Ptoc. ARRL 10th Computer Networking Conference, 1991

DESIGN AND IMPLEMENTATION OF CELP SPEECH PROCESSING SYSTEM USING TMS320C30

A. Langi, VE4ARM and W. Kinsner, VE4WK

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba, Canada R3T-2N2
E-mail: Kinsner@CCM.UManitoba.CA
Radio: VE4WK@VE4KV.MB.CAN.NA

Abstract

This paper presents a design and implementation of a signal processing system for telephone-quality speech transmission through a low bit-rate channel, such as packet-radio, at rates as low as 4.8 kbit/s in either real-time or off-line mode. The system compresses speech signal down to 4.8 kbit/s using the code-excited linear predictive (CELP) coding scheme adapted from the U.S. Federal Standard FS-1016. The system implements the CELP scheme using a floating-point digital signal processor (DSP) to perform real-time, interactive (full- or half-duplex) or fast off-line network-based applications. The system is implemented on a low-cost personal computer (PC) equipped with a TMS320C30 evaluation module (EVM).

1. INTRODUCTION

Current voice systems are no longer sufficient to meet demands on high quality voice communications over long distances. Present systems, operating on HF, VHF, and UHF, use analog signals for speech transmission [Youn90], [Bloo90]. However, the quality of analog voice deteriorates rapidly over long distances. In VHF/UHF transmission, the voice usually becomes too noisy after only a few repeaters. Transmission of digital signals is considered superior to its analog counterpart due to the ability to completely reconstruct the signal at each repeater. This ability results in a higher immunity to noise [Kins89], [Bloo90]. In addition, the digital transmission can use error protection schemes that increase the robustness of the transmitted signals.

The main problem of digital voice transmission through radio channel is the bandwidth limitation of the channel. Direct digitizing of the speech for telephone quality results in digital data with a rate of 64 kbit/s. This rate is too high for the channel, and, consequently, the system needs a speech compression scheme to bring down the bit rate.

A speech compression scheme called code-excited linear predictive (CELP) coding is very attractive for digital speech communication using radio channel [ScAt85]. The coding can compress speech down to a rate of 4.8 kbit/s. The resulting speech quality is better than any of speech compression schemes at the same bit rate. The U.S. Government has adopted the scheme as the proposed Federal Standard 1016 (FS-1016) for digital radio as well as secure voice communication at a rate of 4.8 kbit/s [CaTW90].

Another problem of digital radio communication is how to protect the compressed speech from the channel noise. Forward error correction (FEC) schemes can be used to protect the compressed speech. The FS-1016 CELP uses FEC schemes including a Hamming code. More sophisticated codes can be used such as in the proposed FS-1024 for land mobile radio system. However, the utilization of the codes results in higher bit rates, such as 8 kbit/s for the proposed FS-1024 system [RTWC89].

We have designed and implemented a CELP speech processing system that can be used for voice communication through packet radio. This system is capable of real-time, full-duplex speech communication at a rate of 4.8 kbit/s. It also can be used to support non real-time applications such as voice mail (store-and-forward mode). The system includes a forward error protection scheme to increase the robustness of the speech during the transmission. The system is implemented on a low-cost personal computer (PC-AT) equipped with a TMS320C30 evaluation module (EVM). The system maintains interoperability with other FS-1016 systems. It is also an extension of another implementation on the NEC 77320 EM [Laki90], [Laki91].

2. SYSTEM ARCHITECTURE

2.1 Basic Scheme

Figure 1 shows the system architecture corresponding to a CELP system [LaKi90]. The system consists of a *controller*, a *data processor, and data media. The* controller

manages the timing of the process and the flow of data. The data processor consists of functional blocks, transforming speech data into several required forms for performing speech compression, speech decompression, and error protection processes. The functional blocks in the upper section constitute the CELP ***transmitter,*** while the blocks in the lower section constitute the CELP ***receiver.*** Finally, the data media are storage device and packet radio channel. The speech data can be stored or transmitted on the data media.

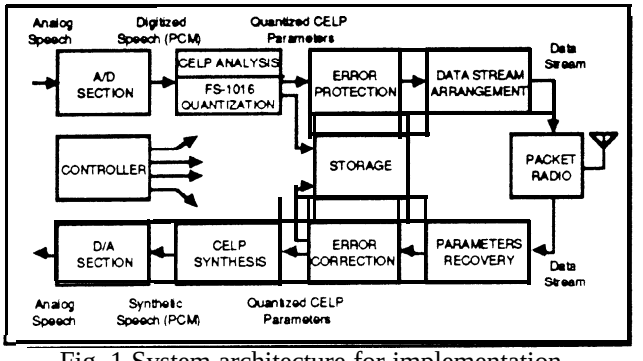

Fig. 1 System architecture for implementation.

The data processing of CELP transmitter begins with converting analog speech signal coming from a microphone and an amplifier, into a pulse code modulation (PCM) form by an analog-to-digital (A/D) converter. The A/D converter operates at an 8-kHz sampling rate and 14-bit resolution. The block called CELP analyzer compresses the digitized speech into speech data called CELP ***parameters. An*** FS-1016 quantizer enables low bit-rate representation of the CELP parameters and standardizes the bit definition of the parameters. A forward error protection scheme adds redundant bits for protecting the quantized CELP parameters from channel noise. The protected CELP parameter bits are reordered according to the FS-1016 protocol, and then applied to a packet radio. A user has an option to store the quantized CELP parameters on a storage device.

On the other hand, the CELP receiver reverses the data processing. The CELP parameters are recovered from a data stream coming from the packet radio. An error correction scheme detects and corrects data errors that are within its correction range. A CELP synthesizer uses the corrected CELP parameters to produces speech in a PCM form. A digital-to-analog (D/A) converter converts the PCM data into an analog form that can be used for speech listening using an amplifier and a speaker.

The main functional blocks of the data processing are the CELP analyzer and synthesizer. The two blocks, originally developed by Atal and Schroeder [ScAt85], enable the compression of PCM speech data into CELP parameters and the decompression of the CELP parameters into PCM speech data. Since the CELP analyzer utilizes a CELP synthesizer due to an analysis-by-synthesis approach, we first describe the CELP synthesizer. In addition, it is easier to understand the role of each CELP parameter using the description of CELP synthesizer.

2.2 CELP Synthesizer

Figure 2 shows a scheme in the CELP synthesizer. The synthesizer uses CELP parameters to produced PCM speech data. As shown in Fig. 2, the CELP parameters are linear predictor (LP) filter coefficients, adaptive code book (ACB) entry and gain factor, and stochastic code book (SCB) entry and gain factor. The parameters control three blocks, e.g., the LP filter, the ACB, and the SCB, to produce speech.

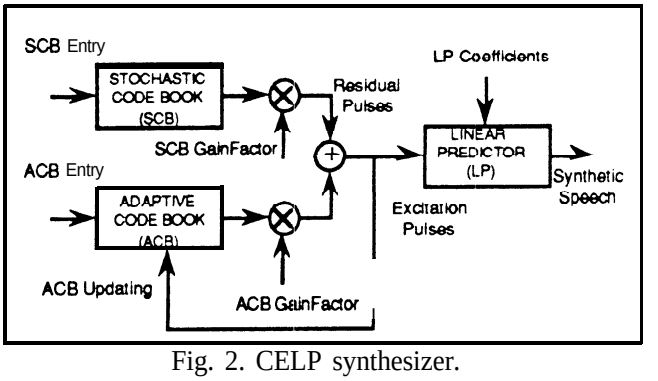

Fig. 2. CELP synthesizer.

The scheme is closely related to a human speech production system. The LP filter represents a human vocal tract. A change in the shape of the vocal tract during speech generation results in a change of frequency characteristics of the vocal tract. The LP filter (an all-pole adaptive digital filter) performs this frequency characteristic change by changing its coefficients. The speech is produced by applying signals called excitation pulses to the LP filter.

Speech has a regular fundamental frequency, called ***pitch frequency,*** which introduces ***tone to the:*** speech. The characteristic of the speech with regular pitch frequency is the speech waveform periodicity. The ACB, that is a code book containing many sequences of previous excitation pulses, can preserve the periodicity by supplying a proper

sequence of previous excitation pulses. The proper sequence is selected by an ACB entry and the amplitudes of the sequence elements are scaled by an ACB gain factor. It should be noted that the pulses stored in ACB are periodically updated with the latest excitation pulses.

The SCB is a code book storing many sequences of pulses, called residual pulses. The pulses are called residual because they can be seen as the remainder of speech samples after vocal tract and pitch frequency informations are taken away. In contrast with the ACB pulses, the residual pulses are fixed, normalized, and obtained experimentally. We use the SCB defined by the proposed FS-1016. An SCB entry selects a particular sequence, and an SCB gain factor scales the amplitude of the normalized pulses. The resulting residual pulses become the excitation pulses after being combined with the ACB pulses.

In summary, the CELP parameters control the CELP synthesizer to produce a particular speech waveform. Different parameters result in different characteristics of speech. Thus, a set of the CELP parameters can be seen as a speech representation. A speech compression is achieved because the CELP parameters require much fewer representation bits. Using the proposed FS-1016 CELP, we can represent 240 speech samples (PCM) in 144 bits of CELP parameters, as shown in Table 1.

Table 1. Bit allocation of proposed FS-1016 CELP system.

TYPE	ALLOCATION	TOTAL
Linear Predictor	3,4,4,4,4,3,3,3,3,3	34
Adaptive CB	Entry : 8+6+8+6 Gain: 5 x 4	48
Stochastic CB	Entry : 9+9+9+9 Gain: 5 x 4	56
Synchronization	1	1
Future Expansion	1	1
Error Correction	4	4

2.3 CELP Analyzer

Figure 3 shows a CELP analyzer. The analyzer compresses the incoming speech into CELP parameters. A block called LPC analysis obtains LP parameters from the input speech. The LPC analysis utilizes a scheme of coefficient estimation/tracking of an all-pole, time-varying filter [Pars86].

The SCB and ACB parameters are obtained using an analysis-by-synthesis scheme. An error minimization block applies every possible combination of the SCB and ACB parameters to a CELP synthesizer. For every set of parameters, there is a corresponding error signal. The set giving the minimal magnitude of error is chosen as the output of the analyzer. A perceptual weigthing (PW) filter weights the error signal prior to the magnitude measurement to compensate nonuniform perceptual effects of error on different speech frequencies.

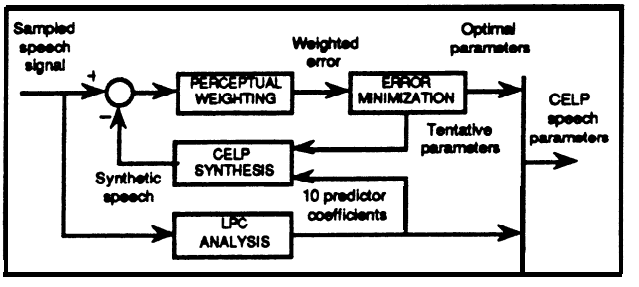

Fig. 3. CELP analyzer.

In practice, an analyzer's scheme may be different from the one described due to the implementation of fast algorithms, such as a *joint optimization* scheme used in our system [CaTW90]. However, every scheme is derived from the original scheme shown in Fig. 3.

3. SYSTEM IMPLEMENTATION

The system is implemented on a PC-AT equipped with a TMS320C30 EVM. The EVM contains a digital signal processor (DSP) and an analog interface. This platform allows us to implement all the blocks of Fig. 1 (except for the packet radio). The CELP algorithms are then coded to run on the EVM and the host PC-AT for performing functions described above.

3.1 Hardware Implementation

All functional blocks are organized for implementation according to a given hardware platform.

Hardware Platform

The host computer has an INTEL 80286 processor, 512 Kbytes memory (minimum), a hard disk, and a serial port (RS232C). The EVM is a half-size board that is installed in a PC-AT slot. The EVM contains a powerful floating point, 32-bit DSP from Texas Instrument (TMS320C30) that has 60 ns instruction cycle. The DSP has 2K words of internal RAM, while the EVM has 16K words of zero-wait

state static RAM for codes and data.

The EVM is equipped with an analog interface circuit (AIC) based on the TLC32044. The TLC32044 is a programmable, 14 bits A/D and D/A in a single chip. It also has a programmable filter and sampling rate controller.

To communicate with its host, the EVM has 16-bit bidirectional ports. At the EVM side, the data interchanges can be driven by either interrupts, or status polling, or direct memory access (DMA). At the host side, the interchanges are driven by status polling.

Block Distribution

The hardware platform influences how the functional blocks are implemented. The A/D and D/A sections are implemented using the AIC of the EVM. The CELP analyzer and synthesizer are implemented on the EVM because they, especially the analyzer, require very high computational power. They are also physically close to the AIC. The error protection, data stream arrangement, and the data media interfaces are done on the host computer. The storage device is the host hard disk. The packet radio must be connected to the host serial port. Finally, the controller is implemented on the PC-AT. The PC-AT then has a user interface on which we can type in commands to choose functions the system has to perform.

3.2 Software Organization

Since some functional blocks have been implemented in hardware, the remaining blocks must be developed in software. However, since the error protection, correction, and data arrangement are look-up table schemes, as well as the media interfaces are standard disk operating system (DOS) procedures, we concentrate on the controller, CELP synthesizer, and CELP analyzer.

Controller

Figure 4 shows a flowchart of the controller. The program begins with initializations of the system including the serial port and the EVM. It also sets the system state to a default setting. The program displays a menu, then wait for a command from a user, and execute the subprograms related to the command.

The most critical and challenging operation mode is the real-time, full-duplex communication mode. We concentrate on this mode, because the other modes are simple modifications of this mode. Figure 5 shows an approximate schedule of launching PC-AT and EVM subprograms during the real-time, full-duplex mode. Notice that the process is always repeated every frame of 240 speech samples,

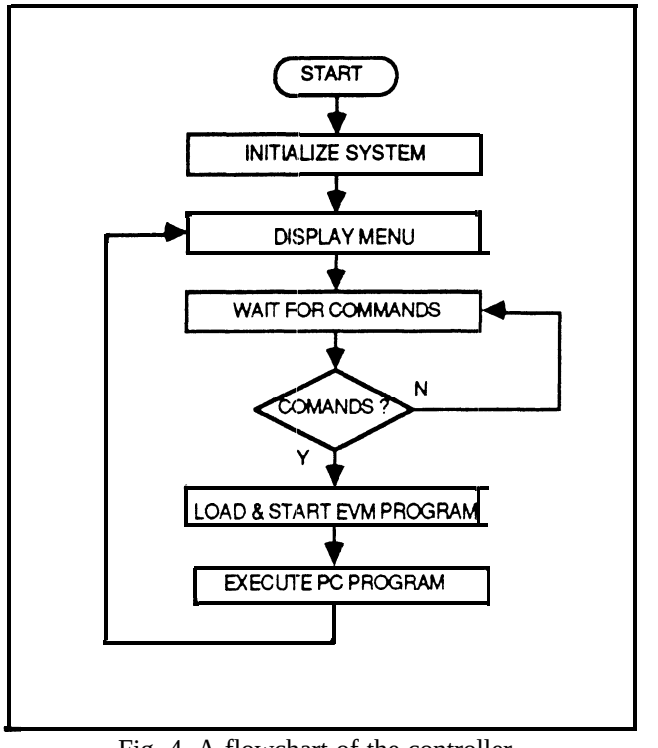

Fig. 4. A flowchart of the controller.

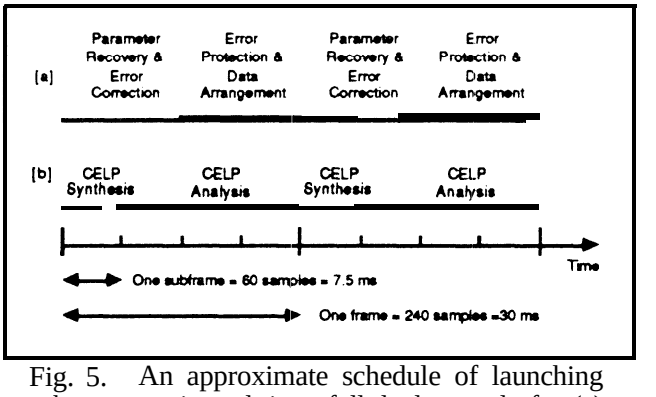

Fig. 5. An approximate schedule of launching subprograms in real-time, full-duplex mode for (a) PC-AT, and (b) EVM. There are interrupts during the execution of the subprograms for analog and serial RS232C I/O on the EVM and PC-AT, respectively. There are also data exchanges between the PC and the EVM at the end of a frame.

The program runs in both foreground (interrupt driven) and background. The main program runs in the background. It consists of a CELP synthesizer and a CELP analyzer.

They are responsible for converting speech data within the given period. The source and destination of the data are several working buffers.

On the other hand, interrupt services run in the foreground. An interrupt occurs in the EVM every 125 μs according to the sample frequency of the A/D. The interrupt service gets a sample from A/D, stores it on a working buffer, takes another synthetic speech sample from another working buffer, and supplies the data to the D/A. Another interrupt also occurs in the PC-AT for serial RS232C communication. Here, its service also does similar response, except the data are the CELP parameters and the port is the RS232C.

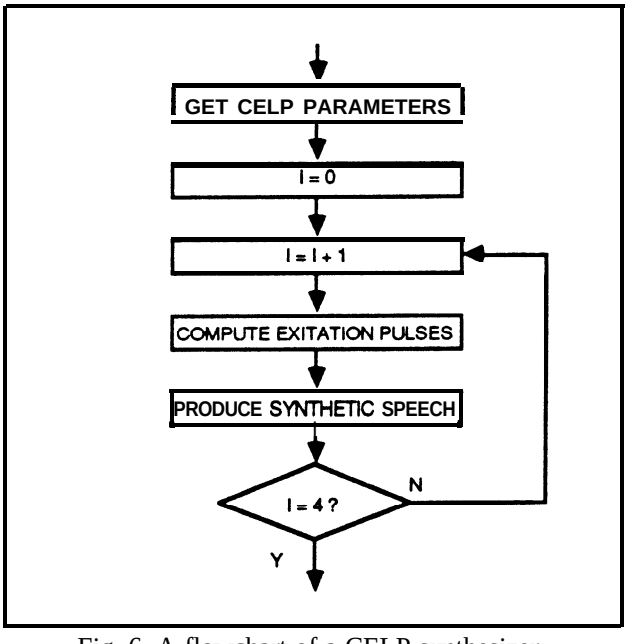

Fig. 6. A flowchart of a CELP synthesizer.

CELP Synthesizer

Figure 6 shows a flowchart of a CELP synthesizer corresponding to the scheme in Fig. 2. The program reads the code book parameters from the working buffer and use them to produce 60 excitation pulses. The content of the ACB is updated using the new excitation pulses. A set of LP coefficient is obtained from the previous and current LSP according to an interpolation rule, and is used to replace the LP filter coefficients. The new LP filter then receives the excitation pulses, and produces 60 samples of synthetic speech. The speech is stored in another working buffer. The process is repeated three more times until 240 synthetic samples have been obtained in the working buffer. The program then returns to the controller, and the controller launches the CELP analyzer.

CELP Analyzer

Figure 7 shows a flowchart of a CELP analyzer corresponding to the scheme shown in Fig. 4 with a slight modification. Here, the PW filter is cascaded with the CELP synthesizer block. The LP filter inside the CELP synthesizer can then be combined with the PW filter. A duplicate of the PW filter is placed before the summation block to compensate the change.

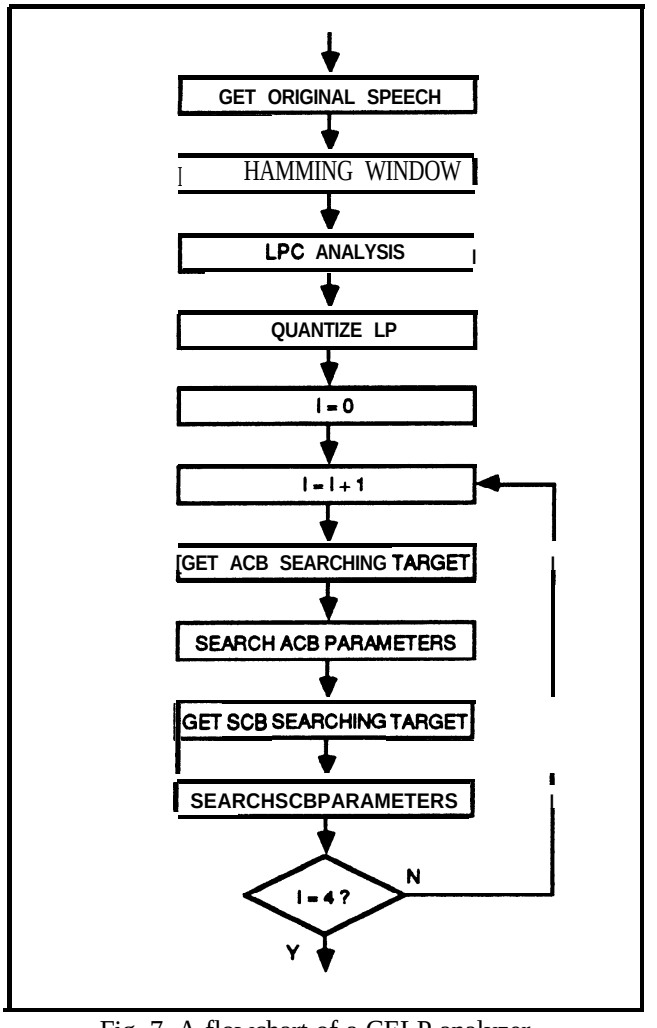

Fig. 7. A flowchart of a CELP analyzer.

The program reads the speech data from the working buffer and weights the data using Hamming window [Pars86]. The program then performs LPC analysis to obtain the LP coefficients. The LP coefficients are quantized into a line spectrum pair (LSP) form [SoJu84].

The quantized LP coefficients are used in the LP filter for finding code book parameters.

To obtain ACB parameters, we must first provide the ACB searching target. The program computes the zero response of the new LP filter and subtracts the zero response from the original signal to compensate ringing from previous frames. The resulting signal is applied to the PW filter to obtain the ACB searching target. A code book searching subroutine (joint optimization) is then called to find the ACB parameters.

To obtain SCB parameters, we must also provide the SCB searching target. The ACB parameters obtained above is used to create excitation pulses. The program applies the pulses to the cascade of LP and PW filter. The resulting signal is subtracted from the ACB searching target mentioned earlier to have the SCB searching target. Having the searching target, the program calls joint optimization subroutine to find the SCB parameters.

The joint optimization scheme automatically quantizes the gain factors. The code book entries are not quantized except for the ACB entries. For each even subframe, the ACB entry is represented by the offset of the actual entry relative to the previous ACB entry.

4. SYSTEM DEVELOPMENT

The use of the EVM makes the system development much easier because the EVM is accompanied with powerful development tools. However, applying careful development methodology is still critical due to several problems. First, fast CELP algorithms are not trivial, and involve many computational processes. Second, the EVM has a limited size of installed memory, resulting in a need to compact the program size and to schedule the use of working buffers. Third, the real-time requirement demands a way of coding every function to achieve execution time as short as possible. Finally, since the TMS320C30 itself is a processor with pipelining and parallel execution, a higher complexity of software must be considered. System development begins with hardware setup, software development, and finally system test.

4.1 Hardware Setup

The final system is a PC-AT equipped with the EVM, a two channel amplifier, a microphone, a speaker, a TNC, and a transceiver. For full-duplex implementation the TNC and the transceiver must be set accordingly. However, a smaller configuration can be used during the development process. A PC-AT is sufficient to code and test EVM routines because the EVM development system has a TMS320C30 simulator. The results can be used for real-time debugging and testing on a PC-AT with the EVM installed.

4.2 Software Development

The algorithms are first simulated on a SUN workstation using C programming. This simulation is essential because the algorithms are complex. The C program is gradually converted into modular subroutines according to the architecture and flowcharts mentioned earlier. Each subroutine module is optimized and, one by one, recoded into TMS320C30. Each module is then tested before being tailored into the EVM program.

There are two ways to convert a C rnodule into its TMS320C30 counterpart. First, the EVM development tools have a C compiler tlhat automatically transfers a C program into a TMS320C30 program. Second, we can write a TMS320C30 program directly with the help of the logical structure and data type a program has. Although we can have full control and very efficient codes using the latter approach, the former approach is more convenient. We utilize both by using the first method to have a draft of code, and using the second method to compact the draft. However, for simple control and time critical process, such as interrupt handling, we use the second method.

Each routine is tested off line using a TMS320C30 simulator. Here, we can check the number of clocks and registers used. In addition, we can control several options, such as pipelining, for easier debugging. A real-time debugger is then used to test the program in actual processor. Here, we can fine tune the software and test the peripheral specific functions that are not fully supported by the simulator. Finally, the verified codes are loaded into the EVM for normal operation or operational tests.

4.3 Preliminary Test

Simple tests have been performed to verify the system. We connected the serial port of the PC-AT to another computer at a rate of 4.8 kbit/s. The other computer received the CELP parameters and immediately returned the parameters back to the PC-AT. The PC-AT reconstructed the speech from the parameters for a listening process. This

test did not cover the effect of the channel noise to the system performance, nor the practical problems that may be encountered in a real application using a TNC and a transceiver. However, it did verify the speech compression capability and quality of the system.

A study to enable the system to be used for mobile communication is desirable for future work. In the future, this capability may be increasingly of interest. The study involves the finding of good error protection schemes, especially to face noise of multipath fading channels.

5. CONCLUSIONS

The design and implementation of a speech processing system for communication through packet radio have been described. The system implements CELP coding scheme to achieve high speech quality in a bit rate as low as 4.8 kHz. It is implemented on a low-cost PC-AT equipped with a TMS320C30 EVM. It is capable of real-time, full-duplex operation, and can be modified for other modes. Although the CELP algorithms are complex, the system is developed easier using the powerful tools accompanied with the EVM and the development method for coding a signal processing application. A simple test verifies the speech compression and quality of the system. Future work and tests are still needed to enhance the system for practical and advanced applications, such as voice mail.

ACKNOWLEDGEMENT

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada, the Inter University Centre (IUC) on Microelectronics, ITB, Indonesia, and Signals and Systems Laboratory, Department of Electrical Engineering, ITB, Indonesia. Special thanks go to Dr. R. McLeod for providing the TMS320C30 development facilities.

REFERENCES

[Bloo90] J. Bloom, "Considering next-generation amateur voice systems," *ARRL/CRRL 9th Computer Networking Conf.*, pp. 10-15, 1990.

[CaTW90] J. P. Campbell, Jr., T. E. Tremain, and V. C. Welch, "The proposed Federal Standard 1016 4800 bps voice coder: CELP," *Speech Technology,* pp. 58-64, Apr./May 1990.

[Kins89] W. Kinsner, "Speech recording and synthesis using digital signal processing," *CRRL Convention,* Winnipeg, MB, Canada, 21 pp., August 1989.

[LaKi90] A. Langi and W. Kinsner, "CELP high-quality speech processing for packet radio transmission and networking," *ARRLICRRL 9th Computer Networking Conf.*, pp. 164-169, 1990.

[LaKi91] A. Langi and W. Kinsner, "Code-excited linear predicitve speech processing for digital transmission and storage," *Proc. the IEEE Western Canada Conference on Computer, Power, and Communication Systems in a Rural Environtment,* IEEE 91CH2927-2, pp. 205-209, 1991.

[RTWC89] D. J. Rahikka, T. E. Tremain, V. C. Welch, and J. P. Campbell, Jr., "CELP coding for land mobile radio applications," *Proc. Int. Conf. Acoustics, Speech & Signal Processing,* IEEE CH2673-2/89, pp. 465-468, 1989.

[ScAt85] M. R. Schroeder and B. S. Atal, "Code-Excited Linear Prediction (CELP): High quality speech at very low bit rates,* *Proc. Int. Conf. Acoustics, Speech & Signal Processing,* IEEE CH2118-8/85, pp. 937-940, 1985.

[Youn90] N. R. Young, "A discussion of the issues and technologies for HF voice communication," *Proc. Canadian Conference on Electrical and Computer Eng.,* EIC, pp. 68.2.1-68.2.4., 3-6 Sept. 1990.