(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 653 443 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
03.05.2006 Bulletin 2006/18

(51) Int Cl.:
G10H 7/10 *(2006.01)*

(21) Application number: **04077975.3**

(22) Date of filing: **29.10.2004**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LI LU MC NL PL PT RO SE SI SK TR
Designated Extension States:
AL HR LT LV MK

(71) Applicant: **Silicon Ip Ltd.**
Dublin 8 (IE)

(72) Inventor: **McNally, Conor**
Dublin 8 (IE)

(74) Representative: **Lane, Cathal Michael et al**
c/o Tomkins & Co.
5 Dartmouth Road
Dublin 6 (IE)

(54) Polyphonic sound synthesizer

(57) The invention relates to a polyphonic synthesizer and a method of synthesizing multiple sounds of multiple timbres. In particular the invention provides a polyphonic synthesiser capable of implementing all synthesis of multiple sounds efficiently in the frequency-domain. The invention provides a synthesiser and method of synthesizing multiple sounds of multiple timbres using a polyphonic synthesizer comprising the steps of processing each sound by applying a set of processing parameters, said processing parameters related to characteristics of each sound, accumulating processed information of each sound in a storage element; and executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a transform output to synthesize multiple sounds.

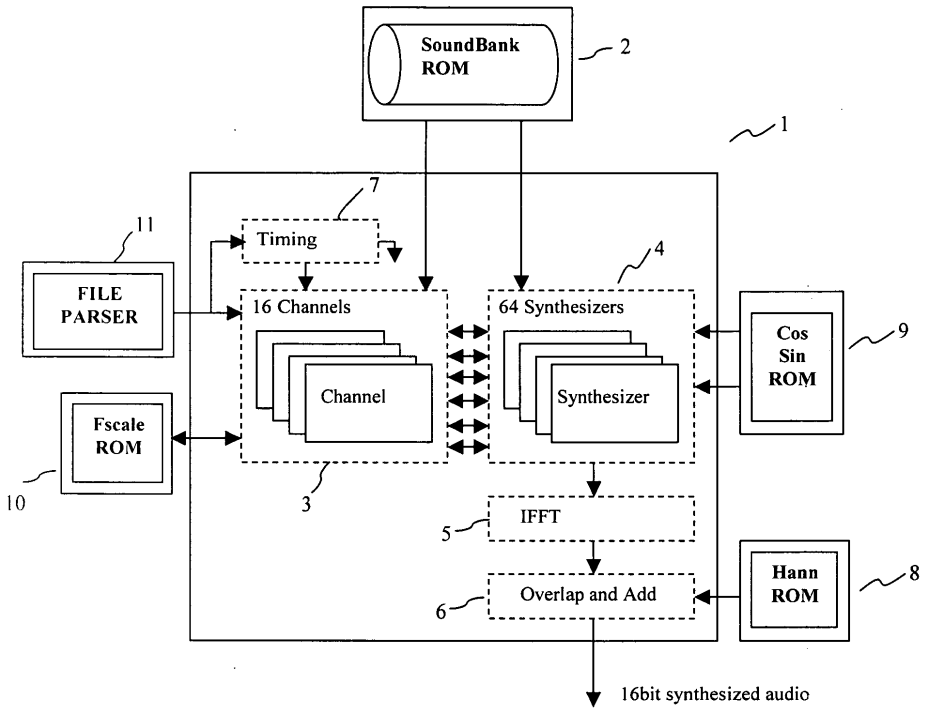

Figure 1

EP 1 653 443 A1

Description

Field of the Invention

[0001] The invention relates to a polyphonic synthesizer and a method of synthesizing multiple sounds of multiple timbres. In particular the invention provides a polyphonic synthesiser capable of implementing all synthesis of multiple sounds efficiently in the frequency domain.

Background to the Invention

[0002] Generally, polyphonic synthesisers are available in many forms, from your mobile phone ring-tone player to a professional keyboard synthesizer. Most polyphonic Synthesizers use the MIDI protocol, http://www.midi.org, as a means of representing and reading a musical composition. Most polyphonic synthesizers use time domain FM or Wavetable synthesis (or a combination of both) as the main synthesis technique. FM synthesis is a method of approximating an instrument's spectral content but in general it will sound synthesized or unrealistic. Wavetable synthesizers store samples of real instruments and then play these samples back during synthesis to give a more realistic sound.

[0003] A basic requirement for a polyphonic synthesizer is a Soundbank that generally stores information required by the synthesizer to reproduce the instruments or sounds it is required to synthesize. The synthesizer should also be capable of synthesizing multiple sounds of multiple timbres simultaneously. The number of sounds that are played at any one time denotes the degree of polyphony. The volume and frequency of each of the sounds should be adjustable at any time during the playback of the sounds. The synthesizer should be able to respond to 'Note On' and 'Note Off' commands in a way that relates to the instrument that is being played. A phase vocoder approach may be used in synthesising sounds. The phase vocoder normally utilises a known technique that uses frequency-domain transformations to implement a variety of audio effects such as time and pitch scaling of sounds. These known techniques are described in 'The Phase Vocoder: Theory and Practice' Organised Sound 2, Cambridge University Press, 1997, Fischman R.

[0004] The Phase Vocoder generally uses a method of pitch scale modification to the notes or sounds through a combination of time scaling and sample rate conversion to synthesize the notes. A typical method of modifying the pitch scale of a note is described in 'Improved Phase Vocoder Time Scale Modification of Audio', IEEE, LaRoche J, Vol. 7, No. 3, May 1999. For example to raise the pitch of a sound by a factor of 2, the sound would be time-stretched by a factor of 2, through frequency domain transformations. Then the output of the inverse Fast Fourier Transform (IFFT) would be re-sampled at half the sample rate. To implement a polyphonic synthesizer using known traditional techniques a separate IFFT would be required for every note with different pitch scaling factor because the time-stretch and re-sampling factors vary with pitch scale. However a problem with having a separate IFFT for each different pitch-scaling factor is that the polyphonic synthesizer is very inefficient in terms of computational complexity. A method of Pitch Scaling in the frequency domain without the need for re-sampling in the time domain is detailed in a publication 'Pitch Scaling Using the Fourier Transform', 1999, Sprenger S M, http://www.dspdimension.com/html/pscalestft.htm. However this technique is only described for pitch scaling of single sounds.

[0005] Various solutions have been proposed to introduce the idea of using a phase vocoder for implementing a polyphonic synthesizer. Patent number US 5,686,683 B1, entitled 'Inverse Transform Narrowband / Broadband Sound Synthesizer' discloses a variety of broadband features to synthesise a more complex sound. These features include numerous different frequency domain noise generation, filtering, interpolation techniques that can be used to generate a more complex sound. The main problem with the operation of this US patent is that each Voice (i.e. an instrument) of a polyphonic sound requires an individual inverse Fourier Transform. The other Voices in the Polyphonic sound require an additional IFFT for each voice. The output of each IFFT for each voice is then blended in time to give the overall polyphonic sound. This multiple transform operation application is computationally very intensive and technically difficult to achieve. The application of synthesizing the notes is thus restricted by the multiple transform operation.

[0006] Another US patent, No: US 5,401,897 B1, and entitled 'Sound Synthesis Process' discloses a sound synthesis process by means of the phase vocoder synthesis process. This patent also discloses a means to add a noise component to the constructed spectrum before the IFFT. The operation of this US patent is limited in terms of synthesis of multiple sounds as the process of inverse Fourier transform has to be repeated for each sound. Then each sound is super-imposed in the time-domain to synthesize the total sound.

Object of the Invention

[0007] It is therefore an object of the present invention to provide a polyphonic sound synthesizer and method of synthesising multiple notes which overcomes the above mentioned problems.

Summary of the Invention

[0008] According to the present invention there is provided a method of synthesizing multiple sounds of multiple timbres comprising the steps of:

processing each sound by applying a set of processing parameters, said processing parameters related to characteristics of each sound;
accumulating processed information of each sound in a storage element; and
executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a
transform output to synthesize multiple sounds.

[0009] One of the primary advantages of the present invention is that when all the sounds are processed individually and accumulated in the storage element or a buffer for a particular time frame a single transform is only required to be executed on all of the accumulated processed sounds for a particular time frame in the buffer to provide a real value output to synthesise multiple sounds.

[0010] In one embodiment the single transform operation is a single inverse Fast Fourier Transform (IFFT).

[0011] Preferably, the step of specifying each sound in terms of a frequency domain representation consisting of a series of time overlapped frames in combination with said set of processing parameters relating to the characteristics of at least one sound is carried out. An advantage of using the frequency domain is that before executing said inverse transform operation the invention can carry out additional processing on the sounds in the frequency domain representation. The additional processing on the sounds comprises one or more of the following: reverberation, headphone externalisation, positional audio, environmental modelling, stereo widening, equalisation and/or bass boost. This has the advantage that the invention can perform additional processing extremely efficiently and with higher quality sound synthesis.

[0012] Ideally the invention carries out the steps of reconstructing the time sampled representation of the sound by:

Optionally shaping the transform output by multiplying said transform by a window function, for each frame; and overlapping and adding together the windowed frames to provide a sampled time
domain representation of multiple sounds.

[0013] Suitably, the step of pitch scaling at least one sound stored in a sound bank to any sound in a key range associated with at least one timbre is carried out. Alternatively, the step of frequency scaling at least one sound stored in said sound bank to a different sound frequency is carried out.

[0014] Frequency scaling at least one sound stored in said sound bank to a different sound frequency may be carried out to implement at least one of the following: pitch bend, legato and/or portamento.

[0015] In another embodiment the step of calculating an envelope scale factor or scale factors on a per frame basis and applying at least one scale factor to at least one processing parameter to generate an envelope scaling factor or scaling factors for a frame is carried out.

[0016] Preferably the invention carries out the step of combining the volume of each sound for a frame with the envelope scale factor or scale factors to provide a total envelope scale factor.

[0017] Ideally the processing of a plurality of sounds is carried out in parallel with at least one of said applied processing parameters.

[0018] In another embodiment the invention carries out the step of sharing the processing during a time frame for simultaneous sounds of different frequency for the same timbre.

[0019] The accumulated processed information for all sounds for each frame may be stored in frequency bins in said storage element and a scaling factor may be applied to each bin before said single transform is executed.

[0020] Alternatively, accumulated processed information for all sounds for each frame is stored in frequency bins in said storage element and a scaling factor may be applied to each bin during execution of said single transform. In one embodiment, the scaling value can be determined from an absolute sum of the accumulated information in said storage element.

[0021] Desirably, the step of applying reverse scaling to said transform output for each time frame, applying a windowing function and overlapping and adding the resulting time frame with at least one preceding time frame is executed.

[0022] In a further embodiment there is provided a polyphonic synthesizer for synthesizing multiple sounds of multiple timbres comprising:

a processor for processing each sound by applying a set of processing parameters, said processing parameters related to characteristics of each sound;

a storage element for storing and accumulating processed information of each sound; and
a transform routine for executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a transform output to synthesize multiple sounds.

[0023] In another embodiment there is provided a synthesizer for synthesizing multiple sounds of multiple timbres comprising:

means for processing each sound by applying a set of processing
parameters, said processing parameters related to characteristics of each sound;
means for accumulating processed information of each sound in a storage element; and
means for executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a transform output to synthesize multiple sounds.

[0024] In a further embodiment there is provided a synthesizer for synthesizing multiple sounds of multiple timbres comprising:

a storage element;
a processor in communication with said storage element, said
processor applying a set of processing parameters to process a plurality of sounds, said processing parameters related to characteristics of each sound and said plurality of processed sounds being stored in said storage element; and
an inverse transformer in communication with said storage element,
said inverse transformer executing a single inverse transform operation on the plurality of processed sounds stored in said storage element,

wherein said single transform operation provides a transform output to synthesize multiple sounds.

[0025] In another embodiment there is provided a MIDI player comprising:

a synthesizer for synthesizing multiple sounds of multiple timbres comprising:

a storage element;
a processor in communication with said storage element, said processor applying a set of processing parameters to process a plurality of sounds, said processing parameters related to characteristics of each sound and said plurality of processed sounds being stored in said storage element; and
an inverse transformer in communication with said storage element, said inverse transformer executing a single inverse transform operation on the plurality of processed sounds stored in said storage element,

wherein said single transform operation provides a real transform output to synthesize multiple sounds.

[0026] There is also provided a computer and/or a computer program comprising program instructions for causing a computer to carry out the above method which may be embodied on a record medium, carrier signal or read-only memory.

Brief Description of the Drawings

[0027] The invention will be more clearly understood from the following description of an embodiment thereof, given by way of example only, with reference to the accompanying drawings, in which:-

Figure 1 is a top level system architecture for carrying out the present invention;
Figure 2 is a detailed channel functional diagram illustrated in Fig 1;
Figure 3 is a detailed synthesizer functional diagram illustrated in Fig 1;
Figure 4 is a diagram of an envelope generator according to another aspect of the present invention;
Figure 5 is a diagram of a storage element illustrated in Fig 1; and
Figure 6 illustrates a further aspect of the present invention.

Detailed Description of the Drawings

[0028] Referring to Figure 1 there is illustrated a system architecture for synthesizing multiple sounds on multiple timbres using a polyphonic synthesizer indicated generally by the reference numeral 1. A sound bank 2 can generate

a sound bank of sounds to store at least one note per timbre. In this specification a "sound" is meant to encompass any physical sound whether it be a musical note or a sound effect of any kind. The different sounds that a synthesiser can produce are sometimes called "patches", "programs", "algorithms" or "timbres". Programmable synthesisers commonly assign "program numbers" or "patch numbers" to each sound. For instance, a sound module might use patch number one for its acoustic piano sound, and patch number thirty six for its fretless bass sound. The association of all patch numbers to all sounds is called a patch map. A synthesiser can generally be regarded as multi-timbral if it can generate two or more different instrument sounds simultaneously. The same sound or note can be played on two different instruments and may sound completely different since each instrument has its own timbre. For the sake of clarity, the present description uses the terms sound and timbre in conjunction with the operation of the synthesizer, but should not be limited to these terms.

[0029] The sound bank 2 is in communication with a channel block 3, which drives the synthesis process, and a polyphonic synthesiser 4. The sound information stored in the sound bank 2 can be generated using a vocoder analysis technique. The polyphonic synthesiser 4 outputs synthesised sounds to a storage element 5, for example a buffer or a memory store. The system architecture also includes an overlap and add (OAA) block 6, a timing block 7, memory blocks 8, 9 & 10 and a file parser 11 the operation of each will be explained in more detail below in the description.

[0030] The sound bank 2 contains a frequency analysed timbre set to support timbres of a specified standard, for example the standard set in the Midi specification, http://www.midi.org. The File parser 11 decodes the score sheet which could be in the form of a MIDI file, to be synthesized by the channel block 3 and synthesiser 4. The file parser 11 outputs timed events detailing the sequence of instruments/notes/parameters to be used in synthesis. The Channel block 3, which drives the synthesis process can include a processor. Input events from the file parser 11, the timing block 7 and soundbank 2 trigger queuing and the setting up of the necessary parameters and modules needed to synthesize the requested sounds. The input events can be controlled directly by a user or input from a pre-programmed application or content file. In the embodiment shown the synthesizer 4 comprises sixty four Synthesizer modules to support sixty four note polyphony. Each synthesizer module is assigned sounds or notes to synthesize from one of a plurality of Channel Modules in the channel block 3. Each synthesizer will synthesize the sounds on a per frame basis, the length of a frame is determined by the timing module 7. Each sound is specified in terms of a frequency domain representation comprising of a series of time overlapped frames in combination with a set of processing parameters relating to the characteristics of at least one sound. The output frames, which are the processed information of each sound, from each synthesizer are accumulated into the storage element or buffer 5 for an individual frame. The buffer 5 handles frequency to time domain conversion by executing a single inverse Fast Fourier Transform (IFFT) on all the accumulated information. The IFFT operates on all the frequency information accumulated into the buffer 5 using a single transformation to provide a real output. The overlap and add (OAA) block 6 uses the real output of the IFFT to reconstruct the synthesized waveform.

[0031] Referring to Figure 2 illustrates the Channel block 3 in more detail, which drives the synthesis process. The Channel block 3 includes a Channel Driver 31, Channel Synthesizer Assignment 32, Channel Buffers 33, Instrument 34 and Remove Note block 35. The operation of the Channel Block 3 is common for both Melodic and Percussion instruments or timbres or sound effects. An example of the difference between the two timbres in the MIDI standard is that Percussion instruments events are only received on MIDI Channel 9 and Melodic instruments are received on all other 15 MIDI channels (i.e. 0-8 &10 -15).

[0032] In operation the Soundbank 2 is accessed with the requested program/sound number to return the Soundbank memory location for the requested instrument 34 to the channel block 4. The requested instrument 34 can be mapped to a different timbre internally in the Soundbank 2. This is used to reduce the number of timbres stored in the Soundbank 2. A minimum of thirty six instruments, which includes both melodic and percussion, are required by the Midi specification standard. The corresponding source note key range for each instrument and memory location of each stored sound is extracted from the Soundbank 2. Instrument 34 also calls a Header Info block to extract header information of each source sound. Each source sound or note has a header section, which contains the configuration parameters used for processing that are critical to drive the synthesis of that sound. The configuration parameters can include:

Note Number for attack frames
Looping frame information
Polynomial information for Envelope generation
Bin dumping positions

[0033] The Channel buffers 33 store the requested Note On numbers, sound velocities and source sound index values assigned by the Channel Driver 31. The Remove Note block 35 handles Note Off events by removing the corresponding note information from the channel buffers and from the assigned synthesizer. The Channel Synth Assign block 32, assigns the sounds set-up and queued in the Channel Buffers 33 to synthesizers within the synthesizer block 4. Up to four sounds that are frequency scaled from the same source sound can be grouped into a single synthesizer module.

Additional sounds for the same source sound are assigned to different synthesizers in groups of four. The synthesizer 4 is loaded with Channel number (i.e. instrument pointer) and a pointer to the source sound header information (i.e. which sound to frequency scale from). The synthesizer 4 is loaded with the sound numbers, sound velocities and frequency scaling values to synthesize.

[0034] Referring to Figure 3 the synthesizer 4 is illustrated in more detail. The synthesizer 4 will synthesize a new frame for each of the sounds that are assigned to it in the frequency domain representation and accumulate each sound frame into the storage element 5. Each synthesizer 40a-d can synthesize up to four sounds from the same source sounds, for example a chord of 4 simultaneous Piano sounds may be frequency scaled from the same source sound. Sounds are assigned to synthesizer modules 40a-d in the Channel block 3. When a Note On command is received the synthesizer 4 is loaded with information of sound number, velocity, frequency scale value, frame count number, source note header pointer information. The synthesizer 4 incorporates the main synthesizer functions block, a memory buffer 41, a memory access controller 42 and an Envelope generator 43. The Envelope generator 43 calculates the Envelope magnitude on a per frame basis and is combined with sound volumes to give the total sound Envelope magnitude. If the Envelope magnitude has decayed to zero there is no need to synthesize a frame for the currently selected synthesizer. Otherwise a new frame is synthesized for the sounds used by the current synthesizer. The memory access controller is used to update the memory pointers used to read Bin, Mag, and Phase values from the Soundbank 2 for the current frame.

[0035] The basic synthesis procedure can be optimised for various different synthesis requirements. In some applications no frequency scaling is required as is the case with percussion instruments. To reduce processing load a synthesis algorithm is broken out into separate functions, each facilitating an optimised synthesis path for each of the possible synthesis scenarios. Each synthesis function is a variation of the same synthesis algorithm.

[0036] Referring to Figure 4 the Envelope Generator block 43 handles generation of the waveform Envelope during synthesis. An Envelope scale factor is calculated on a per frame basis and applied to the note velocity values, assigned to the synthesizer by the Channel 3, to give the total Envelope scaling factor. The Envelope generator 43 employs a polynomial generator 431 capable of generating linear, quadratic or cubic curves and Envelope generation control 432. The polynomial generator 431 handles generation of the polynomial,

$$\text{Eqn. 1}: \qquad Poly = P_o x^3 + P_1 x^2 + P_2 x^1 + P_3$$

[0037] The polynomial coefficients (P_o, P_2, P_2, P_3) are read by the Channel 3 on each new program change command. Channel 3 invokes initialisation of the new polynomial coefficients when a program change is received. The polynomial is calculated on a per frame basis. The Memory Access controller 42 handles control of the Soundbank 2 memory location pointers. These pointers point to the Soundbank memory locations that correspond to the position where the Bin, Magnitude and Phase values for the current frame are stored. The Memory Access control 42 pointers are updated on a per frame basis.

[0038] Another method of generating the envelope is by using Attack, Decay, Sustain and Release method, other wise known as an ADSR Envelope method. The Attack phase is the initial striking of the sound where the most transients occur and the most rich dynamics of the sound occur. The Decay phase immediately follows the Attack phase where the sound reduces in amplitude to a steady state called the Sustain phase. The Sustain phase is generally simpler in terms of frequency content and decays gently over a period of time. This phase lends itself well to looping frames and just applying a decaying envelope in order to save on memory. Finally, when the sound is released, it decays to zero over a period of time. Each of these phases of the envelope can be controlled independently in terms of their duration and shape. Other envelopes could also be used but the ADSR or variations are generally accepted to be the best for musical sounds, for example DAHDSR (Delay, Attack, Hold, Decay, Sustain, Release) envelope.

[0039] Referring to Figure 5 illustrates the operation of the storage element 5 in more detail. The storage element 5 handles storage and conversion of the frequency-domain synthesis frames into Time domain synthesis frames. The storage element 5 comprises Real and Imaginary IFFT buffers 51 and 52, an IFFT scaling block 53 and the IFFT routine 54. The IFFT buffers 51 and 52 comprise of two (Real and Imaginary) 1024x2byte arrays which act as a transformer to execute the inverse transform. During synthesis of the frames, before the IFFT routine, the lower half (513 Bins) of the buffers are used. At the start of a new synthesis frame the lower 513 bins of the IFFT buffers are reset to zero. During the synthesis frame the processed bin information of each note frame is accumulated into the IFFT buffer 51 and 52. After the last sound frame of the current synthesis frame is accumulated into the IFFT buffers 51 and 52, the buffers are scaled by the scaling block 53 to avoid overflow in the IFFT calculation. The scaling factor is determined by using the result of the sum of the absolute value of the IFFT buffers to drive threshold detectors, if the sum is determined to exceed

a certain threshold. The scaling factor will be 2^n, where n = 0, 1 or 2. The scaling factor can be applied with one of two methods:

1. Applied to all bins of the IFFT buffers before input to the IFFT calculation.
2. If the IFFT routine supports scaling as part of the underlying calculation where a scaling of 2 can be applied at each radix of the calculation. Then scaling can be applied within the IFFT calculation. This method will give less quantization noise than scaling at the input but it will also require more processing to support the scaling within the IFFT

[0040] The present invention can support both options but the first is preferable for optimising computational requirements as the improvement in sound quality offered by the second is not perceivable. This is mainly due to the fact that scaling is only used during loud (high polyphony) sections of the synthesized sound and small quantization errors during a loud section are not perceivable. The effect of scaling is then removed from the output of the single IFFT calculation. This is done in the OAA block 7. It will be appreciated that the use of a single transform only, improves the synthesising process as only a single calculation is needed.

[0041] The IFFT routine is a standard 1024-point (16Bit) IFFT. The IFFT does not require any overflow checking/ scaling support, although IFFT routines that incorporate such features may also be utilized. During synthesis only the bins up to N/2 +1 (i.e. 513) are used. This is because the FFT has conjugate symmetry about the point N/2 +1. Some IFFT routines copy the conjugate information internally and some do not. For the case where the IFFT routine does not load the conjugate information internally the upper half of the IFFT buffer is loaded with the conjugate of the lower half using the following equation:

$$\text{Eqn. 2:}\quad x(\tfrac{N}{2}+1+m) = conj(x(\tfrac{N}{2}+1-m)),\ \text{where m} = 0,1,2,\ldots\tfrac{N}{2}$$

[0042] Referring to Figure 6 the overlap and add (OAA) block 6 uses the real output of the buffer 5 to reconstruct the synthesized waveform. The OAA block 6 contains scaling and windowing functions 61 and an OAA buffer 62. The scaling that was applied to the input of the IFFT is removed from the real output from the IFFT. Before overlap and adding the frames a Hanning window from the memory block 8 is first applied to the IFFT frame. The Hanning window is stored as a look up table in the memory block 8. After scaling and window tapering the resulting frame is overlapped and added to the preceding OAA frames, where the overlap is equal to Hop samples (i.e.256). The lower Hop samples are output as the next frame of synthesized 16-bit audio.

[0043] Only the real part of the IFFT output is required to synthesize the sound. For memory efficiency it is best to use an in-place IFFT which copies the output result to the input IFFT vector/array. Additionally in certain circumstances the imaginary and/or complex part of the IFFT may be used in order to synthesise the multiple sounds.

[0044] It will be appreciated that the functionality of the Channel block for Percussion instruments differs slightly from that of Melodic instruments. This is because Program Change and Note Off commands are not used on with percussion instruments and also because they do not require frequency scaling.

[0045] It will be further appreciated that the Polyphonic Synthesizer described in this specification can be used as a midi player, however the invention is not restricted to the midi specification and can be used for general polyphonic sound synthesis. It will be appreciated that the application is not limited to be applied to the Midi player standard and can be used in any other synthesizing format, for example sound formats besides MIDI include: SP-MIDI, DLS (Downloadable Sounds), SMAF, iMelody and XMF (eXtensible Music Format). In this specification a polyphonic synthesizer/ midi-player that is capable of implementing all synthesis in the frequency domain, with use of the Phase Vocoder, can be applied to the present invention. The term polyphonic refers to the ability of a synthesiser to play more than one sound at a time. The terms storage element and buffer are used interchangeably and should be interpreted broadly.

[0046] Other applications of the invention include implementation of the synthesizer in a stereo system or multi-speaker systems in addition to mono systems.

[0047] The embodiments in the invention described with reference to the drawings comprise a computer apparatus and/or processes performed in a computer apparatus. However, the invention also extends to computer programs, particularly computer programs stored on or in a carrier adapted to bring the invention into practice. The program may be in the form of source code, object code, or a code intermediate source and object code, such as in partially compiled form or in any other form suitable for use in the implementation of the method according to the invention. The carrier may comprise a storage medium such as ROM, e.g. CD ROM, or magnetic recording medium, e.g. a floppy disk or hard disk. The carrier may be an electrical or optical signal which may be transmitted via an electrical or an optical cable or by radio or other means.

[0048] The words "comprises/comprising" and the words "having/including" when used herein with reference to the present invention are used to specify the presence of stated features, integers, steps or components but does not preclude the presence or addition of one or more other features, integers, steps, components or groups thereof.

[0049] It is appreciated that certain features of the invention, which are, for clarity, described in the context of separate embodiments, may also be provided in combination in a single embodiment. Conversely, various features of the invention which are, for brevity, described in the context of a single embodiment, may also be provided separately or in any suitable sub-combination.

[0050] The invention is not limited to the embodiments hereinbefore described but may be varied in both construction and detail.

Claims

1. A method of synthesizing multiple sounds of multiple timbres using a polyphonic synthesizer comprising the steps of:

   processing each sound by applying a set of processing parameters, said processing parameters related to characteristics of each sound;
   accumulating processed information of each sound in a storage element; and
   executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a transform output to synthesize multiple sounds.

2. The method of claim 1 wherein said single inverse transform operation is a single inverse Fast Fourier Transform (IFFT).

3. The method of claim 1 or 2 comprising the step of specifying each sound in terms of a frequency domain representation consisting of a series of time overlapped frames in combination with said set of processing parameters relating to the characteristics of at least one sound.

4. The method as claimed in any preceding claim comprising the step of reconstructing a time sampled representation of the sound by :

   overlapping and adding together the frames to provide a sampled time domain representation of multiple sounds.

5. The method of any claims 1 to 3 comprising the step of reconstructing a time sampled representation of the sound by:

   shaping the real transform output by multiplying said transform by a window function, for each frame.

6. The method of claim 5 comprising the further step of:

   overlapping and adding together the windowed frames to provide the sampled time domain representation of multiple sounds.

7. The method of claim 5 or 6 wherein said window function is a Hanning window function.

8. The method of any preceding claim comprising the step of pitch scaling at least one sound to any sound in a key range associated with at least one timbre.

9. The method of any preceding claim comprising the step of frequency scaling at least one sound to a different sound frequency.

10. The method of any of claims 1 to 9 comprising the step of frequency scaling at least one sound stored in said sound bank to a different sound frequency to implement at least one of the following: pitch bend, legato and/or portamento.

11. The method of any preceding claim comprising the step of calculating an envelope scale factor on a per frame basis and applying said calculation to at least one processing parameter to generate an envelope scale factor for a frame.

12. The method as claimed in any preceding claim comprising the step of combining the volume of each sound for a frame with the envelope scale factor to provide a total envelope scale factor.

13. The method as claimed in any preceding claim comprising the step of processing a plurality of sounds in parallel with said applied processing parameters.

14. The method as claimed in any preceding claim comprising the step of sharing the processing during a time frame for simultaneous sounds of different frequency for the same timbre.

15. The method as claimed in any preceding claim wherein the accumulated processed information for each sound are stored in separate bins in said storage element and applying a scaling factor to each bin before said single transform is executed.

16. The method as claimed in any of claims 1 to 14 wherein accumulated processed information for each sound are stored in separate bins in said storage element and applying a scaling factor to each bin during execution of said single transform.

17. The method as claimed in claims 15 or 16 comprising the step of determining the scaling value from an absolute sum of the accumulated information in said storage element.

18. The method as claimed in preceding claim comprising before executing said inverse transform operation carrying out additional processing on said sounds in the frequency domain representation.

19. The method as claimed in claim 18 wherein said additional processing on said sounds comprises one or more of the following: reverberation, headphone externalisation, positional audio, environmental modelling, stereo widening, equalisation and/or bass boost.

20. The method as claimed in any preceding claim comprising the step of applying to said real transform reverse scaling to said transform output for each time frame, applying said window function and overlapping and adding the resulting time frame with at least one preceding time frame.

21. A computer program comprising program instructions for causing a computer to perform the method of any one of claims 1 to 20.

22. A computer program as claimed in claim 21 embodied on a record medium.

23. A computer program as claimed in claim 21 embodied on a carrier signal.

24. A computer program as claimed in claim 21 embodied on a read-only memory.

25. A polyphonic synthesizer for synthesizing multiple sounds of multiple timbres comprising:

a processor for processing each sound by applying a set of processing parameters, said processing parameters related to characteristics of each sound;
a storage element for storing and accumulating processed information of each sound; and
a transform routine for executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a transform output to synthesize multiple sounds.

26. A synthesizer for synthesizing multiple sounds of multiple timbres comprising:

means for processing each sound by applying a set of processing
parameters, said processing parameters related to characteristics of each sound;
means for accumulating processed information of each sound in a storage element; and
means for executing a single inverse transform operation on the accumulated information stored in said storage element wherein said single transform operation provides a transform output to synthesize multiple sounds.

27. A synthesizer for synthesizing multiple sounds of multiple timbres comprising:

a storage element;
a processor in communication with said storage element, said processor applying a set of processing parameters

to process a plurality of sounds, said processing parameters related to characteristics of each sound and said plurality of processed sounds being stored in said storage element; and
an inverse transformer in communication with said storage element, said inverse transformer executing a single inverse transform operation on the plurality of processed sounds stored in said storage element,

wherein said single transform operation provides a transform output to synthesize multiple sounds.

28. A MIDI player comprising:

a synthesizer for synthesizing multiple sounds of multiple timbres comprising:

a storage element;
a processor in communication with said storage element, said processor applying a set of processing parameters to process a plurality of sounds, said processing parameters related to characteristics of each sound and said plurality of processed sounds being stored in said storage element; and
an inverse transformer in communication with said storage element, said inverse transformer executing a single inverse transform operation on the plurality of processed sounds stored in said storage element,

wherein said single transform operation provides a transform output to synthesize multiple sounds.

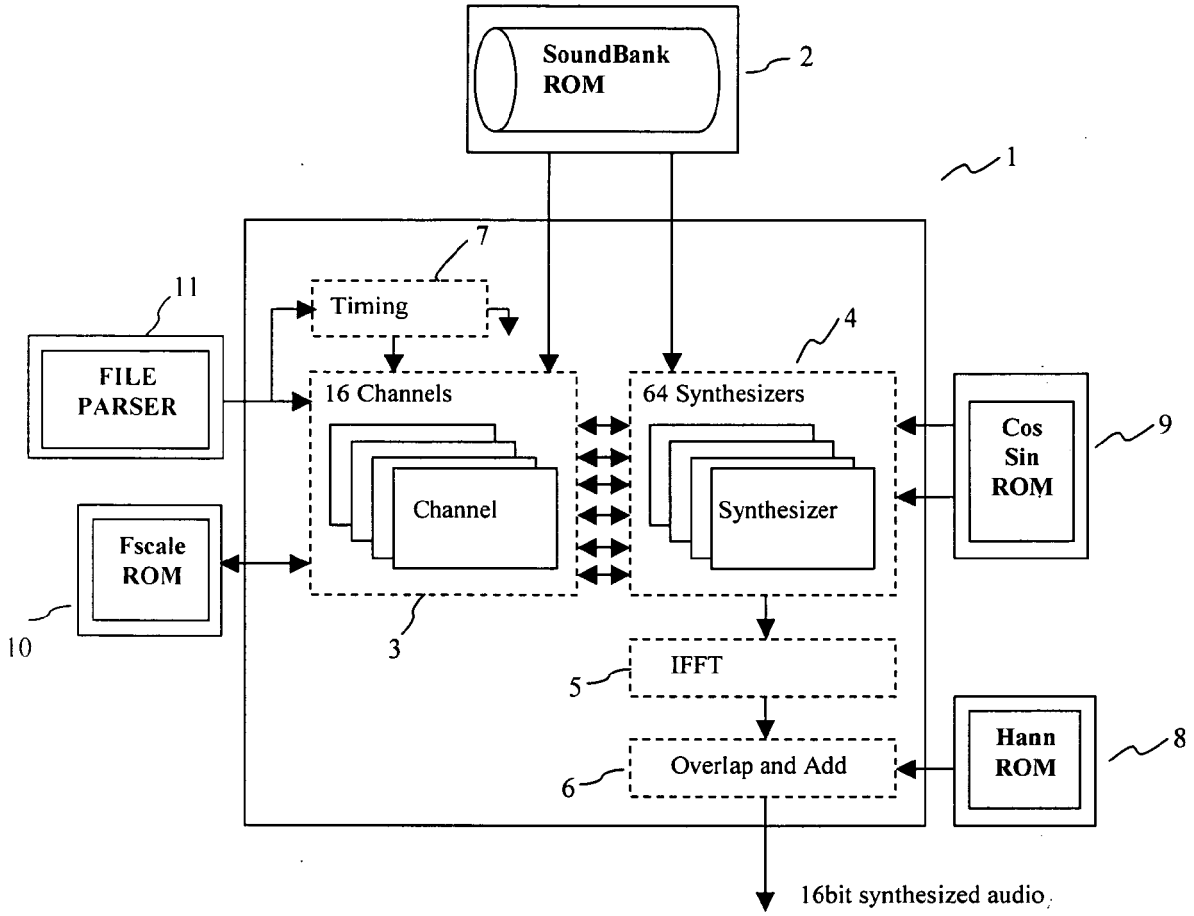
SoundBank ROM
2
1
7
11
Timing
4
FILE PARSER
16 Channels
64 Synthesizers
Cos Sin ROM
9
Channel
Synthesizer
Fscale ROM
10
3
IFFT
5
Hann ROM
8
Overlap and Add
6
16bit synthesized audio

Figure 1

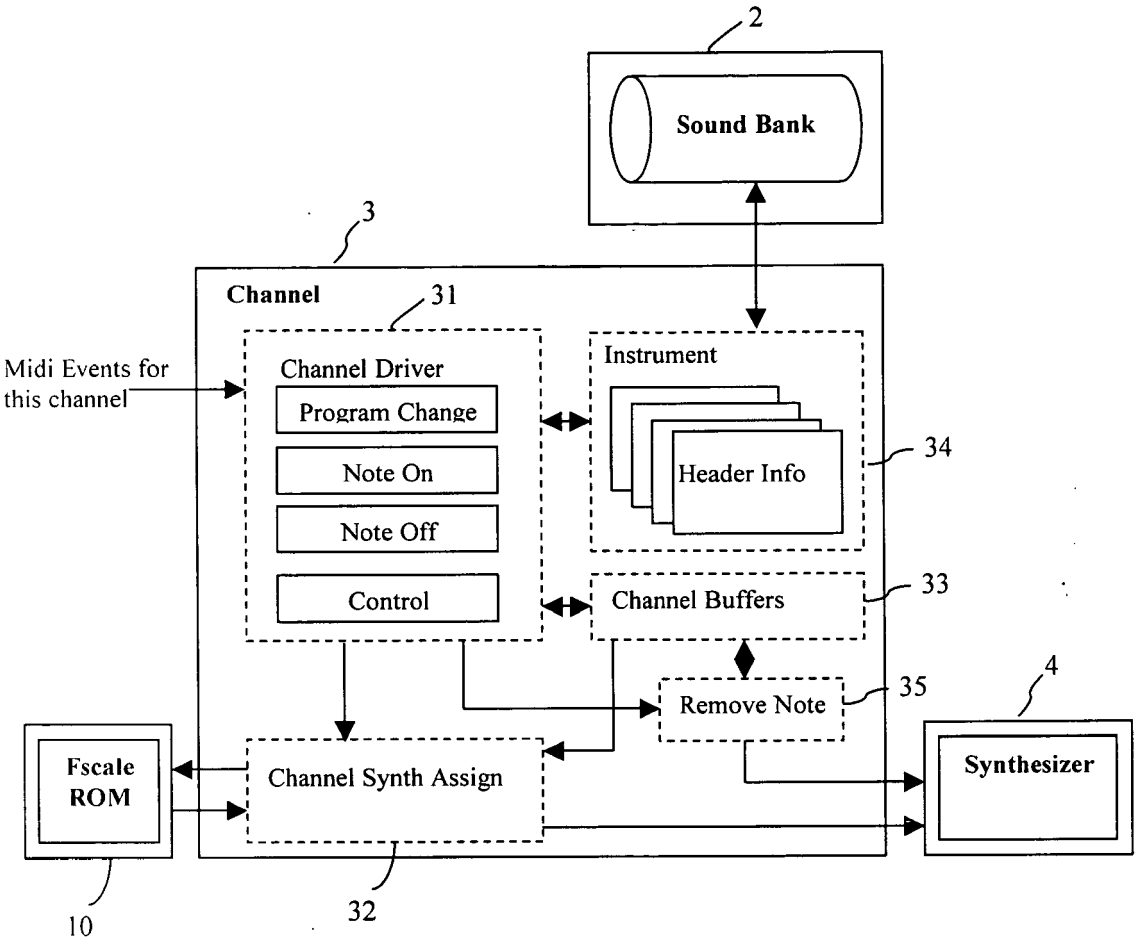
2
Sound Bank
3
Channel
31
Midi Events for this channel
Channel Driver
Program Change
Note On
Note Off
Control
Instrument
Header Info
34
Channel Buffers
33
Remove Note
35
4
Synthesizer
Fscale ROM
Channel Synth Assign
10
32

Figure 2

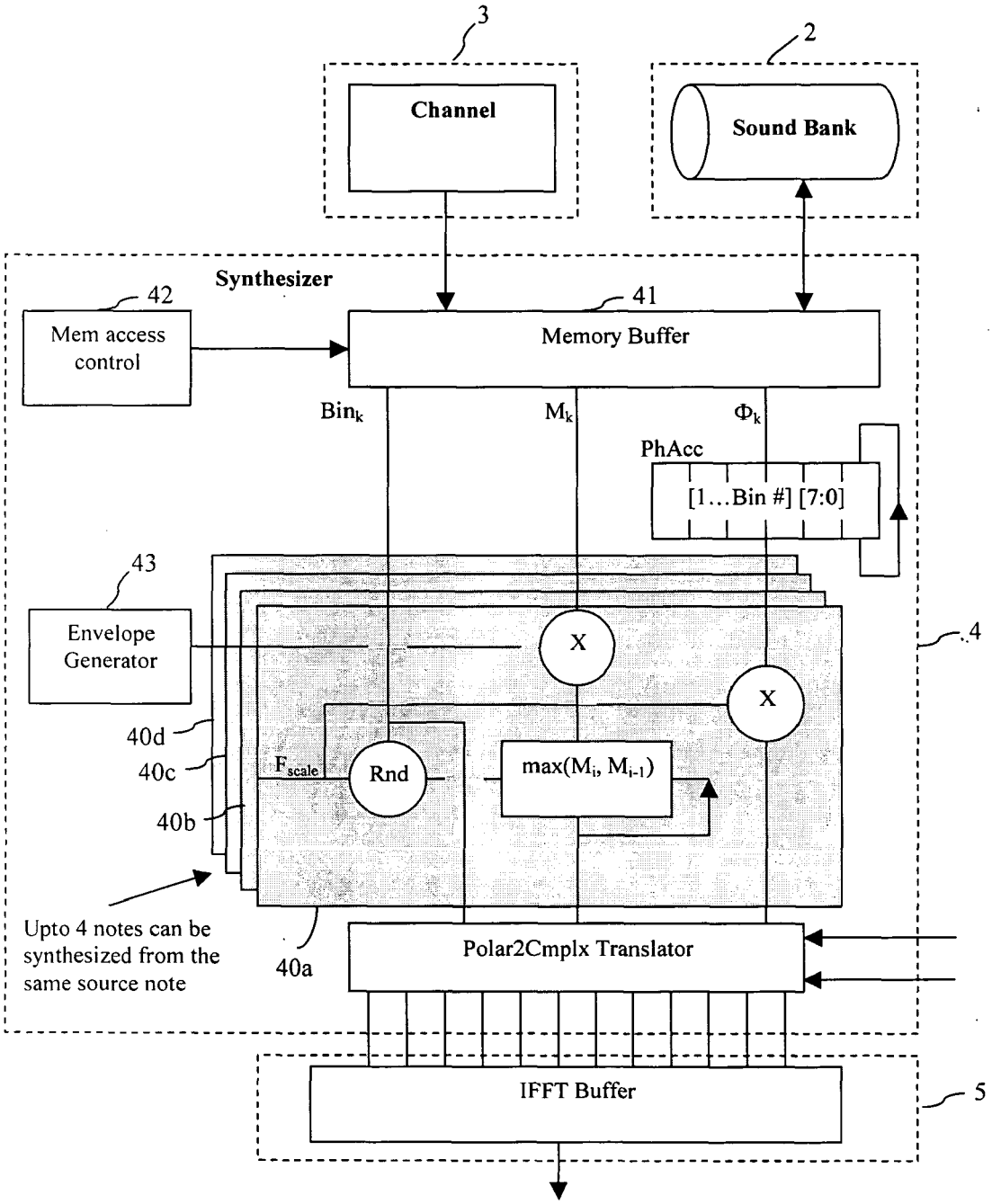
3
Channel
2
Sound Bank
Synthesizer
42
Mem access control
41
Memory Buffer
Bin$_k$
M$_k$
Φ$_k$
PhAcc
[1...Bin #] [7:0]
43
Envelope Generator
X
X
4
40d
40c
40b
F$_{scale}$
Rnd
max(M$_i$, M$_{i-1}$)
Upto 4 notes can be synthesized from the same source note
40a
Polar2Cmplx Translator
IFFT Buffer
5

Figure 3

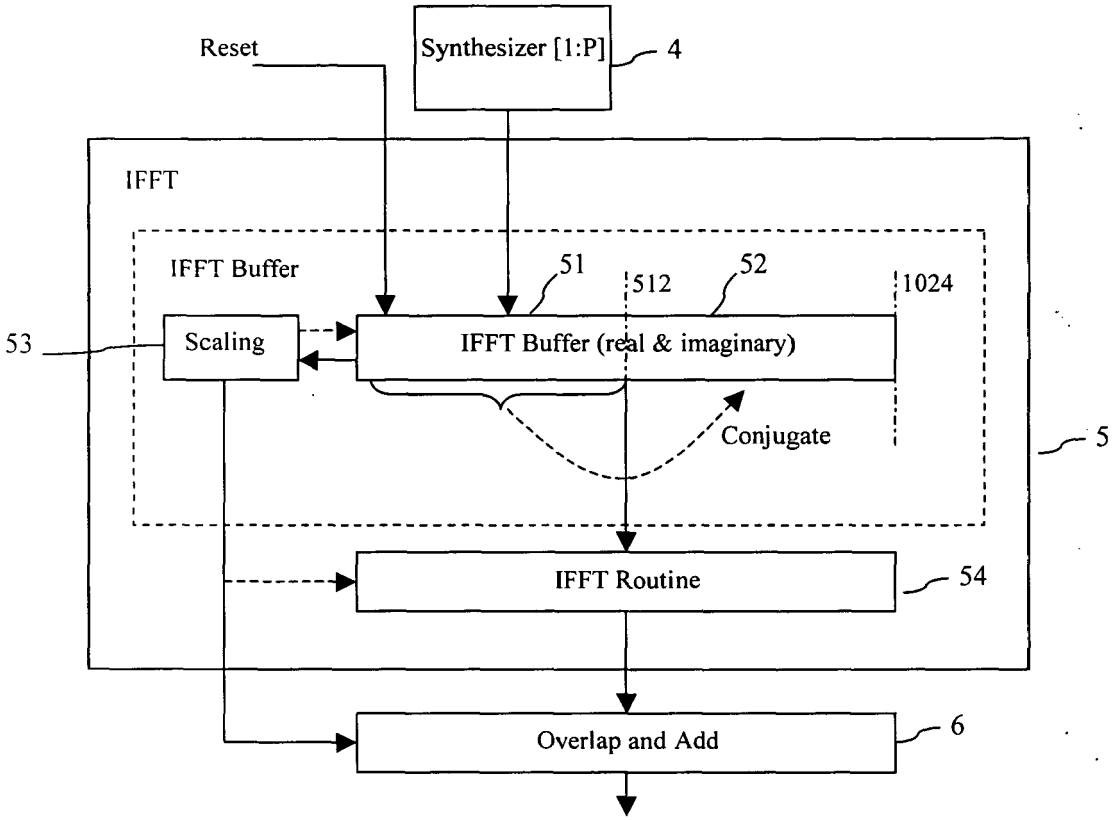

Figure 5

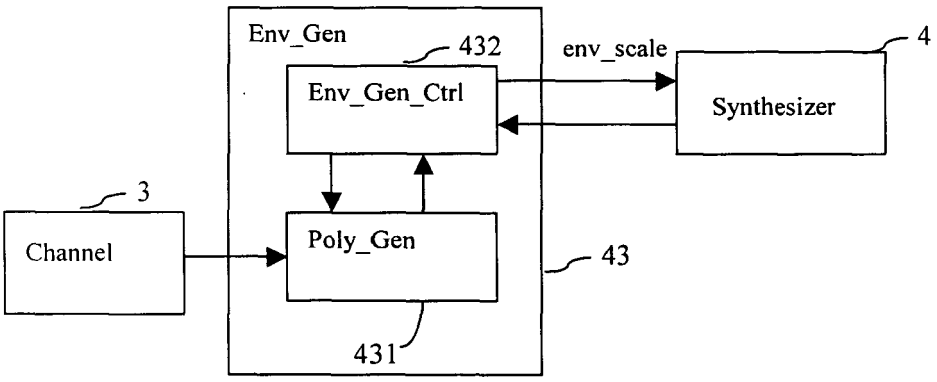

Figure 4

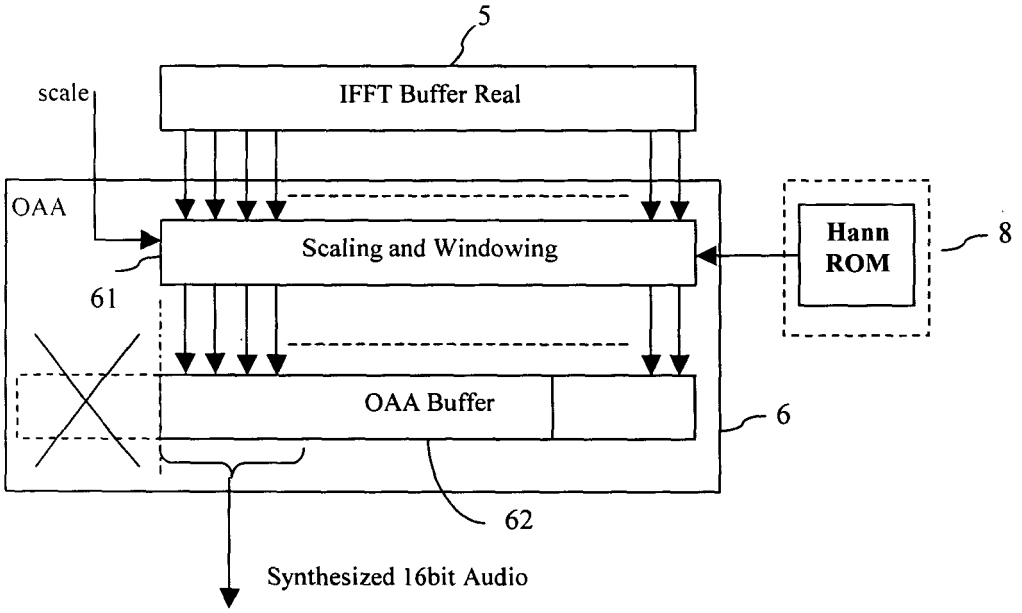
5
IFFT Buffer Real
scale
OAA
Scaling and Windowing
Hann
ROM
8
61
OAA Buffer
6
62
Synthesized 16bit Audio

Figure 6

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 04 07 7975

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	GEORGE B ET AL: "Fixed frame rate ABS/OLA sinusoidal modeling applied to polyphonic music synthesis" MULTIMEDIA SIGNAL PROCESSING, 1997., IEEE FIRST WORKSHOP ON PRINCETON, NJ, USA 23-25 JUNE 1997, NEW YORK, NY, USA,IEEE, US, 23 June 1997 (1997-06-23), pages 59-64, XP010233897 ISBN: 0-7803-3780-8 * the whole document *	1-28	G10H7/10
A	WO 00/07114 A (KONINKL PHILIPS ELECTRONICS NV) 10 February 2000 (2000-02-10) * abstract * * page 1, line 10 - line 28 * * page 2, line 8 - line 16 * & DATABASE WPI Derwent Publications Ltd., London, GB; AN 2000-183286 * abstract *	15-17	
A	US 6 137 839 A (GATHERER ALAN ET AL) 24 October 2000 (2000-10-24) * abstract * & DATABASE WPI Derwent Publications Ltd., London, GB; AN 2001-101357 * abstract *	15-17	TECHNICAL FIELDS SEARCHED (Int.Cl.7) G10H
D,A	US 5 686 683 A (FREED ADRIAN) 11 November 1997 (1997-11-11) * abstract * * column 1, line 5 - line 60; figures 3,4 * * column 3, line 55 - line 10 *	1-9, 25-28	

-/--

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	13 January 2005	Lecointe, M

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 04 07 7975

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
D,A	US 5 401 897 A (RODET XAVIER ET AL) 28 March 1995 (1995-03-28) * column 1, line 22 - line 54 * * abstract * * figure 1 * * column 2, line 56 - column 3, line 43 * -----	1-9, 25-28	
A	X. RODET AND P. DEPALLE: "SPECTRAL ENVELOPES AND INVERSE FFT SYNTHESIS" JOURNAL OF THE AUDIO ENGINEERING SOCIETY, 4 October 1992 (1992-10-04), XP002313030 * abstract * * pages 1-2-3 * -----	1,25-28	
			TECHNICAL FIELDS SEARCHED (Int.Cl.7)

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	13 January 2005	Lecointe, M

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

2

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 04 07 7975

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

13-01-2005

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 0007114 A	10-02-2000	WO 0007114 A1	10-02-2000
US 6137839 A	24-10-2000	CA 2217073 A1	01-04-1999
		EP 0806852 A2	12-11-1997
		JP 10075279 A	17-03-1998
		US 6021167 A	01-02-2000
		US 6055268 A	25-04-2000
		US 6002722 A	14-12-1999
		US 5910970 A	08-06-1999
		US 5987061 A	16-11-1999
		US 6021158 A	01-02-2000
		US 5970088 A	19-10-1999
		US 5999563 A	07-12-1999
		US 6038251 A	14-03-2000
		US 6044107 A	28-03-2000
US 5686683 A	11-11-1997	AU 7518196 A	15-05-1997
		DE 69629934 D1	16-10-2003
		DE 69629934 T2	22-07-2004
		EP 0860003 A1	26-08-1998
		JP 11513821 T	24-11-1999
		WO 9715915 A1	01-05-1997
US 5401897 A	28-03-1995	FR 2679689 A1	29-01-1993
		WO 9303478 A1	18-02-1993
		JP 6502023 T	03-03-1994
		JP 3098031 B2	10-10-2000

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82