

- 1 -

MULTIPLE INSTANCE IMPLEMENTATION OF SPEECH CODECS

Field of the Invention

5 This invention relates to audio coding, and is particularly applicable in the field of telecommunication and the internet, for example, in which multi-channel speech coders are implemented using a single digital signal processing (DSP) device.

Background

10

Because of the computationally expensive nature of encoding and decoding real-time audio, traditionally audio coders/decoders ("codecs") have been implemented by specialized integrated circuit chips. An alternative is to provide the audio codec functions in software (firmware, micro-code, etc.) for operation on a more general purpose Digital Signal Processor ("DSP").

15 With the deployment of more powerful Digital Signal Processors, it can be possible for more than one codec to be real-time implemented using a single DSP chip. For example, it may be possible for a powerful DSP to process a plurality of different audio signals or channels in real-time, to thereby implement a plurality of voice coders ("vocoders") using a single processing device.

20

Summary of the Invention

In accordance with the present invention, there is provided a method for implementing a plurality of encoders and/or decoders (codecs) using a single digital signal processing (DSP) device, wherein the function of each codec is performed by the DSP according to an instruction code program, the method comprising the steps of:

25 providing an instruction code program stored in a first memory for controlling the DSP to function as a codec;

 providing a second memory including a plurality of memory segments;

- 2 -

implementing a plurality of codecs using the DSP by running said instruction code program in the first memory a plurality of times in re-entrant instances, wherein each codec instance is provided access to a respective separate memory segment in the second memory for storing data used in encoding/decoding a respective separate data stream.

5

The present invention also provides a data coding and/or decoding system in which a plurality of encoders and/or decoders (codecs) are implemented using a single digital signal processing (DSP) device, comprising:

a digital signal processor (DSP);

10 a first memory coupled to the DSP and containing an instruction code program, the function of each codec being performed by the DSP, in use, according to the instruction code program;

a second memory coupled to the DSP and partitioned to include a plurality of separate memory segments;

wherein a plurality of codecs are implemented using the DSP by running the instruction
15 code program in the first memory a plurality of times in re-entrant instances, and wherein each codec instance is provided access to a respective separate memory segment in the second memory for storing data used in encoding/decoding a respective separate data stream.

Preferably a third memory is also provided which is accessible by each of the codec
20 instances for shared storage of temporary variables and data buffering in encoding/decoding said respective separate data streams.

Each codec instance preferably accesses the corresponding memory segment using indirect addressing based on at least one index register, the at least one index register being set for each
25 codec instance to modify addressing of variables for that codec instance to the corresponding memory segment.

In a preferred form of the invention the plurality of memory segments are contiguous in the second memory, and the at least one index register is set for each codec instance according to

- 3 -

an offset based on the difference in address from a first of said memory segments to the memory segment corresponding to that codec instance. Preferably each of the memory segments in said second memory is the same size.

5 In one form of the invention, the first memory is provided with a plurality of instruction code programs for implementing different kinds of codecs, wherein different codec instances may be selected from the different kinds of codec. Preferably, then, each of the memory segments in the second memory is the same size, and the size of the memory segments is selected according to the maximum memory required by any of the plurality of different kinds
10 of codec.

This methodology can be used to modify the existing DSP assembly code or to implement a new algorithm with a multiple instance feature. The multiple-instance implementation can be used to process multi-channel signals using one software module.

15

Brief Description of the Drawings

The invention is described in greater detail hereinafter, by way of example only, with reference to preferred embodiments thereof and the accompanying drawings, in which:

20 Figure 1 is a block diagram of a simple audio encoding system employing a DSP;

Figure 2 is a block diagram of a data memory structure for a first embodiment of the present invention; and

Figure 3 is a block diagram of a data memory structure for a second embodiment of the invention employing a plurality of different codec types.

25

Detailed Description of the Preferred Embodiments

The embodiments of the invention described hereinbelow are in the context of audio coders implemented by instruction codes in the form of software, firmware or micro-code operating

- 4 -

on a general purpose digital signal processor chip. In particular, the described embodiments relate to implementation of speech codecs of the type which is the subject of ITU-T Recommendation G.729. However, it will be appreciated by those of ordinary skill in the art that the present invention is not necessarily limited to such an environment, and may further be
5 used to implement coders for other forms of real-time data streams including other forms of voice and/or facsimile transmission or storage coding.

A simple audio encoding system 10 based on a general purpose DSP is illustrated in block diagram form in Figure 1. The audio coding system 10 comprises a digital signal processor
10 (DSP) 12 which is coupled to receive input data and produce coded output signals. The DSP 10 is coupled to a storage memory 14. The memory 14 provides data to the DSP 12 to enable coding of the received input data to take place. The memory 14 may contain various different forms of data, including read-only data which is permanently stored in the memory, and temporary data which is transiently stored during coding operations. The memory 14 may also
15 contain stored instruction codes used by the DSP to perform the coding, which would ordinarily be permanently stored in read-only form.

One of the considerations which must be dealt with in implementing multiple codecs on a single DSP by operating multiple instances of the DSP instruction codes which control the
20 codec operation is the arrangement of storage memory so that each instance of the codec can have efficient memory usage and not interfere with other codec instances in use at the same time. In accordance with an embodiment of the present invention, this is achieved by arranging the data memory as illustrated in the block diagram of Figure 2. This exemplary embodiment relates to implementation of a 4-channel multiple instance vocoder on a single DSP. As shown
25 in Figure 2, the data memory is partitioned into different parts, in this case comprising read-only memory (ROM), local RAM, and static RAM. The ROM is used to store all the read only data, which may include the program instruction codes and the like. Local RAM is used for storage of temporary variables and for data buffering. The static RAM, on the other hand, is used for storage of all the global and historical variables required for the coding operations, for example

- 5 -

data which will be used for processing the next frame of output. As shown in the block diagram, in this case the static RAM is partitioned into separate areas for each DSP codec instance. Thus, static RAM_i is constrained for use only by the i^{th} coder ($i = 1, 2, 3, 4$), while the data ROM and local RAM are shared by all four channels.

5

The preferred form of the present invention involves partitioning the data memory based on their properties and using an index register or index modifier to modify the address of the static RAM segment data used for each codec instance. The program code memory can be arranged as usual. If, however, program memory is used for data accessing, this part of memory
10 also needs to be partitioned in the same way as data memory such as X or Y. This methodology allows the easy management of the operating system or real-time kernel due to the facility of opening and closing the various instances.

First, the partition of the data memory according to its accessing property is obtained, for
15 example in the embodiment shown in Figure 2 involving ROM data, temporary local RAM, and static or global RAM segments. The data ROM and temporary local RAM are shared for every instance. Only the static RAM segments are used exclusively by corresponding codec instances. This static RAM segment is accessed using an *index register* by adding an offset value to address the static RAM segment for a particular instance. This is the basis of method to
20 implement a new algorithm for multiple codec instances on a single DSP. Using this method, as described in greater detail below, existing DSP program instructions can be modified relatively easily to operate multiple codec instances.

When modifying existing DSP codes for multiple instance operation, it is not necessary to
25 change all of the existing variable names using the arrangement of the data memory with partitions as described above. For static data memory access, it is only necessary to modify the existing addressing mode to obtain an indirect addressing scheme using the *index* and *address registers*. One example is shown in the Table below using a D950 DSP made by STMicroelectronics. For other DSPs, a similar approach can be employed. In the Table, ax0

- 6 -

is one of the address registers and ix1 and ix2 are index registers.

Before (non-re-entrance)	Modified (re-entrance)
5 <i>Direct</i> var1.x = Register1 (note #1)	<i>Indirect</i> ax0 = #var1 (note #2) *(ax0 + ix2) = Register1 (note #3)
10 <i>Indirect</i> ax0 = #var10 (note #2) Rep N times loop1 (note #5) Register = *ax0+ix1 (note #6) ... loop1 (note #7)	<i>Indirect</i> ax0 = #var10 (note #2) ax0 = ax0+ix2 (note #4) Rep {N-1} times loop1 (note #5) Register = *ax0+ix1 (note #6) ... loop1 (note #7)

15 In order to pass the addressing information to every channel vocoder, two index registers should be reserved for this reentrant purpose, say IX2 and IY2 if using a D950 DSP. By doing so, the address of all the static variables stored in the respective static RAM segments will be modified by these two index registers from the beginning.

20

There are two methods by which stored data can be access from the static RAMs. One is direct addressing and the other is indirect addressing. However, for reentrant implementations, access to the static RAM must be through indirect addressing. Therefore, to achieve the reentrant feature and run multiple instance codecs, the existing instruction
 25 codes for the DSP should be modified accordingly, using the index register, indicated as an example in the Table above.

If the addressing instruction codes are modified accordingly and data memory is arranged as shown in the Figure 2, the multiple-instance 4-channel system can operate as follows,

- 7 -

Repeat following codes

/*first instance*/

Ix2=#offsetx₁

Iy2=#offsety₁

5 Call coder

/* second instance*/

Ix2=#offsetx₂

Iy2=#offsety₂

Call coder

10

/*third instance*/

Ix2=#offsetx₃

Iy2=#offsety₃

Call coder

15

/*fourth instance*/

Ix2=#offsetx₄

Iy2=#offsety₄

Call coder

20

where offsetx_i=offsety_i=0, offseti=addressx_i-addressx₁, offsety_i=addressy_i-addressy₁, for i=1, 2, 3. Address x_i does not necessarily equal address y_i. One example of the static RAM size is about 1560 words in X memory and 256 words in Y memory for a codec operating according to ITU-T G.729. The static RAM size would typically be the same for each
25 instance of the G.729 vocoder.

This is applicable for a new implementation or for converting existing DSP instruction coding to a re-entrant multiple instance implementation. Since this method does not require a change of the variable names in the DSP coding, it is efficient in both MIPS and porting

- 8 -

effort if compared to using pointers.

If there are more than one kind of speech codec in a system, the method described herein is still applicable. For a system using a plurality of different codecs, the memory may be
5 arranged as illustrated in Figure 3, for which the multiple-instance accessing procedures are described below.

In Figure 3 the memory map for a 4-channel system is shown, in which the 4 active channels could be any possible combination of codecs operating according to ITU-T G729, G723.1 and FAX. In this case, the size of ROM segment comprises ROM data for all of the
10 codec types. The size of the temporary local RAM segment is the maximum local RAM size used by any of the codecs. All of the different codecs share the local RAM segment. Similarly, each of the static RAM segments are of a size to accommodate the frame processing storage requirements of any one of the possible codecs implementations. This enables different codecs to make use of different static RAM segments.

15

One example is a 4-channel system with two different speech codecs, say G723.1 and G729. The program instruction codes are, in this example, stored separately in a program memory (not shown). The data memory is arranged as shown in Figure 3. Such a 4-channel system multiple instance system can operate as follows,

20

```
Repeat following codes
/*first instance*/
Ix2= #offsetx1
Iy2= #offsety1
25 If (codec_flag1==CODEC1)
    Call coder1
    Else
    Call coder2
```

- 9 -

```

/*second instance*/
Ix2=#offsetx2
Iy2=#offsety2
If (codec_flag2==CODEC1)
5   Call coder1
   Else
   Call coder2

/*third instance*/
10  Ix2=#offsetx3
   Iy2=#offsety3
   If (codec_flag3==CODEC1)
   Call coder1
   Else
15  Call coder2

/*fourth instance*/
   Ix2=#offsetx4
   Iy2=#offsety4
20  If (codec_flag4==CODEC1)
   Call coder1
   Else
   Call coder2

```

25 where $\text{offsetx}_i = \text{offsety}_i = 0$, $\text{offsetx}_i = \text{addressx}_i - \text{addressx}_1$, $\text{offsety}_i = \text{addresy}_i - \text{addresy}_1$ for $i=1, 2, 3$. The flag variable *codec_flag_i* is writeable by an operating system or system controller, so as to dictate which type of codec is used for each instance.

The foregoing detailed description of the present invention has been presented by way of

- 10 -

example only, and many variations to the specific details therein may be apparent to those of ordinary skill in the art without departing from the scope of the invention. Accordingly, such specifics are not intended to be considered limiting to the scope of the invention which is defined in the appended claims.

5

- 11 -

Claims

1. A method for implementing a plurality of encoders and/or decoders (codecs) using a single digital signal processing (DSP) device, wherein the function of each codec is performed by the DSP according to an instruction code program, the method comprising the steps of:
 - 5 providing an instruction code program stored in a first memory for controlling the DSP to function as a codec;
providing a second memory including a plurality of memory segments;
implementing a plurality of codecs using the DSP by running said instruction code program in said first memory a plurality of times in re-entrant instances, wherein each codec
10 instance is provided access to a respective separate memory segment in said second memory for storing data used in encoding/decoding a respective separate data stream.
2. A method as claimed in claim 1, wherein a third memory is provided which is accessible by each of the codec instances for shared storage of temporary variables and data buffering
15 in encoding/decoding said respective separate data streams.
3. A method as claimed in claim 1 or 2, wherein each codec instance accesses the corresponding memory segment using indirect addressing based on at least one index register, the at least one index register being set for each codec instance to modify addressing of
20 variables for that codec instance to the corresponding memory segment.
4. A method as claimed in claim 3, wherein the plurality of memory segments are contiguous in said second memory, and said at least one index register is set for each codec instance according to an offset based on the difference in address from a first of said memory
25 segments to the memory segment corresponding to that codec instance.
5. A method as claimed in any one of claims 1 to 4, wherein each of the memory segments in said second memory is the same size.



- 12 -

6. A method as claimed in claim 1, 2 3, or 4, wherein said first memory is provided with a plurality of instruction code programs for implementing different kinds of codecs, and wherein different codec instances may be selected from the different kinds of codec.
- 5 7. A method as claimed in claim 6, wherein each of the memory segments in said second memory is the same size, and the size of the memory segments is selected according to the maximum memory required by any of the plurality of different kinds of codec.
8. A data coding and/or decoding system in which a plurality of encoders and/or decoders
10 (codecs) are implemented using a single digital signal processing (DSP) device, comprising:
a digital signal processor (DSP);
a first memory coupled to the DSP and containing an instruction code program, the function of each codec being performed by the DSP, in use, according to said instruction code program;
- 15 a second memory coupled to the DSP and partitioned to include a plurality of separate memory segments;
wherein a plurality of codecs are implemented using the DSP by running said instruction code program in said first memory a plurality of times in re-entrant instances, and wherein each codec instance is provided access to a respective separate memory segment in said
20 second memory for storing data used in encoding/decoding a respective separate data stream.
9. A system as claimed in claim 8, including a third memory coupled to said DSP which is accessible by each of the codec instances for shared storage of temporary variables and data buffering in encoding/decoding said respective separate data streams.
- 25 10. A system as claimed in claim 8 or 9, wherein each codec instance accesses the corresponding memory segment using indirect addressing based on at least one index register, the at least one index register being set for each codec instance to modify addressing of variables for that codec instance to the corresponding memory segment.



- 13 -

11. A system as claimed in claim 10, wherein the plurality of memory segments are contiguous in said second memory, and said at least one index register is set for each codec instance according to an offset based on the difference in address from a first of said memory segments to the memory segment corresponding to that codec instance.

5

12. A system as claimed in any one of claims 8 to 11, wherein each of the memory segments in said second memory is the same size.

13. A system as claimed in any one of claims 8 to 11, wherein said first memory is provided
10 with a plurality of instruction code programs for implementing different kinds of codecs, and wherein different codec instances may be selected from the different kinds of codec.

14. A system as claimed in claim 13, wherein each of the memory segments in said second memory is the same size, and the size of the memory segments is selected according to the
15 maximum memory required by any of the plurality of different kinds of codec.

15. A method for implementing a plurality of encoders and/or decoders (codecs) using a single digital signal processing (DSP) device, wherein the function of each codec is performed by the DSP according to an instruction code program, the method comprising the steps of:
20 providing an instruction code program stored in a first memory for controlling the DSP to function as a codec;

providing a second memory including a plurality of memory segments;

implementing a plurality of codecs using the DSP by running said instruction code program in said first memory a plurality of times in re-entrant instances, wherein each codec
25 instance is provided access to a respective separate memory segment using an indirect addressing mode based on at least one index register in said second memory for storing data used in encoding/decoding a respective separate data stream.



- 14 -

16. A data coding and/or decoding system in which a plurality of encoders and/or decoders (codecs) are implemented using a single digital signal processing (DSP) device, comprising:

a digital signal processor (DSP);

a first memory coupled to the DSP and containing an instruction code program, the
5 function of each codec being performed by the DSP, in use, according to said instruction code program;

a second memory coupled to the DSP and partitioned to include a plurality of separate memory segments;

wherein a plurality of codecs are implemented using the DSP by running said instruction
10 code program in said first memory a plurality of times in re-entrant instances, and wherein each codec instance is provided access to a respective separate memory segment using an indirect addressing mode based on at least one index register in said second memory for storing data used in encoding/decoding a respective separate data stream.

15

