Samples

This Chapter explains how to make use of sampled sound to enhance your programs.

Modern computers are able to store sound frequencies in the form of digits. Your computer is a digital sound synthesizer, and AOZ is ready, willing and able to harness its power. There are many digital sound samplers on the market, ready to be plugged into your computer for grabbing sound samples off CD, cassette, radio and microphone.

Bank number 5 is usually held as the default sample bank.

Playing a sound sample

SAM PLAY

instruction: play a sound sample from the sample bank

```
Sam Play sample number
Sam Play voice,sample number
Sam Play voice,sample number, frequency
```

The SAM PLAY command is used to play a digital sound sample through your audio system. Simply define the number of the required sample held in the bank. There is no limit to the number of samples that can be stored, other than available memory.

There are two optional parameters that can also be given. A voice parameter can be placed immediately after the SAM PLAY command, in front of the sample number. This is a bitmap containing a list of the voices the sample will use. There is one bit for each of the four available voices, and to play a voice the relevant bit should be set to 1, as explained in the last chapter. The other parameter can be given after the sample number, and this governs the frequency of the sound. The frequency parameter sets the speed at which the sample will be played back, and the setting is given in Hertz. This is a professional standard of measurement, but as a rule of thumb, a rate of 4000 is acceptable for simple sound effects, with 10000 for recognisable speech. By changing this playback rate, the sample
pitch can be adjusted over a very wide range, allowing a single sample to generate many different sounds.

The following example loads up a bank of ready-made samples stored on your TODO - "AOZ Examples" package, and allows you to play them in random order:
TODO - "Update AMOS with AOZ below"

```
Load "AMOSPro_Examples:Samples/Mixture.Abk",5
Sam Bank 5
Curs Off : Cls 0: Paper 0
Locate 0,10
Centre "Press a key from A to J"
Do
    A$=Inkey$
    A=Asc(A$)
    If A>96 And A<107
        Sam Play A-96
    End If
Loop
```

You can try playing the keys rapidly, like a miniature drum kit, as well as holding down the keys to get some hammer-drill effects! The next example demonstrates the use of two voices for a simple echo effect, and how frequency changes can alter the sample:
TODO - "Update AMOS below with AOZ"

```
Load "AMOSPro_Examples:Samples/Mixture.Abk",10
Sam Bank 10
Sam Play 1,12 : Wait 5: Sam Play 2,12: Rem Simple echo effect
Wait Key
Sam Play 1,13,2000: Rem Lower Pitch
Wait Key
Sam Play 1,13,5000: Rem Higher Pitch
```

SAM STOP

instruction: stop one or more samples playing

```
Sam Stop
Sam Stop voices
```

This simple command is used to stop all samples playing through your loudspeaker system. If it is followed by an optional voice parameter, only the selected voices will be switched off. Voices are chosen using a binary bit-map, where any bit that is assigned to 1 will have the associated voice terminated. Otherwise it will be ignored. The voices are associated with the following bits:

```
Voice 3210
Bitmap %1111
```

For example, the next line would stop samples playing on voices 3 and 1:

```
Sam Stop %1010
```

Changing a sample bank

Digital sound samples are normally stored in memory bank 5, but there are no restrictions on assigning other banks to hold samples.

SAM BANK

instruction: change the current sample bank

```
Sam Bank bank number
```

The SAM BANK instruction dictates that all future SAM PLAY commands will take samples from the newly specified memory bank. If several parallel banks are set up, AOZ can swap between them with a simple call to the SAM BANK command. To hear all of the samples used in one of the AOZ example games, load the following file and listen to what is stored in the memory bank, using this routine:
TODO - "Update AMOS with AOZ below"

```
Load "AMOSPro_Productivity:Wonderland/Wonderland Samples.Abk"
Sam Bank 6
For A=1 To 5
    Print "Sample number ";A
    For B=1 To 3
        Sam Play A
    Wait 20
    Next B
Next A
```

Playing a sample from memory

Samples do not have to be held in a special bank. In fact, a "raw" sound sample can be stored anywhere in the computer's memory using BLOAD, and then played with the following command:

SAM RAW

instruction: play a raw sample from memory

```
Sam Raw voice,address,length,frequency
```

SAM RAW plays a raw sample, and can be used to scan through any program or sound library discs, searching for a sample that matches the given parameters.

The voice parameter has already been explained. Address refers to the location address of the sound sample, which is normally inside an AOZ memory bank, but can be anywhere. The length parameter confirms the length in bytes of the sample to be played. Finally, frequency dictates the playback speed of the original sample,
given in Hertz.

A typical raw sample command looks like this:
TODO - "Update AMOS with AoZ below"

```
Reserve as work 10,21480
R$="AMOSPro_Examples:Samples/Mixture.Abk"
Bload R$,Start(10)
Sam Raw 15,Start(10),length(10),3000
```

SAM LOOP ON

SAM LOOP OFF

instructions: toggle repetition of a sample

```
Sam Loop On
Sam Loop Off
```

There are many instances where a single sample needs to be repeated over and over again. SAM LOOP ON ensures that all sound samples which follow this instruction will be looped continuously. To turn off the looping facility, simply call the SAM LOOP OFF command. Add a SAM LOOP ON instruction to the last example, before the SAM
RAW command, and hear the result.

Double buffered sampling

Please note that double buffered sample is not needed on modern machines, the original Amiga instructions being simply ignored by AOZ.

Samples are ideal for generating realistic sound effects directly from AOZ programs. However, as the samples get longer, their memory requirements become prohibitive! If sound samples are used sparingly, several seconds of perfect audio effects can be conjured up by an unexpanded Amiga. Unfortunately, continuous
soundtracks would seem like an impossibility, with one minute of digital sound consuming almost a megabyte of data!

The principle of Double Buffered Sampling is very similar to the display system employed by the screen Double Buffer. It works by storing two sample banks in memory, the physical and the logical sample banks.

The physical bank holds the sample which is currently being played through your loudspeakers, and the logical bank contains the sample that is being loaded from disc.

At the beginning of a program, the physical and logical banks are loaded with the first two blocks of sample data.

The physical sample can then be played with a SAM RAW command, and whenever the physical bank runs out of information, the banks are swapped over to create a seamless audio signal. The logical bank now becomes the physical bank, and vice versa.

AOZ provides three powerful commands used to control this process directly. SAM SWAP activates double buffered sampling, SLOAD loads a specific section of the sample file in memory, and the SAM SWAPPED function checks for the need to load a new block of sample data.

Here is the procedure for using double buffered sampling:

Reserve two memory banks in CHIP Ram, to hold the logical and physical samples. The size of these banks will depend on the playback frequency, and the amount of background graphics.

It is good practice to start with a small value, such as 8k, and then increase this value until the resulting sound effect is perfect.

Load the first two segments of the sample into the logical and physical memory banks, using the SLOAD command.

Play the physical sample, using a SAM RAW instruction.

Activate the sample swapping process with SAM SWAP.

Continue with the main program, and simply test the status of the sample at regular intervals using the SAM SWAPPED function. A value of -1 (True) will indicate the need to load the next segment of sample data with SLOAD.

The swapping system is then re-initialised by SAM SWAP, and the process continues.

SLOAD

instruction: load a section of a sample

```
Sload channel number To address,length
```

The SLOAD instruction is an extended version of the BLOAD command, and it is designed to load selected parts of a file into memory, one section at a time.

First give the channel number of a sample file stored on disc, which has been previously opened with an OPEN IN instruction. This is followed by the destination address of the data in memory. This will normally be the start address of an AOZ memory bank. Finally, the length of the sample section is given, in the form of the number of bytes to be loaded into memory. These bytes will be loaded directly from the current position of the file pointer, and this pointer may be moved anywhere in the file, using the POF function. This means that you have
complete control over the starting point of the loading operation. Obviously, if the requested position lies outside of the current file, an error will be reported. If the length is larger than the actual file, AOZ will read all
of the remaining bytes up to the end of the file.

After the data has been successfully loaded, the file pointer will be moved to the next byte in the sample automatically.

It should be apparent that the SLOAD command can be used for applications other than sample loading. It will work equally well with any drive, including the internal floppy drive, but only a hard disc, Ram-disc or CD ROM will
be fast enough to load samples, for use with the SAM SWAP command! This is explained below.

SSAVE

instruction: save a data chunk anywhere into an existing file

```
Ssave channel number, start To end
```

This command is the reverse of SLOAD. It allows you to save a chunk of memory data into a file opened with OPEN OUT or APPEND. Use the LOF, POF and EOF functions to control where you want to position the data within the file.

SAM SWAP

instruction: activate sample-switching

```
Sam Swap voices To address,length
```

The SAM SWAP command activates the automatic sound-swapping system. It specifies the location and size of a logical sample which has been loaded previously with the SLOAD instruction. The sample will be played through loudspeakers the instant that the current physical sample runs out of data.

The voices parameter is a bit-pattern that defines which voices are to be used for playing the sample. Each bit in this pattern sets the swapping on a particular voice, according to the following format:

```
Voice 3210
Bitmap %1111
```

The address parameter gives the address of the next logical sample in memory.

Length is simply the number of bytes to be played of the new section of sample.
It is essential to note that SAM SWAP only works with an existing sample. It does not actually play a sample through a loudspeaker. Therefore, it has to be initialised by a SAM RAW command before it can be used, which will start off the physical sample and set up the playback speed for the entire sample of sound. For example:

```
Sam Raw %0011,Start(5),20000,12000 : Rem 12000 is the playback speed
Rem Swap the samples assigned to voices 0 and 1 using data in bank 6
Sam Swap %0011,Start(6),20000 : Rem 20000 is the length of logical sample
```

SAM SWAPPED

function: test for successful sample swap

```
value=Sam Swapped(voice number)
```

Use the SAM SWAPPED function to test the specified voice to see if the logical and physical samples have been exchanged by a SAM SWAP command. The voice number is a value from zero to 3, and if the sample is being played on a number of voices simultaneously, any of those voice numbers can be used.

SAM SWAPPED will make a report by giving one of the following three values:
-1 means that the previous physical sample has finished playing, and the samples have been successfully swapped. It is now time to load a new sample into the logical memory bank, and call the SAM SWAP command again. Please note that this value is also returned after a normal SAM RAW command has finished playing a sample.
Zero means that the physical sample is currently being played, and there is a fresh logical sample waiting for the automatic switching operation. In this case, there is no need to load any further information into memory.

1 means that the sample player has run out of data, and that the sample swapping operation has failed. You will now need to re-initialise the sample from the beginning, using a SAM RAW command. If this value is repeatedly given,
your logical and physical sample banks are probably too small. Try increasing their size to the next sensible value.

The SAM SWAPPED function should be called at regular intervals while the sample is being played. It can be used as part of the main program loop, or called automatically after a set period, using the EVERY command.

Here is a typical listing, that demonstrates how these commands should be used:

```
Reserve As Chip Work 10,10000
Reserve As Chip Work 11,10000
Open In 1, "Dh0:Name_of a_big_sample"
L=Lof(1) : C=0 : A=Start(10)
Sload 1 To Start(10),10000 : C=C+10000
Sload 1 To Start(11),10000 : C=C+10000
Sam Raw %1111,Start(10),10000,10000
Do
    Gosub CHECK_SAM
    If C>L Then Goto FINI
    Sam Swap %1111 To Start(11),10000
    Sload 1 To Start(10),10000 : C=C+10000
    Gosub CHECK SAM
    If C>L Then Goto FINI
    Sam Swap %1111 To Start(10),10000 : C=C+10000
    Sload 1 To Start(11),10000
Loop
CHECK_SAM:
Repeat
A=Sam Swapped(1)
Locate 0,0: Print A;" "
Until A=-1
Return

FINI:
Close 1: End
```

TODO - "Update to include AOZ PC abilities too"