TEXAS INSTRUMENTS

Application Report
SLAA361 – July 2007

Speech and Sound Compression/Decompression With MSP430

Christian Hernitscheck *MSP430 Application Europe*

ABSTRACT

The report explains an IMA Adaptive Differential Pulse Code Modulation (ADPCM) compression/decompression algorithm and the steps to use the ADPCM library on the MSP430. The usage of the ADPCM library is described for two voice recorder examples that use the signal-chain-on-chip feature of the MSP430 microcontrollers.

Contents

1 Introduction 2
2 Compression and Decompression Algorithms 2
3 MSP430 Signal Chain on Chip 5
4 Performance on the MSP430 7
5 References 8

List of Figures

1 DPCM Encoder and Decoder Block Diagram 2
2 Comparison of 8-Bit Audio Data and Difference of Successive Samples (Differential PCM Step Size = 1) 3
3 ADPCM Encoder and Decoder Block Diagram 4
4 Signal-Chain-on-Chip Solution Using MSP430F169 5
5 Signal-Chain-on-Chip Solution Using MSP430FG4618 6

1 Introduction

Sound recorders are implemented with relative ease on microcontrollers (MCUs). There are many MCUs that have an integrated analog-to-digital (A/D) converters. Sound captured by a microphone followed by an amplifier is fed to the analog input of an A/D converter. The recorded sound can then be stored in memory (flash or RAM). A button pressed can trigger the MCU to play back the recorded sound. This is done by moving the stored data to a digital-to-analog (D/A) converter followed by an audio power amplifier.

Such a sound recorder can be easily realized using the MSP430. The MSP430 microcontroller takes advantage of the integrated peripherals to form an on-chip analog signal chain. In addition, the CPU of the MSP430 is powerful enough to perform compression of the recorded sound.

2 Compression and Decompression Algorithms

The simplest way to realize, for example, a voice recorder is to store the A/D conversion results (e.g., 12-bit samples) directly in the flash memory. Most of the time, the audio data does not use the complete A/D converter range, which means that redundant information is stored in the flash memory. Compression algorithms remove this redundant information, thereby reducing the data that must be stored.

Adaptive differential pulse code modulation (ADPCM) is such a compression algorithm. Various ADPCM algorithms exist, and differential coding and adaptation of the step-size of the quantizer scheme is common to all. Before taking a closer look at the IMA ADPCM algorithm, which is used in the associated code, a short description of the differential PCM coding is given.

2.1 Differential Pulse Code Modulation (DPCM)

DPCM encodes the analog audio input signal using the difference between the current and the previous sample. Figure 1 shows a DPCM encoder and decoder block diagram. In this example, the signal difference, d(n), is determined using a signal estimate, Se(n), rather than the previous input. This ensures that the encoder uses the same information available to the decoder. If the true previous input sample were used by the encoder, an accumulation of quantization errors could occur. This leads to a drift of the reconstructed signal from the original input signal. By using a signal estimate as shown in Figure 1, the reconstructed signal, Sr(n), can be prevented from drifting from the original input signal. The reconstructed signal, Sr(n), is the input to the predictor, which determines the next signal estimate, Se(n+1).

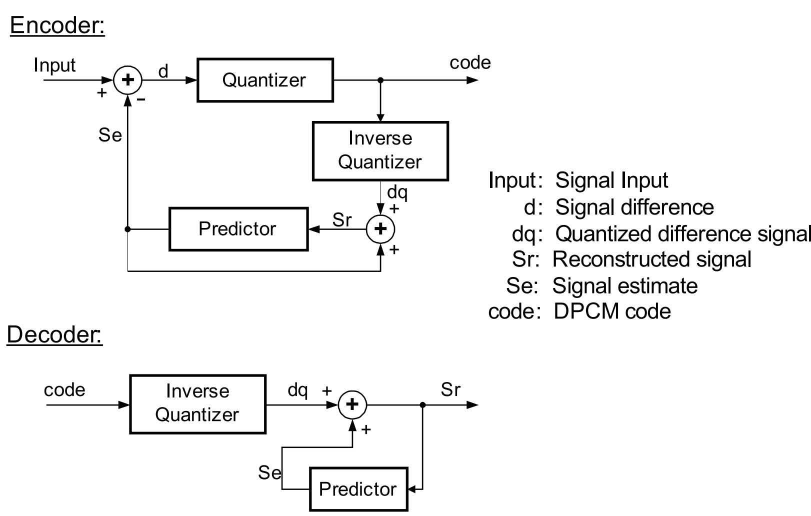

Figure 1. DPCM Encoder and Decoder Block Diagram

Figure 2 shows a small part of a recorded audio stream. Analog audio input samples (PCM values) and the differences between successive samples (DPCM values) are compared in the two diagrams in Figure 2.

The range of the PCM values is between 26 and 203, with a delta of 177 steps. The encoded DPCM values are within a range of –44 and 46, with a delta of 90 steps. Despite a quantizer step size of one, this DPCM encoding already shows a compression of the input data. The range of the encoded DPCM values could be further decreased by selecting a higher quantizer step size.

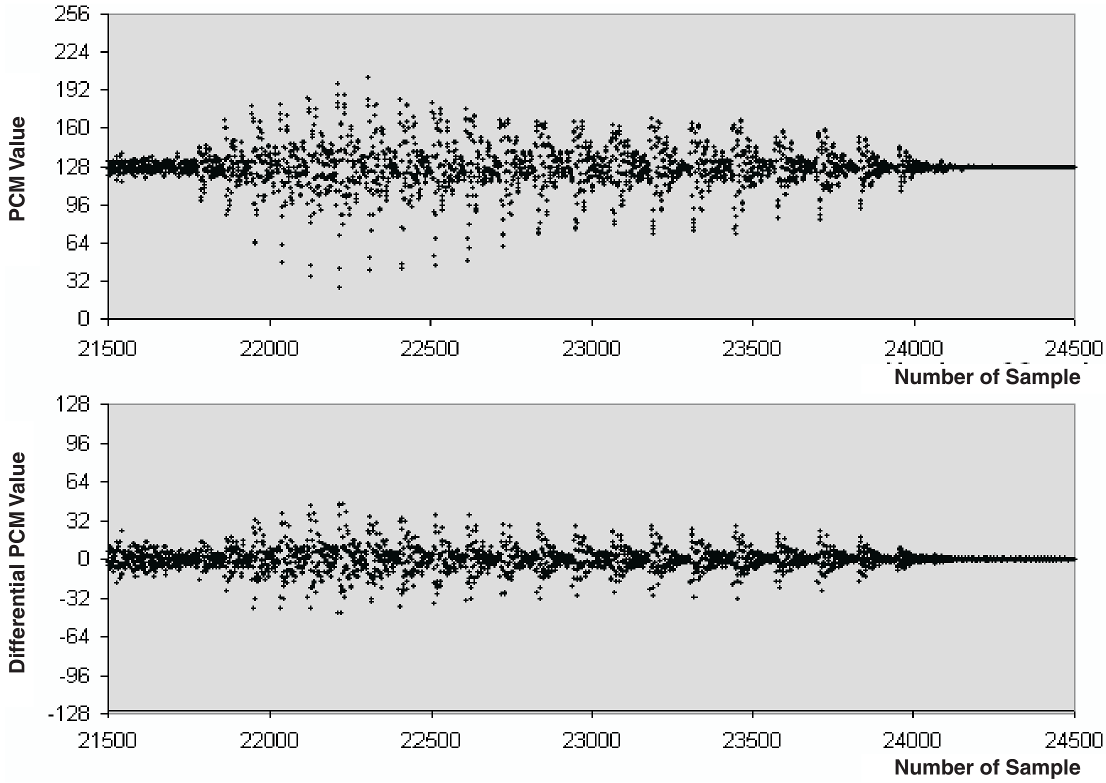

Figure 2. Comparison of 8-Bit Audio Data and Difference of Successive Samples (Differential PCM Step Size = 1)

2.2 *Adaptive Differential Pulse Code Modulation (ADPCM)*

ADPCM is a variant of DPCM that varies the quantization step size. Amplitude variations of speech input signals are seen between different speakers or between voiced and unvoiced segments of the speech input signal. The adaptation of the quantizer step size takes place every sample and ensures equal encoding efficiency for both low and high input signal amplitudes. Figure 3 shows the modified DPCM block diagram including the step-size adaptation.

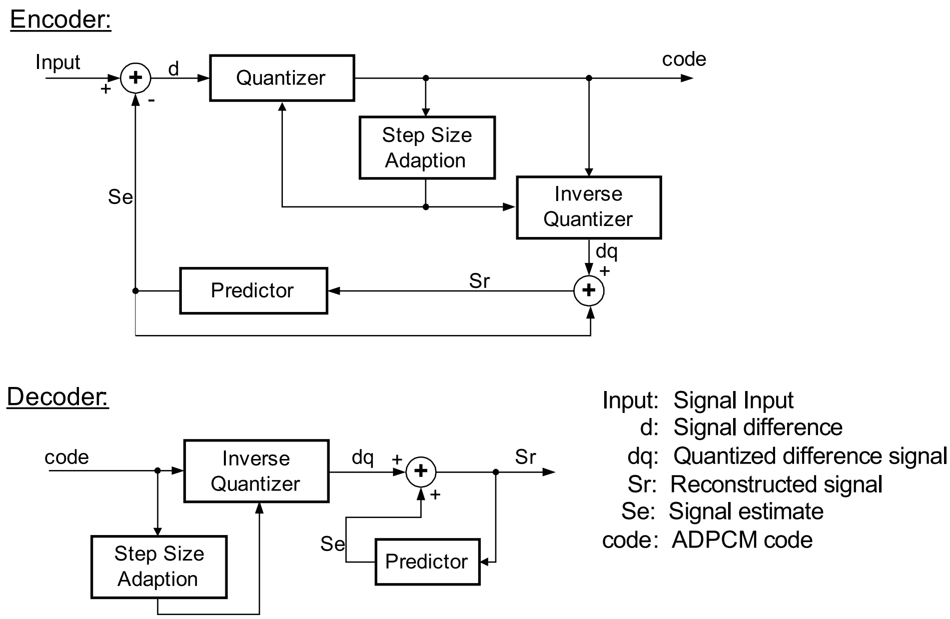

Figure 3. ADPCM Encoder and Decoder Block Diagram

The ADPCM encoder calculates the signal estimate, (Se), by decoding the ADPCM code. This means that the decoder is part of an ADPCM encoder. Hence, the encoded audio data stream can only be replayed using the decoder. This means that the decoder must track the encoder.

The initial encoder and decoder signal estimate level, as well as the step-size adaptation level, must be defined before starting encoding or decoding. Otherwise, the encoded or decoded value could exceed the scale.

3 MSP430 Signal Chain on Chip

The MSP430 family of microcontrollers offers a variety of on-chip peripherals. To complete a signal-chain-on-chip solution, the MSP430 must have at least one analog input (to the A/D converter) and a D/A converter. Two MSP430 solutions are introduced in the following sections.

3.1 MSP430F169 Signal-Chain-on-Chip Solution

The MSP430F169 has an integrated 12-bit SAR A/D converter, a hardware multiplier module that allows efficient realization of digital filters, and an integrated 12-bit D/A converter module. Such a signal chain circuit is shown in Figure 4 using the MSP430F169.

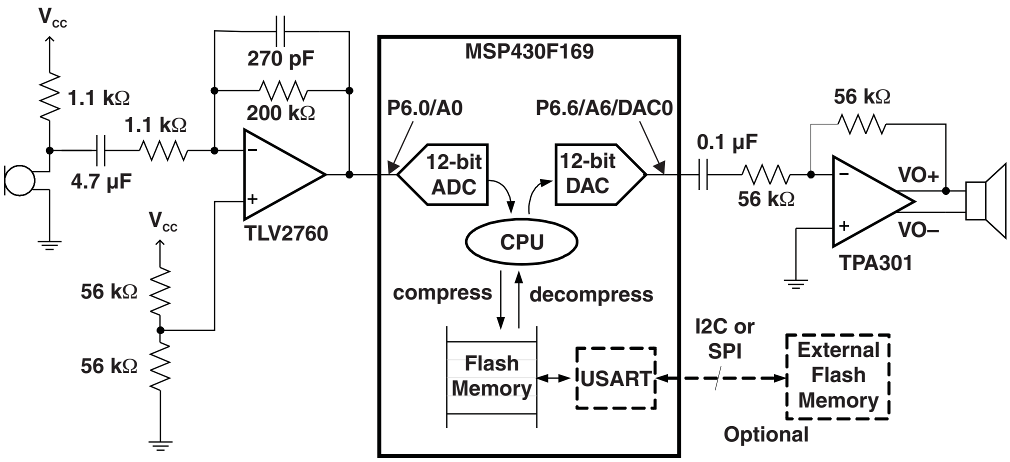

Figure 4. Signal-Chain-on-Chip Solution Using MSP430F169

This configuration also works with an external serial flash memory, which could be used for storage of the audio data. It is possible to connect an external flash memory via the I^2C or SPI interface of the MSP430. The CPU loading could be greatly reduced by using the MSP430F169 DMA module for automatic transfers of received data to RAM.

3.2 MSP430FG4618 Signal-Chain-on-Chip Solution

Another signal-chain-on-chip solution could be realized with an MSP430FG4618. While the MSP430F169 has 60-Kbyte integrated flash memory, the MSP430FG4618 has 116-Kbyte flash memory. Another advantage of the MSP430FG4618 is the integrated operational amplifier module. These operational amplifiers can be used to amplify the microphone input and the analog output of the D/A converter. Figure 5 shows the signal chain of the MSP430FG4618 solution. This is the configuration of the MSP430FG4618/F2013 Experimenter's Board from Texas Instruments. This evaluation board can be used with the associated code example.

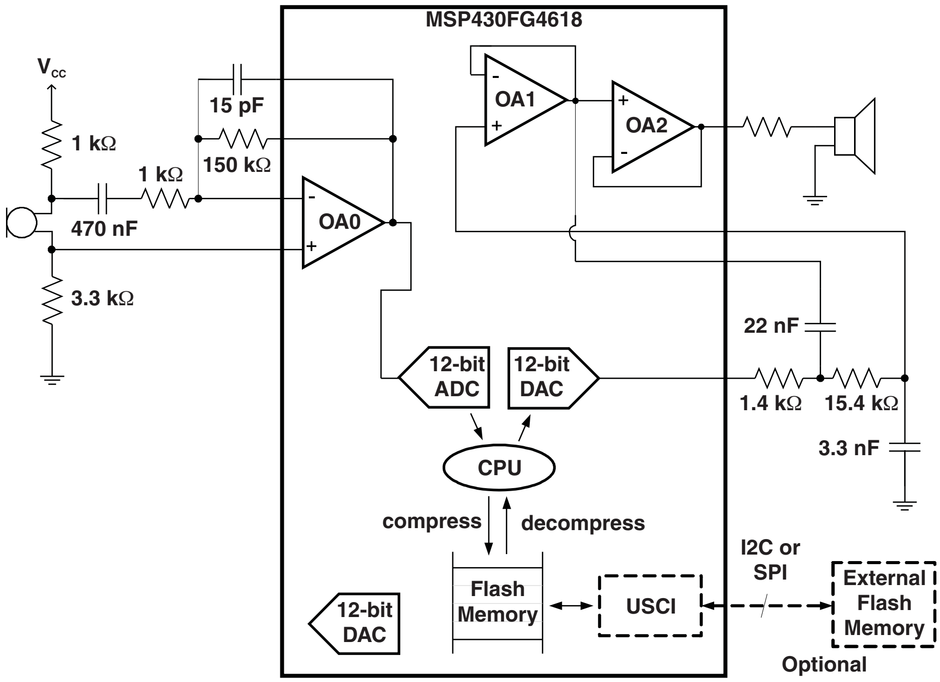

Figure 5. Signal-Chain-on-Chip Solution Using MSP430FG4618

The output signal of the microphone is quite small and must be amplified. The MSP430's operational amplifiers can be used in different operating modes. Using the amplifier in PGA mode allows a maximum amplification of only 15, which is not enough for the microphone amplifier. Hence, additional gain is provided using external components, and operational amplifier OA0 in Figure 5 is used in general-purpose amplifier mode. There are eight performance settings for the amplifiers, allowing performance (gain-bandwidth product and slew rate) to be traded with current consumption. The high-performance mode (fast mode) is chosen for all amplifiers (OA0, OA1, and OA2).

Further information about the operational amplifier usage can be found in the *MSP430FG4618/F2013 Experimenter's Board User's Guide*.

The universal serial communication interface (USCI) could be used to store the audio data in an external flash memory. The connection to the external memory could be done either via an I^2C bus or and SPI bus.

4 Performance on the MSP430

There are some "*.wav" file examples in the associated code file demonstrating the quality of a decoded ADPCM data. Comparing these files by using for example media player software on the PC helps to get an idea about the quality that can be realized using an ADPCM compression algorithm. Note that the quality can be improved by increasing the audio sample rate and optimizing the audio sample size (resolution).

4.1 Using the Associated Code

There are two software projects included in the associated code. These two versions are based on the two descriptions in Section 3, and both use the IMA ADPCM algorithm.

The usage of the ADPCM functions is simple. First, ADPCM.h must be included in the application code. This header file declares the ADPCM functions of the ADPCM.c file. Before each sequence of recording or replaying audio data, the ADPCM_Init() function must be called. This function defines the start values for the signal estimate (Se) and the step size pointer that is used for the quantizer step size adaptation. The encoder and decoder are synchronized by these settings. Encoding is realized by calling the ADPCM_Encoder(int value) function, and playback is realized by calling ADPCM_Decoder() for each audio sample. The following code segment demonstrates how this could be done.

```
#include "ADPCM.h"

void main(void)
{   // initialization of application software

    while(1)     // Main Loop
    {  // application software
       if (P1IN & 0x01)
           record();
       if (P1IN & 0x02)
           play();
    }
}

void record(void)
{   // initialization for recording (A/D converter, timer, amplifier, ...)
    ADPCM_Init();    // this is done before recording is started

    // start recording
}

void play(void)
{   // initialization for recording (D/A converter, timer, amplifier, ...)
    ADPCM_Init();    // this is done before playback is started

    // start playback
}
```

The following measurements of the ADPCM function execution times were made with IAR Embedded Workbench™ KickStart™ version 3.42A. The default optimization settings were used for the measurements.

ADPCM_Encoder() function call requires between 114 and 126 cycles.

ADPCM_Decoder() function call requires between 99 and 109 cycles.

Note that this only covers the compression/decompression algorithms. For realizing recording and playback functionality additional code is needed.

5 References

1. *MSP430x4xx User's Guide* (SLAU056)
2. MSP430F169 data sheet (SLAS368)
3. MSP430FG4618 data sheet (SLAS508)
4. *32-kbit/s ADPCM With the TMS32010* (SPRA131)
5. *A Low-Cost 12-Bit Speech CODEC Design Using the MSP430F13x* (SLAA131)
6. *MSP430FG4618/F2013 Experimenter's Board User's Guide* (SLAU213)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
Low Power Wireless	www.ti.com/lpw	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265