) Analog Front-End
6) Telephone Line Interface

For the purpose of understanding the operation of the Automatic Gain Control (AGC), the discussion is limited to only the analog front end.

Modem Analog Front End

The function of the analog front end (AFE) in the modem is to convert the analog signals received on the telephone line to digital data that can be processed by a digital signal processing device, in this case the TMS320C17. Depending on the modem standard that is implemented, the modem AFE could further assist the DSP by preventing as many of the unwanted signals as possible from being received by the DSP. This reduces the signal conditioning and preprocessing required by the DSP, which, in turn, reduces the computational requirement.

In the implementation described here, the modem AFE performs the bandpass filtering, a single-step gain stage, and the A/D-D/A conversions. Although the modem hardware also includes the two-to-four wire conversion and the proper telephone line interface and impedance matching, it will not be considered in this discussion.

Split-band Filtering

In Frequency Division Multiplexing (FDM) modems, the originating and answering stations use different carrier frequencies to transmit data[2]. For V.22 bis modems, the originating modem transmits data using a 1200-Hz carrier and receives signals from the remote modem at 2400 Hz. Since these signals are carried over the two-wire Public Switched Telephone Network (PSTN) for a full duplex communication, both signals are present in the telephone line simultaneously. For a modem to prevent its transmitted signal from interfering with its received signal, it must eliminate its own transmit signal at its receiver. Since the two modems use separate carrier frequencies to transmit, this task becomes relatively easy. It is done by bandpass filtering the received signal with the passband filter being centered at the transmit carrier frequency of the remote modem.

This implementation uses a commercially available modem filter that has special modes to allow call-progress signal monitoring. This filter must provide adequate adjacent channel rejection while maintaining linear phase. The filter must operate over the entire dynamic range required by the modem, typically from 0 dBm to –43 dBm. For better Signal-to-Noise Ratio (SNR) and linear phase, it is desirable not to operate the filter and the Analog-to-Digital converter at very low signal levels. If signals are weak, an external gain stage (turned on/off under software control) in the receive signal path easily accomplishes this goal.

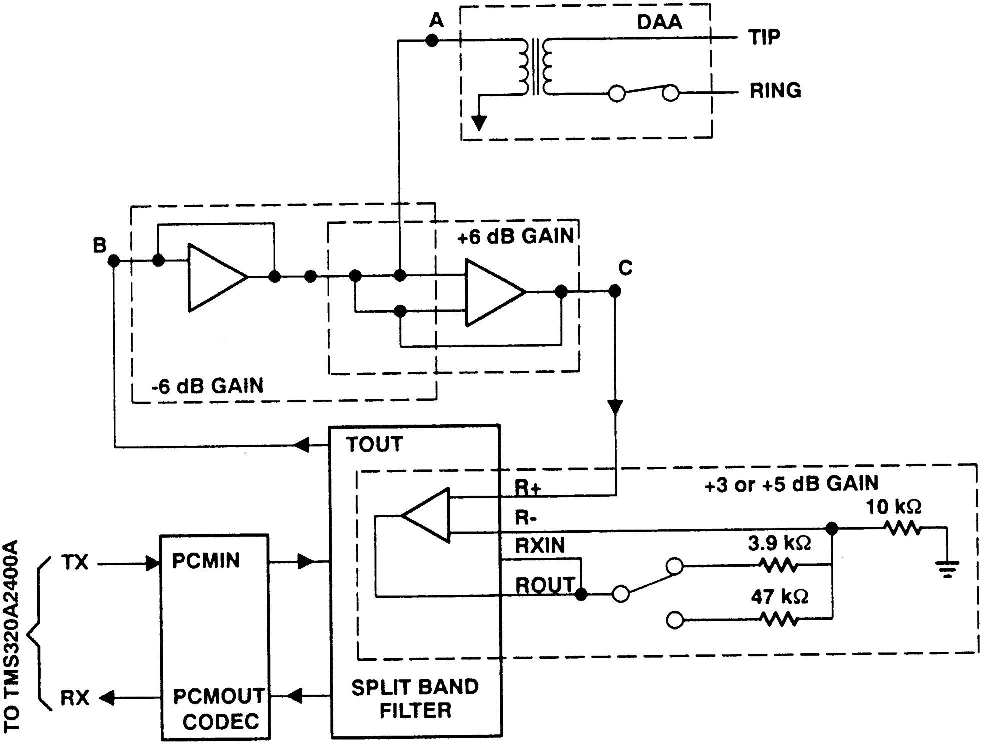
Hardware Gain Control

The hardware gain switch is implemented by changing the gain in the analog input buffer to the filter. When the average signal energy falls below –28 dBm, the DSP sets a status line to the modem controller. The controller, in turn, switches on a different resistor in the feedback circuit of the op-amp, increasing the gain by 12 dB. This switching is normally done only once during call initialization. However, if the connection starts with low-level signals and later the signals become stronger due to change in line impedance, the DSP resets this status line to the controller. The modem controller then turns off the external gain stage.

When the modem received signal is actually at the threshold level, it is possible that the external gain could frequently be turned on and off by slight changes in signal level. To prevent this, a 4-dB hysteresis has been established between external gain On and Off. This means the external gain will be turned On when the average signal level is less than –24 dBm and will be turned Off when the level is more than –28 dBm. Figure 4 shows the AFE schematic of the modem.

	LINE (dBm)	AFE GAIN (dB)	CODEC (dBm)
Rx level	–12 –24 –25 –43	0 0 12 12	–9 –21 –10 –28

Figure 4. Modem AFE Schematic



Codec Interface

The TMS320A2400A features hardware companding logic to interface directly to a µ-law codec[1]. The SCLK output provides the master clock frequency for the codec, and the FR provides the transmit and receive framing signal to the codec. Since the modem algorithm uses a 9.6-kHz sampling frequency, the codec must complete one A/D,D/A conversion at this rate.

The DSP serial port control register was programmed to provide an SCLK which is generated by dividing the DSP's input clock by ten. Thus, using an 18.432–MHz crystal as the DSP's clock input, a 1.8432-MHz SCLK was generated. The TCM29C19 uses an internal divide ratio of 192 to generate the 9.6-kHz sampling rate.

Software

The previous section provided a brief overview of the hardware design issues associated with the AGC for a V.22 bis modem. DSP implementation issues are the focus throughout the rest of this report. All values are represented in decimal format unless otherwise noted. Data values in a digital system are not integers, but they must be manipulated as such on an integer processor. Appendix B provides an overview of fractional number representation on a two's-complement fixed-point device.

We choose to represent the signal within the AGC loop in S4.11 format. Recall that the $\{I_n,Q_n\}$ sequence can assume any value from the sequence $\{\pm 1, \pm 3\}$. This means that the sequence is bound in the ±3 range. We use three bits to represent the values in the given range, while the rest of the 12 bits can be treated as the fractional part that accommodates noise. Allocating an extra bit to the $\{I_n,Q_n\}$ sequence fully represents the RMS signal and allows for some gain hit.

For QAM signals, experimentation has shown that the ratio of peak signal to RMS signal is approximately 3 to 1. The maximum peak signal that can be represented using S4.11 notation is 16 (see Appendix B); therefore, 16 represents the peak value a QAM signal can attain using this notation. The RMS_{max} is hence 5.33, which corresponds to approximately 14.5 dB (20 log 5.33). We design the system to work with a 10-dB gain hit. It follows that the AGC should maintain the signal level at approximately 4.5 dB or 1.69 RMS level.The constant level of 1.69 RMS represented in S4.11 format is 3461.12. The AGC loop maintains an average squared level of 2.86, or $(1.69)^2$, per sample. Therefore, to determine the average baud energy, the sample energy must be multiplied by 16. The resultant value (45.8) is represented in S10.5 format (corresponding to 1466 (05BAh) in S15.0 format), the actual value used in the implementation (see Appendix E for the code listing).

As shown in the previous section, the reference energy for a V.22 bis modem is 5. This corresponds to the energy level of the constellation points (1,3) and (3,1), shown in Figure 3. Hence it is possible to map the average baud energy of 5 into 45.8. Extending the mapping to the other energy levels results in the following:

Average Baud Energy	S10.5 Format	S15.0 Format
1 maps into	9.16	292
5 maps into	45.8	1466
9 maps into	82.4	2632

Error Windowing and Weighting

In the previous section, the need was established for an energy window around the nominal baud energy level to compensate for the effects of intersymbol interference. The AGC is not designed to, and should not be expected to, compensate for ISI. The equalizer in the modem receiver is designed for this purpose [6]. Experimental window values of 1320 and 950 were chosen for QAM and DPSK modes of operation, respectively.

The windowed error signal must be weighted appropriately to provide an approximate one-to-one relationship between the positive and negative energy errors. In Figure 3, the disparity between the positive and negative errors can be observed. Assume that the received points are (6,6) and (0.5,0.5). The QAM signal energy can be calculated as

$$E_{QAM} = 1/2 \; (I_n^{\,2} + Q_n^{\,2}) \qquad (11)$$

Therefore, the energy values of the received points are 36 and 0.25, respectively. When these energy values are represented in S10.5 (10552 and 73, respectively) and the deviation from the nominal energy level of 1466 is calculated, full scale error values of 9086 and –1393, respectively, are obtained. This indicates a nonlinear relationship between the received constellation points signal energy with respect to the nominal baud energy level. It is important to determine the weighting factor to provide a parity between positive and negative errors while the AGC operates in the steady state or tracking mode. Appendix D provides a Fortran program to determine the best value for the expansion ratio of negative and positive energy values.

AGC State Detector

The AGC always operates in one of two modes:

- Slew – (fast tracking mode) AGC uses a large step size to track the signal.
- Tracking – AGC adjusts the signal level by adjusting the gain factor via an exponential integrator loop.

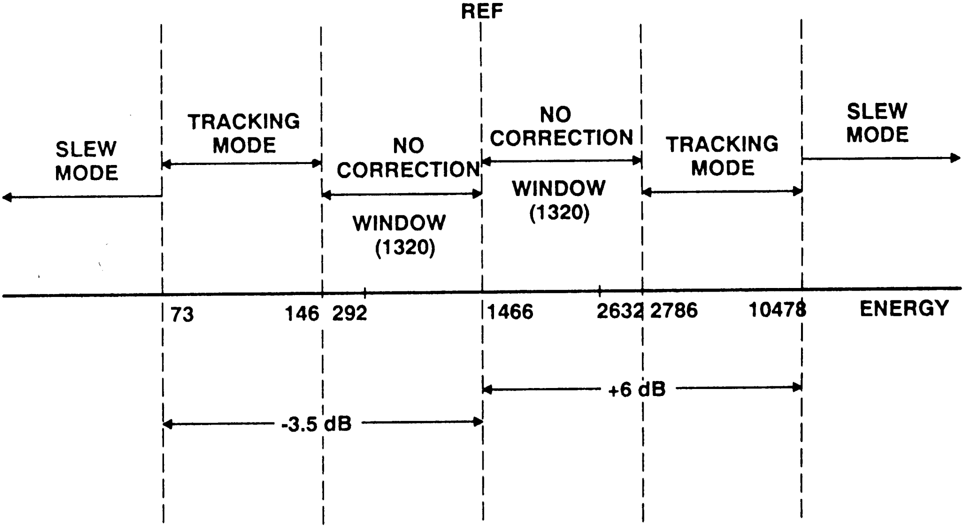
It is important to design the AGC to ignore relatively small gain changes on the telephone line. Otherwise, the AGC loop responds to the smallest variation in the signal level by switching to the slew mode. In this application, the AGC is designed to simply track the incoming signal when the received signal level varies by not more than ± 6 dB from the window values. These levels are calculated as follows:

$$10 \log(x/2632) = +6\ dB \rightarrow x = 10478 \tag{12}$$

$$10 \log(x/292) = -6\ dB \rightarrow x = 73 \tag{13}$$

As long as the incoming signal stays within these boundaries, the AGC simply adjusts the gain factor; otherwise, it will switch to the slew mode. Once the AGC determines that the error signal is within the tracking mode boundary, it switches back to the slow tracking mode as shown in 5.

Figure 5. AGC Operating Modes



Appendix C provides a FORTRAN program that determines the best weighting factor for a given QAM signal range. A weighting factor of 2 provided the approximate one-to-one relationship. Since DPSK signals do not have amplitude variations, a value of 1 was chosen for the weighting factor when the modem operates in the V.22/Bell 212A mode.

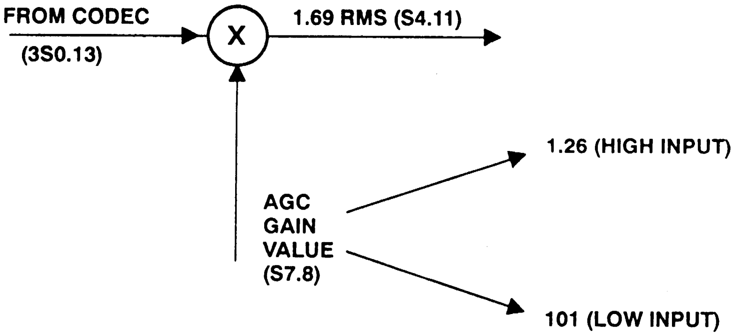
An upper and lower boundary for the AGC gain value must be determined. The V.22bis standard[3] requires the modem to operate at a signal level of –43 dBm. Therefore, the AGC is designed to work from the 0-dBm signal level to –50 dBm.

The DSP2400 contains a DSP-activated 12-dB gain switch. Therefore, our design should really have to cover only the range of 0 dBm to –38 dBm levels. The maximum codec output value is 1FFEh (8190 decimal) because the codec output is converted from 8-bit log value to 13-bit two's-complement value. When this value is saved in a data memory location of the TMS320C17 DSP, the number is sign-extended and is represented in 3S0.13 format. The RMS_{max} is therefore 2730, which corresponds to a signal level of 0 dBm in our system. The minimum acceptable signal level from the codec corresponding to the –38-dBm level is computed as follows:

$$\begin{aligned} -38 &= 20 \log (RMS_{\min} / 2730) \\ RMS_{\min} &= 34.4 \end{aligned} \qquad (14)$$

Given the maximum and minimum codec output values and the constant RMS output, it follows that $\alpha_{min} = 1.26$ and $\alpha_{max} = 101$ as shown in Figure 6.

Figure 6. AGC Gain Value Computation



The gain value requires 7 bits to represent; therefore, the S7.8 format is used to represent the α values.

Exponential Integrator Loop

When the total baud energy stays within the window limits, the AGC is in the tracking mode and simply compensates for the changes in the signal levels by adjusting the gain factor appropriately. The gain factor is computed and updated via an exponential integrator loop. The exponential integrator loop implements the following function:

$$\alpha_{n+1} = \alpha_n \times (1 - \mathrm{K}e) \qquad (15)$$

where the constant K determines the speed of convergence of the AGC closed loop. In our implementation, K is set to 1/2. This value corresponds to step sizes of ± 6 dB when the AGC is in the slew mode. The error signal is in S0.15 format while α_n is in S7.8 format with the multiplication result in 2S7.23 format. When the upper half of the accumulator (ACCH) is saved with a left shift, the result is in S7.8 format. A further multiplication by 0.5 is necessary before carrying out the subtraction operation. Note that a divide by 2 is equivalent to a right shift, which cancels out the effect of the previous left shift. Therefore, saving ACCH with no shift accomplishes multiplication by K as shown in Appendix E.

The AGC is designed to declare carrier present when signal levels greater than –43 dBm appear at the input of the receiver. The response time for tone detection depends on the AGC design. The AGC uses a constant that is subtracted from a hysteresis counter, and presence of energy is declared when the counter underflows. It takes 9 bauds for the energy to be detected, corresponding to a response time of 15 ms.

Conclusion

This application report has presented design and implementation techniques for an all-digital automatic gain control. The AGC has been implemented on a TMS320C17 digital signal processor as part of a commercial modem product (DSP2400). The approach of using a programmable processor resulted in minimal hardware configuration with excellent performance. The DSP implementation allows you to fine tune the AGC for your particular modem design, regardless of the modulation technique used.

Acknowledgements

The author wishes to acknowledge the contribution of Technekron Communications Systems and George Troullinos of Texas Instruments. This report is based on their work.

References

1) *DSP2400 Modem User's Guide*, Texas Instruments, 1987.
2) Troullinos, G., et al, *Theory and Implementation of a Splitband Modem Using the TMS32010* (literature number SPRA013), Texas Instruments, 1986.
3) Recommendation V.22 bis, "2400 bits per second duplex modem using the frequency division technique standardized for use on the general switched telephone network and on point-to-point 2-wire leased telephone-type circuits," *CCITT Redbook*, Volume III, 1984.
4) Mercy, D.V., "A Review of Automatic Gain Control Theory," *The Radio and Electronic Engineer*, Volume 51, Number 11/12, November/December 1981.
5) Green, D.N., "Global Stability Analysis of Automatic Gain Control Circuits," *IEEE Transactions on Systems and Circuits*, Volume CAS-30, Number 2, February 1983.
6) Falconer, D.D., "Jointly Adaptive Equalization and Carrier Recovery in Two-Dimensional Digital Communication Systems," *Bell System Technical Journal*, Volume 55, Number 3, March 1976.
7) Antoniou, A., *Digital Filters: Analysis and Design*, John Wiley & Sons, 1986.
8) Lovrich, A., et al, "An All Digital Automatic Gain Control," *Proceedings of ICASSP 88*, Pages 1734–1737.

Appendix A

QAM Signal Energy

The general form of a QAM signal is written as

$$\begin{aligned} s(t) &= R(t)\cos\left[\omega_c t + \phi(t)\right] \\ &= I_n \cos \omega_c t + Q_n \sin \omega_c t \end{aligned} \tag{16}$$

The energy in a signal s(t) is defined as

$$E_{QAM} = \int_{-\infty}^{\infty} s^2(t)\, dt \tag{17}$$

Substituting (16) into (17) results in

$$\begin{aligned} E_{QAM} &= \int_0^T s^2(t)\, dt \\ &= \frac{1}{2\pi}\int_0^{2\pi} \left(I_n{}^2 \cos^2\omega_c t + Q_n{}^2\ \omega_c t + 2I_n Q_n \sin \omega_c t \cos \omega_c t \right) dt \\ &= \frac{1}{2\pi}\int_0^{2\pi} 1/2 \left[I_n{}^2 \left(1 + \cos 2\omega_c t \right) \right] dt + \frac{1}{2\pi}\int_0^{2\pi} 1/2 \left[Q_n\ 2^2 \left(1 - \cos 2\omega_c t \right) \right] \\ &\quad + \frac{1}{2\pi}\int_0^{2\pi} I_n Q_n \sin 2\omega_c t\, dt \end{aligned} \tag{18}$$

When the three terms in (18) are integrated, the sine and cosine terms drop out since the average energy of sinusoidal signals is zero. Therefore, (18) simplifies to

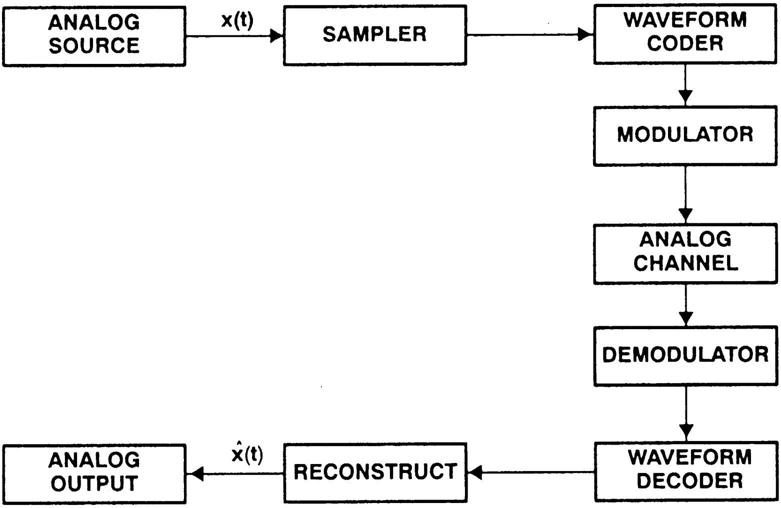
$$E_{QAM} = 1/2 \left(I_n{}^2 + Q_n{}^2 \right) \tag{19}$$

Appendix B

Fractional Number Representation Overview

A typical digital communication system is shown in Figure 7. Two blocks (marked as waveform coder and waveform decoder) are of interest. These blocks are collectively referred to as a codec, especially when both coder and decoder are implemented on a single device. An example is the TCM29C13 PCM codec, which consists of an amplitude quantizer and binary codeword generator.

Figure 7. A Typical Communication Channel



The quantized data represent instantaneous values of a continuous-time signal in digital form. On the TMS320C17, these data values are represented in two's-complement arithmetic[7]. The binary representation of a two's-complement value is as follows:

$$A = a_0 + \sum_{i=1}^{15} a_i \, 2^{-i} \tag{20}$$

Consider that the incoming samples are coming from a 16-bit linear ADC. The data coming out of the ADC consist of a sign bit at the most significant location, followed by the binary point. This information can be represented in Q15 format or, alternately, S0.15 format. This translates into the following upperbound and lowerbound limits with increments of 2^{-15} (0.00003051):

$$\begin{gathered} (2^{15} - 1) / 2^{15} = 0.99996948 \\ -2^{15} / 2^{15} = -1 \end{gathered} \tag{21}$$

If two Q15 (S0.15) numbers are multiplied, the result is a number in Q30 (SS0.30) format. When the Q30 number resides in the 32-bit accumulator of the TMS320C17, the binary point follows the second most-significant bit. Assuming that the output of the encoder section is also Q15 format, the Q30 number must be adjusted by left-shifting by one while maintaining the most-significant 16 bits of the result. This is accomplished with a sach y,1. This instruction shifts the Q30 (SS0.30) number to the left by one and, following the shift, stores the upper 16 bits of the accumulator. The y value is in Q15 (S0.15) format.

The S notation is used consistently throughout this application report. The following table should assist you with the conversion between Q notations, S notations, and equivalent decimal representations.

Table 1. S Notation, Q Notation, and Decimal Conversion Information

Q Notation	S Notation	Decimal Equivalent
Q15	S0.15	−1 N≤ 0.9999695
Q14	S1.14	−2 N≤ 1.9999390
Q13	S2.13	−4 N≤ 3.9998779
Q12	S3.12	−8 N≤ 7.9997559
Q11	S4.11	−16 N≤ 15.9995117
Q10	S5.10	−32 N≤ 31.9990234
Q9	S6.9	−64 N≤ 63.9980469
Q8	S7.8	−128 N≤ 127.9960938
Q7	S8.7	−256 N≤ 255.9921875
Q6	S9.6	−512 N≤ 511.9804375
Q5	S10.5	−1024 N≤ 1023.96875
Q4	S11.4	−2048 N≤ 2047.9375
Q3	S12.3	−4096 N≤ 4096.875
Q2	S13.2	−8192 N≤ 8191.75
Q1	S14.1	−16384 N≤ 16383.5
Q0	S15.0	−32768 N≤ 32767

Appendix C

The following is a Fortran program listing that creates a table of AGC gain values and its relation to the input signal strength. The table also includes the corresponding peak input signal level and its RMS equivalent.

dbm.for

```
c  This program determines the values of dBm signal levels in a QAM system:
c  Double Precision Peak, RMS, Alpha, dBm.
c
c     - Peak is the peak signal input to the receiver side.
c
c     - RMS is the RMS signal input to the receiver, where in a QAM system is
c       equal to on third of the peak value.
c
c     - Alpha is the gain value.
c
c  In this AGC design, the max codec input is actually +2.2 dBm (equivalent to
c  8190 peak.

        open      (1,file = 'dbm.dat',status = 'new')

        write     (1,8)
   8    format    (5x,' dbm',15x,'peak',16x,'rms',17x,'alpha')
        write     (1,9)
   9    format    (5x,' ===',15x,'====',16x,'===',17x,'=====')
                  dbm = -27.1
                  max = 2.
 999    if        (dbm .gt. max) goto 1000
                  dbm = dbm + 0.1
                  peak = 8190. * (10 ** ((dbm - max) / 20.))
                  rms = peak / 3.
                  alpha = 3461.12 / rms
        write     (1,10) dbm,peak,rms,alpha
  10    format    (f10.1,10x,f10.4,10x,f10.4,10x,f10.2)
        goto      999
c
1000    stop
        end
```

```
 dBm     Peak        RMS         Alpha

-27.0    290.5922     96.8641    35.73
-26.9    293.9571     97.9857    35.32
-26.8    297.3609     99.1203    34.92
-26.7    300.8042    100.2681    34.52
-26.6    304.2873    101.4291    34.12
-26.5    307.8108    102.6036    33.73
-26.4    311.3751    103.7917    33.35
-26.3    314.9807    104.9936    32.97
-26.2    318.6280    106.2093    32.59
-26.1    322.3175    107.4392    32.21
-26.0    326.0498    108.6833    31.85
-25.9    329.8252    109.9417    31.48
-25.8    333.6444    111.2148    31.12
-25.7    337.5079    112.5026    30.76
-25.6    341.4160    113.8053    30.41
-25.5    345.3694    115.1231    30.06
-25.4    349.3686    116.4562    29.72
-25.3    353.4141    117.8047    29.38
-25.2    357.5065    119.1688    29.04
-25.1    361.6462    120.5487    28.71
-25.0    365.8338    121.9446    28.38
-24.9    370.0700    123.3567    28.06
-24.8    374.3552    124.7851    27.74
-24.7    378.6900    126.2300    27.42
-24.6    383.0751    127.6917    27.11
-24.5    387.5109    129.1703    26.80
-24.4    391.9980    130.6660    26.49
-24.3    396.5372    132.1791    26.19
-24.2    401.1288    133.7096    25.89
-24.1    405.7737    135.2579    25.59
-24.0    410.4723    136.8241    25.30
-23.9    415.2254    138.4085    25.01
-23.8    420.0335    140.0112    24.72
-23.7    424.8972    141.6324    24.44
-23.6    429.8173    143.2724    24.16
-23.5    434.7943    144.9314    23.88
-23.4    439.8290    146.6097    23.61
-23.3    444.9220    148.3073    23.34
-23.2    450.0740    150.0247    23.07
-23.1    455.2856    151.7619    22.81
-23.0    460.5575    153.5192    22.55
-22.9    465.8905    155.2968    22.29
-22.8    471.2853    157.0951    22.03
-22.7    476.7425    158.9142    21.78
-22.6    482.2629    160.7543    21.53
-22.5    487.8473    162.6158    21.28
-22.4    493.4963    164.4988    21.04
-22.3    499.2107    166.4036    20.80
-22.2    504.9913    168.3304    20.56
-22.1    510.8388    170.2796    20.33
-22.0    516.7540    172.2513    20.09
-21.9    522.7378    174.2459    19.86
```

-21.8	528.7908	176.2636	19.64
-21.7	534.9139	178.3046	19.41
-21.6	541.1079	180.3693	19.19
-21.5	547.3736	182.4579	18.97
-21.4	553.7119	184.5706	18.75
-21.3	560.1236	186.7079	18.54
-21.2	566.6095	188.8698	18.33
-21.1	573.1706	191.0569	18.12
-21.0	579.8076	193.2692	17.91
-20.9	586.5214	195.5071	17.70
-20.8	593.3130	197.7710	17.50
-20.7	600.1833	200.0611	17.30
-20.6	607.1331	202.3777	17.10
-20.5	614.1633	204.7211	16.91
-20.4	621.2750	207.0917	16.71
-20.3	628.4690	209.4897	16.52
-20.2	635.7464	211.9155	16.33
-20.1	643.1080	214.3693	16.15
-20.0	650.5548	216.8516	15.96
-19.9	658.0879	219.3626	15.78
-19.8	665.7082	221.9027	15.60
-19.7	673.4167	224.4722	15.42
-19.6	681.2145	227.0715	15.24
-19.5	689.1026	229.7009	15.07
-19.4	697.0820	232.3607	14.90
-19.3	705.1539	235.0513	14.72
-19.2	713.3192	237.7731	14.56
-19.1	721.5790	240.5263	14.39
-19.0	729.9345	243.3115	14.23
-18.9	738.3867	246.1289	14.06
-18.8	746.9369	248.9790	13.90
-18.7	755.5860	251.8620	13.74
-18.6	764.3352	254.7784	13.58
-18.5	773.1858	257.7286	13.43
-18.4	782.1389	260.7130	13.28
-18.3	791.1956	263.7319	13.12
-18.2	800.3573	266.7858	12.97
-18.1	809.6250	269.8750	12.82
-18.0	819.0000	273.0000	12.68
-17.9	828.4835	276.1612	12.53
-17.8	838.0769	279.3590	12.39
-17.7	847.7814	282.5938	12.25
-17.6	857.5983	285.8661	12.11
-17.5	867.5288	289.1763	11.97
-17.4	877.5743	292.5248	11.83
-17.3	887.7361	295.9120	11.70
-17.2	898.0156	299.3385	11.56
-17.1	908.4141	302.8047	11.43
-17.0	918.9331	306.3110	11.30
-16.9	929.5738	309.8579	11.17
-16.8	940.3378	313.4459	11.04
-16.7	951.2264	317.0755	10.92
-16.6	962.2411	320.7470	10.79
-16.5	973.3833	324.4611	10.67
-16.4	984.6545	328.2182	10.55
-16.3	996.0563	332.0188	10.42
-16.2	1007.5901	335.8634	10.31
-16.1	1019.2574	339.7525	10.19
-16.0	1031.0599	343.6866	10.07
-15.9	1042.9990	347.6663	9.96
-15.8	1055.0764	351.6921	9.84
-15.7	1067.2936	355.7645	9.73
-15.6	1079.6522	359.8841	9.62
-15.5	1092.1540	364.0513	9.51
-15.4	1104.8006	368.2669	9.40
-15.3	1117.5936	372.5312	9.29
-15.2	1130.5347	376.8449	9.18
-15.1	1143.6257	381.2086	9.08
-15.0	1156.8682	385.6227	8.98
-14.9	1170.2641	390.0880	8.87
-14.8	1183.8151	394.6050	8.77
-14.7	1197.5231	399.1744	8.67
-14.6	1211.3897	403.7966	8.57
-14.5	1225.4170	408.4723	8.47
-14.4	1239.6066	413.2022	8.38
-14.3	1253.9606	417.9869	8.28
-14.2	1268.4808	422.8269	8.19
-14.1	1283.1691	427.7230	8.09
-14.0	1298.0275	432.6758	8.00
-13.9	1313.0579	437.6860	7.91
-13.8	1328.2624	442.7541	7.82
-13.7	1343.6430	447.8810	7.73
-13.6	1359.2016	453.0672	7.64
-13.5	1374.9405	458.3135	7.55
-13.4	1390.8615	463.6205	7.47
-13.3	1406.9669	468.9890	7.38
-13.2	1423.2589	474.4196	7.30
-13.1	1439.7394	479.9131	7.21
-13.0	1456.4108	485.4703	7.13
-12.9	1473.2753	491.0918	7.05
-12.8	1490.3350	496.7783	6.97
-12.7	1507.5922	502.5307	6.89
-12.6	1525.0493	508.3498	6.81
-12.5	1542.7086	514.2362	6.73
-12.4	1560.5723	520.1908	6.65
-12.3	1578.6429	526.2143	6.58
-12.2	1596.9227	532.3076	6.50
-12.1	1615.4142	538.4714	6.43
-12.0	1634.1198	544.7066	6.35
-11.9	1653.0420	551.0140	6.28
-11.8	1672.1833	557.3944	6.21
-11.7	1691.5463	563.8488	6.14
-11.6	1711.1335	570.3778	6.07
-11.5	1730.9475	576.9825	6.00
-11.4	1750.9909	583.6636	5.93
-11.3	1771.2664	590.4221	5.86
-11.2	1791.7767	597.2589	5.80
-11.1	1812.5245	604.1748	5.73

-11.0	1833.5126	611.1709	5.66
-10.9	1854.7437	618.2479	5.60
-10.8	1876.2206	625.4069	5.53
-10.7	1897.9462	632.6487	5.47
-10.6	1919.9234	639.9745	5.41
-10.5	1942.1550	647.3850	5.35
-10.4	1964.6441	654.8814	5.29
-10.3	1987.3936	662.4645	5.22
-10.2	2010.4066	670.1355	5.16
-10.1	2033.6860	677.8953	5.11
-10.0	2057.2350	685.7450	5.05
-9.9	2081.0566	693.6855	4.99
-9.8	2105.1541	701.7180	4.93
-9.7	2129.5307	709.8436	4.88
-9.6	2154.1895	718.0632	4.82
-9.5	2179.1338	726.3779	4.76
-9.4	2204.3670	734.7890	4.71
-9.3	2229.8923	743.2974	4.66
-9.2	2255.7133	751.9044	4.60
-9.1	2281.8332	760.6111	4.55
-9.0	2308.2556	769.4185	4.50
-8.9	2334.9839	778.3280	4.45
-8.8	2362.0218	787.3406	4.40
-8.7	2389.3727	796.4576	4.35
-8.6	2417.0403	805.6801	4.30
-8.5	2445.0283	815.0094	4.25
-8.4	2473.3404	824.4468	4.20
-8.3	2501.9804	833.9935	4.15
-8.2	2530.9519	843.6506	4.10
-8.1	2560.2590	853.4197	4.06
-8.0	2589.9054	863.3018	4.01
-7.9	2619.8951	873.2984	3.96
-7.8	2650.2320	883.4107	3.92
-7.7	2680.9203	893.6401	3.87
-7.6	2711.9639	903.9880	3.83
-7.5	2743.3669	914.4556	3.78
-7.4	2775.1336	925.0445	3.74
-7.3	2807.2681	935.7560	3.70
-7.2	2839.7748	946.5916	3.66
-7.1	2872.6578	957.5526	3.61
-7.0	2905.9216	968.6405	3.57
-6.9	2939.5706	979.8569	3.53
-6.8	2973.6092	991.2031	3.49
-6.7	3008.0420	1002.6807	3.45
-6.6	3042.8735	1014.2912	3.41
-6.5	3078.1083	1026.0361	3.37
-6.4	3113.7511	1037.9170	3.33
-6.3	3149.8067	1049.9356	3.30
-6.2	3186.2797	1062.0932	3.26
-6.1	3260.4977	1086.8326	3.18
-5.9	3298.2525	1099.4175	3.15
-5.8	3336.4445	1112.1482	3.11
-5.7	3375.0787	1125.0262	3.08
-5.6	3414.1602	1138.0534	3.04
-5.5	3453.6943	1151.2314	3.01
-5.4	3493.6862	1164.5621	2.97
-5.3	3534.1412	1178.0471	2.94
-5.2	3575.0646	1191.6882	2.90
-5.1	3616.4619	1205.4873	2.87
-5.0	3658.3386	1219.4462	2.84
-4.9	3700.7002	1233.5667	2.81
-4.8	3743.5523	1247.8508	2.77
-4.7	3786.9005	1262.3002	2.74
-4.6	3830.7508	1276.9169	2.71
-4.5	3875.1088	1291.7029	2.68
-4.4	3919.9804	1306.6601	2.65
-4.3	3965.3717	1321.7906	2.62
-4.2	4011.2885	1337.0962	2.59
-4.1	4057.7370	1352.5790	2.56
-4.0	4104.7234	1368.2411	2.53
-3.9	4152.2539	1384.0846	2.50
-3.8	4200.3347	1400.1116	2.47
-3.7	4248.9723	1416.3241	2.44
-3.6	4298.1731	1432.7244	2.42
-3.5	4347.9436	1449.3145	2.39
-3.4	4398.2904	1466.0968	2.36
-3.3	4449.2202	1483.0734	2.33
-3.2	4500.7397	1500.2466	2.31
-3.1	4552.8559	1517.6186	2.28
-3.0	4605.5754	1535.1918	2.25
-2.9	4658.9055	1552.9685	2.23
-2.8	4712.8531	1570.9510	2.20
-2.7	4767.4253	1589.1418	2.18
-2.6	4822.6295	1607.5432	2.15
-2.5	4878.4729	1626.1576	2.13
-2.4	4934.9630	1644.9877	2.10
-2.3	4992.1072	1664.0357	2.08
-2.2	5049.9131	1683.3044	2.06
-2.1	5108.3883	1702.7961	2.03
-2.0	5167.5406	1722.5135	2.01
-1.9	5227.3779	1742.4593	1.99
-1.8	5287.9081	1762.6360	1.96
-1.7	5349.1392	1783.0464	1.94
-1.6	5411.0793	1803.6931	1.92
-1.5	5473.7367	1824.5789	1.90
-1.4	5537.1196	1845.7065	1.88
-1.3	5601.2364	1867.0788	1.85
-1.2	5666.0957	1888.6986	1.83
-1.1	5731.7060	1910.5687	1.81
-1.0	5798.0760	1932.6920	1.79
-0.9	5865.2145	1955.0715	1.77
-0.8	5933.1305	1977.7102	1.75
-0.7	6001.8329	2000.6110	1.73
-0.6	6071.3309	2023.7770	1.71
-0.5	6141.6336	2047.2112	1.69
-0.4	6212.7504	2070.9168	1.67
-0.3	6284.6906	2094.8969	1.65
-0.2	6357.4639	2119.1546	1.63

-0.1	6431.0799	2143.6933	1.61
0.0	6505.5483	2168.5161	1.60
0.1	6580.8790	2193.6263	1.58
0.2	6657.0819	2219.0273	1.56
0.3	6734.1673	2244.7224	1.54
0.4	6812.1453	2270.7151	1.52
0.5	6891.0262	2297.0087	1.51
0.6	6970.8206	2323.6069	1.49
0.7	7051.5389	2350.5130	1.47
0.8	7133.1918	2377.7306	1.46
0.9	7215.7903	2405.2634	1.44
1.0	7299.3452	2433.1150	1.42
1.1	7383.8676	2461.2892	1.41
1.2	7469.3688	2489.7896	1.39
1.3	7555.8600	2518.6200	1.37
1.4	7643.3528	2547.7843	1.36
1.5	7731.8586	2577.2862	1.34
1.6	7821.3893	2607.1298	1.33
1.7	7911.9567	2637.3189	1.31
1.8	8003.5729	2667.8576	1.30
1.9	8096.2499	2698.7500	1.28
2.0	8190.0000	2730.0000	1.27

Appendix D

Appendix D provides a Fortran program that calculates an optimal value for the expansion ratio of negative and positive energy values, subject to some constraints (maximum signal levels). The program searches expansion ratios with their corresponding error values up to a maximum value defined by the user. The value that produces the least error is chosen as the optimal value. In this implementation, the tracking mode window is 6 dB for positive errors and at least 3.5 dBs wide for negative errors. The program, however, calculates the expansion window in 6-dB range. Error values are calculated using no-worse windows data. The index value for positive and negative errors correspond to the actual signal level in tenths of dBs.

```
c
c
c    Program to determine the best value for the expansion ratio of negative
c    energy values and that of positive ones.
c
          double precision neg(60), poserr(60), negerr(60)
          double precision sigma(60), maxerr, minerr, bingo
          double precision total(400)
          open  (1,file = 'n1.dat',status = 'new')
c
c    clear all the total values
c
            do 100  n = 1,400
  100       total (n) = 0.

            write(*,1)
    1       format(1x, 'enter positive dbm level')
            read (*,*) dbmpos
            write(*,2)
    2       format(1x, 'enter negative dbm level')
            read (*,*) dbmneg
            write(*,8)
    8       format(1x, 'enter maximum value for N')
            read (*,*) nn
c
c    determine positive errors
c
c    Since the AGC operates in the tracking mode close to the  boundary, more
c    weight must be given to these regions.
c
c    0.0 to 1.0 dB       10 pts
c    1.1 to 2.0 dB       10 pts
c    2.1 to 3.0 dB  5 pts
c    3.1 to 4.0 dB  1 pts
c    4.1 to 5.0 dB  1 pts
c    5.1 to 6.0 dB  1 pts
c
            do 200 i = 1,20
  200       poserr (i) = 2632. * (( 10. ** ( float(i) / 100.)) - 1. )
c 200       write    (1,5) i,poserr(i)
c
            do 201 i = 22,30,2
  201       poserr  (i) = 2632. * (( 10. ** ( float(i) / 100.)) - 1. )
c 201       write (1,5) i,poserr(i)
c
            do 201 i = 22,30,2
  202       poserr (i) = 2632. * (( 10. ** ( float(i) / 100.)) - 1. )
c 202       write(1,5) i,poserr(i)
c
c    determine negative errors
c
c    We do the same thing with the negative errors.
c
            do 300 k = 1,20
  300       negerr (k) = 292.*(1. - (10. ** (float(-k) /100.)))
c 300       write(1,9) k,negerr(k)
    9       format (1x, 'negative error(',i2,') = ',1x,f20.4)
c
            do 301 k = 22,30,2
  301       negerr (k) = 292.*(1. - (10. ** (float(-k) /100.)))
c 301       write(1,9) k,negerr(k)
c
            do 302 k = 40,60,10
  302       negerr (k) = 292.*(1. - (10. ** (float(-k) /100.)))
c 302       write(1,9) k,negerr(k)
c
c    Assuming that the mapping is actually linear, then the following criteria
c    is used to determine the optimum value for N.
c
c
c    total(n) = Sigma [ e  - e  * n ]
c        k   +k -k
c
            do 400 n = 1,nn
            do 400 k = 1,60
            sigma(k) = poserr(k) - float(n) * negerr(k)
  400       total(n) = total(n) + sigma(k)
c
c    Now it is time to determine the minimum value of the error.
c
            do 500 n = 1,nn
            if ( bingo .lt. 0.) goto 504
            if ( total(n) .le. total(n+1)) goto 501
            bingo = total(n+1)
            itr = n+1
            goto 502
  501       bingo = total(n)
            itr = n
  502       if ( n+1 .gt. nn) goto 503
  500       continue
  503       itr = n
            bingo = total(n)
            goto 510
  504       itr = n-1
            bingo = total(n-1)
c
c    Calculate maximum and minimum energy levels
c
            do 600 i = 1,60
  600       write (1,5) i, poserr(i)
    5       format(1x, 'positive error(',i2,') = ',1x,f20.4)
c
            do 601 k = 1,60
            neg(k) = itr * negerr(k)
  601       write (1,6) k, negerr(k), neg(k)
    6       format(1x, 'negative error(',i2,') = ',1x,f20.4,
           +' equivalent to',f20.4)
            maxerr = 2632. * (( 10. ** ( dbmpos / 10. )) - 1.)
```

```
            minerr = 292. * float(itr) * ( 1. - ( 10. ** (dbmneg / 10. )))
            write  (1,3) maxerr
    3       format (1x, ' Maximum energy level is ',f20.4)
            write  (1,4) minerr
    4       format (1x, ' Minimum energy level is ',f20.4)
c
c    Output the N value
c
  510       write(1,7) itr, bingo
    7       format(1x,'N =',i3,' with the corresponding error of ',f20.4)
c
            end
```

Appendix E

```
;*****************************************************************************
;
;   agc.asm
;
;*****************************************************************************
;
;   front end agc function.
;
;
;*****************************************************************************
;
;   the average signal squared is computed by the main program and stored in
;   avesqr, which is cleared by this routine after using it. the routine uses
;   a window whose width depends on the modulation (1200,2400) and a error
;   weighting, which also depends on that rate. we first set those values:
;
agc:
        sovm
        lac       gn            ; check for afe switching
        bnz       switch
;
;   check if 2400 and change those values
;
        lack      3
        and       status        ; if status bits 0 and 1
2 = 2400
        sub       one,1
        blez      agc0          ; if 2 , do not modify tmp0 and tmp1
;
;   for 2400 , 2 - tmp0 and 1320 - tmp1
;
        lack      2             ; it is 2400
        sacl      tmp0
        lt        one
        mpyk      1320
        pac
        sacl      tmp1
        b         agc1
;
agc0:
        lack      1             ; it is 1200
        sacl      tmp0          ; weighting factor - tmp0
        lt        one
        mpyk      950
        pac
        sacl      tmp1          ; window - tmp1
```

```
;
;   now subtract reference from baud energy to get error. the baud energy is
;   in s10.5 format. the AGC maintains that level at 2.86x16 = 46.7  (h'5b6
;   in s10.5). the agcref is therefore h'5b6
;
agc1:
        lac       avesqr
        bgez      cont1         ; for negative energy
        lac       one,15        ; set to max positive
        sub       one           ; energy level - forced
        slew      mode
ont1:
        lt        one
        mpyk      agcref        ; agcref = h'5b6
        spac                    ;avesqr - agcref - acc
;
;   compare the error to window (tmp1).
;   if error  window = error - window - error
;   if -window  error  window = 0 - error
;   if error  -window  = tmp0 x (error + window) - error
;   if the average baud energy is a, the peak baud energy for QAM signals is
;   1.8 a and the minimum is 0.2 a.
;   the window is therefore chosen to be 0.8 a in either direction. with
;   agcref = h'5b6 the window is h'492
;   for dpsk signals the variations in baud energy are entirely due to isi
;   and distortion and therfore the window is much smaller (h'a).
;   first check if error  window
;
        sub       tmp1
        sacl      tmp3          ; error - window - tmp3
        bgez      agc2
;
;   error  window = check if error  -window. in which case zero the error.
;   first zero the error (i.e. assume error  -window) and modify if wrong
;   assumption.
;
        lark      ar1,0
        sar       ar1,tmp3      ; assume error  -window
;
;   check asumption
;
        add       tmp1,1        ; error + window - acc
        bgez      agc2          ; assumption is right
;
;   error  -window = tmp0x(error+window) - tmp3
;
        sacl      tmp3
        lt        tmp3
        mpy       tmp0
        pac
        sacl      tmp3
```

```
;
;    at this point, the weighted windowed error is contained in tmp3. we
;    consider it an s.15 number and use it to update the agc gain alpha. first
;    we determine whether to slew or not. if error is larger than 1EA6h or
;    smaller than F5E7h, go into slewing mode by setting error to 7FFFh or
;    8000h respectively. otherwise leave it unchanged.
;
agc2:
        lac     tmp3
        ldpk    1
        sub     possm
        blez    agc3            ; do not slew
;
        ldpk    0               ; enter slew mode
        lac     one,15
        sub     one
        sacl    tmp3            ; tmp3  7FFFh
        b       agc4
;
agc3:
        add     possm           ; acc  tmp3
        add     negsm
        ldpk    0
        bgez    agc4            ; do not slew
;
        lac     one,15          ; enter slew mode
        add     one
        sacl    tmp3            ; tmp3  8000h
;
;    the following lines update the gain alpha using an exponential integrator
;    alpha = alpha(1-kxerror)   (error = tmp3) where alpha is of format s7.8
;    and error is s0.15 and k = 0.5. alpha * error : s7.8 * s.15 = s7.24. by
;    keeping acch without left shift the multiplication by k is accomplished.
;    alpha is upperbounded to 35.73 in s7.8
;
agc4:
        lack    maxalp
        tblr    tmp0
        zalh    alpha
        lt      tmp3            ; error - t
        mpy     alpha
        spac                    ; alpha (1 - 0.5*error) - acc
        sach    alpha
;
;    check if alpha  max alpha
;
        subh    tmp0
        blez    agc5
;
        lac     tmp0
        sacl    alpha
```

```
;
;    zero baud energy register
;
agc5:
        zac
        sacl    avesqr
;
;****************************************************************************
;
;    energy detect loop
;
;****************************************************************************
;
;    start by reading in hysterisis counter increment constant
;
        lack    hysinc
        tblr    tmp5
;
;    check if afe gain is on or off
;
edt:
        lack    080h
        and     sturd
        bz      edt1
;
;    afe gain is on, check if energy detect is on (sturd[6] = 1)
;
        lac     one,6
        and     sturd
        bz      edt01           ; if zero = energy is not
;
;    detected = check if level is larger than -43.5 dbm. if sturd[6] is one,
;    check if level less than -48 dbm
;
        lack    thres1
        tblr    tmp0
        lac     tmp0
        sub     alpha
        blz     edt2            ; if  0 then no energy detect
;
;    check if afe gain stage should be bypassed
;
        lack    thres3
        tblr    tmp0
        lac     tmp0
        sub     alpha
        blz     edt3
;
;    is afe gain on?
;
        lack    080h
        and     sturd
        bz      edt3            ; if gain is off exit
```

```
;
;    bypass afe gain
;
        lack    07fh
        and     sturd
        sacl    sturd
        lack    04h
        sacl    gn
        ret
;
;    decrement hysterisis counter
;
edt2:
        bv      edt21           ; clear overflow bit
edt21:
        zalh    hyst            ; hysteresis counter
        subh    tmp5            ; tmp5 = 1927 = 32768/15
        sach    hyst
;
;    in case of overflow declare loss of energy detect
;
        bv      edt02
        ret
edt02:
        lack    0bfh
        and     sturd
        sacl    sturd
        ret
;
;    following lines are executed if afe gain is high but no energy detect.
;    check if alpha  21.28 (i.e. receive level -43.5 dbm) and increment
;    hysteresis counter if it is. otherwise, exit.
;
edt01:
        lack    thres2          ; 21.28 in s7.8
        tblr    tmp0
        lac     tmp0
        sub     alpha
        blz     edt3
;
;    alpha  21.28 = increment hyst. counter
;
        bv      edt011          ; clear overflow bit
edt011:
        zalh    hyst
        addh    tmp5            ; tmp5 contains inc. f0f
        sach    hyst
;
;    detect bit sturd[6] =1.
;
        bv      edt04           ; in case of overflow set energy
        ret
```

```
edt04:
        lac     one,6
        or      sturd
        sacl    sturd
        ret
;
;    if afe gain stage is bypassed, check level of alpha
;
edt1:
        lack    thres5
        tblr    tmp0
        lac     tmp0
        sub     alpha
        bgz     edt3
;
;    if alpha  thres5 (20.09 in s7.8) then turn afe gain status word bit on.
;
        lack    080h
        or      sturd
        sacl    sturd
        lack    014h
        sacl    gn
edt3:
        ret

;####################################################################
;
;    routine for switching the afe on/off
;
;####################################################################
;    zero baud energy register
;
switch:
        zac
        sacl    avesqr
;
        lack    010h            ; mask off unwanted bits
        and     gn
        bz      afeoff
;
;    check if the gain should be on
;
        lack    0fh             ; mask off the afe on bit
        and     gn
        sub     one             ; decrement the counter
        bz      swtch1
;
        sacl    gn              ; save gn value
        lack    010h            ; load the afe on bit
        or      gn
        sacl    gn              ; restore afe on bit
        ret
```

```
;
swtch1:
        sacl     gn             ; reset the gn value to zero
        lack     thres6         ; load the new alpha value
        tblr     alpha          ; reset alpha to 5.05
        ret
;
afeoff :
        lac      gn
        sub      one            ; decrement the counter
        bz       swtch2
;
        sacl     gn
        ret
;
swtch2:
        sacl     gn
        lack     thres4         ; load new alpha value
        tblr     alpha          ; reset alpha to 8.98 in s7.8
        ret
;
;################################################################################
*
;
;    RECEIVER PER SAMPLE PROCEDURE
;
rstskf:
        lac      rin            ; input 2's complement sample
;
;    high pass filter the incoming signal to remove the dc component.
;
        ldpk     1
        sacl     x1
        zals     stlsb
        addh     st
        sub      st,tau
        addh     x1
        sub      x1,tau-1
        subh     x2
        add      x2,tau-1
        sacl     stlsb
        sach     st
        dmov     x1
        ldpk     0
        sach     tmp1
```

```
;
;    multiply input sample by agc gain alpha. the output format is s4.11
;    requiring some manipulations
;
        lt       tmp1           ; multiply by agc word
        mpy      alpha
        pac
;
;    shift accumulator eight four times before storing
;
        sacl     tmp0
        sach     tmp1
        lac      tmp0,8
        sach     tmp0
        lac      one,8          ; mask off any sign extension
        sub      one            ; 00ff - acc
        and      tmp0
        sacl     tmp0
        lac      tmp1,8
        add      tmp0
        sacl     tmp1
;
; update the signal power estimate avesqr. avesqr = avesqr + (tmp1)2
; avesqr is zeroed by the agc routine once per baud.
;
        lac      tmp1,15
        sach     tmp0           ; tmp0 in s5.10
        lt       tmp0
        mpy      tmp1
        pac
        addh     avesqr         ; avesqr in s10.5
        sach avesqr
        .
        .
        .
;
;    end of receiver per sample task
;
```