# Sound and Music in Videogames

Harry Holmwood

Sony Computer Entertainment Europe

# Use of Sound Effects

- To enhance atmosphere
- Provide audible ‘feedback’
- To surprise players
- Give clues

# Sound effects

- Be careful with tuning when playing with music
- Provide positive feedback
- Realistic is too boring
  - Be ‘bigger than life’

# Music

- Enhance atmosphere
- Provide excitement
- Build up expectations....
- either fulfill them or not (© Fatman)

# Interactive Music

- Music which reacts to in-game action
- Sudden changes
  - Dramatic effect
  - Scary
  - Easy to do
- Smooth changes
  - Less dramatic
  - More subtle
  - Harder to do ?

# Sudden changes

- Branching MIDI
  - Separate MIDI sequences
  - Stop one, start another
  - Use an SEP file
- CD-DA
  - No processor overhead
  - Best quality
  - Long seek times
  - High storage requirements

# Multi-channel CD-XA

- Interleave multiple audio channels
- Playback one at a time
- Saves disk space
- No processor overhead

# Example CTI file fragment

```
XAFileAttributes  Form1  Audio
XAVideoAttributes ApplicationSpecific
XAAudioAttributes ADPCM_C Stereo
XAInterleavedFile  even.xa    c:\gamedata\even.xa

XAChannelInterleave TimeCritical 1-2-3-4

XAChannel           1
     XAFileAttributes    Form2  Audio
     Source              c:\data\wav\d1.xa
     MinLength 270000
        ;note this is the length of the longest of the 4
 XAEndChannel

; ..... other 3 channels in here
XAEndInterleavedFile
```

# Smooth branches

- Layered MIDI sequences
  - One for each character in an adventure game
  - Use ‘markers’ in the MIDI sequence to synchronise branches
  - Can’t jump into a sequence - running status
  - Lower quality than CD audio
- Atmosphere loops
  - Fade sample loops to alter atmosphere
  - No MIDI sequencing
  - Uses a lot of Sound RAM

# Atmosphere Loops

- Several samples playing simultaneously
- Fade them in and out to change atmosphere
- SsUtSetVVol() to change a channel’s volume
- Dynamically start and stop loops as required to save polyphony

# MIDI

- Very low storage space for sequences
- Must store instrument sounds in SPU RAM
- LibSnd provides MIDI playback functionality
- Can adjust tempo etc in real-time

# Sequencers

- Sequencing with SoundDelicatessen ?
- Don’t bother
  - Use a separate sampler
  - Write your music
  - Convert at the end

# Why doesn’t it work ?

- SoundDelicatessen written for Apple MIDI Manager
- Sequencers NOT written for MIDI Manager
- OMS can work

# So how do you do it?

- Preferably, use two computers
  - One (Mac, PC, ST etc) for sequencing
  - One for Sound Artist Card
  - Treat card as any other MIDI device
- Otherwise (what I do)
  - Write music using a separate sampler (eg SampleCell, AKAI etc)
  - Convert samples once composition is complete

# Sampling

- Equipment needed
  - Sampler
  - Mixer
  - EQ
  - Compressor
- Sample from DAT / CD
  - Can re-sample if necessary
  - Can EQ and compress sample
  - Normalise samples

# Sample Editing

- Sample must start and end cleanly
- Loop points must be on 28 sample boundaries
- Divisible by 28
- No other markers
- Use a crossfade to smooth loop

# Sample Editing

- Save as AIFF mono files
- Any sample rate up to 48kHz
- Convert with AIFF2VAG

PlayStation TM

# CD-DA

- Provides the highest quality
- Can be played in a standard CD player
- Drive plays at single speed
- Cannot interleave data or graphics

# CD-XA

- Audio encoded as XA-ADPCM
  - Approx. 4 times smaller
  - 37.8 kHz or 18.9 kHz
- Use RAW2XA on the Mac
  - Takes 18.9/37.8 kHz mono or stereo
  - Sound Designer II format
  - Batching - ‘Interactive’ mode

# Producing audio for video

- Use XA-ADPCM audio
  - Created with RAW2XA
  - Can be interleaved with video
  - MOVCONV / BUILDCD

# Ideally

- Have video on BetaCam tape
  - MIDI interface with SMPTE
  - Synchronise sequencer to video
- More likely
  - Write down the frame numbers
  - Guess

# Interleaving using BUILDCD

- SN Systems provide BUILDCD with the CD-Emulator
- For 37.8 kHz stereo, can interleave one sector of audio for every seven of video (double speed).
- Output a .CCS file, which can be used with the CD-Generator software
- Use XA files created by RAW2XA - remove subheaders

# Interleaving with MOVCONV

- Can convert WAVs to XA files (ie no need for RAW2XA on the Mac)
- Use 16-bit wavs at either 37.8 or 18.9 Khz
- No need to remove subheaders

# PlayStation Sound Specs

- PlayStation uses ADPCM sound compression
  - Approx 4:1 compression ratio
  - 16-bit compressed down to 4
  - Similar to SNES sound chip
- 24 simultaneous sounds
- 512kB sound RAM to store samples

# Pitch setting

- Raise samples by up to 2 ocataves
- Lower samples by up to 12 octaves
- Can specify fine intervals of a semitone or less

# Getting samples into sound RAM

- ➢ Samples built into VAB files - a sound bank
- ➢ VAB split into .VH (Header) and .VB (Body) files using VABSPLIT.EXE
- ➢ Open the VAB header - gives you a VAB id
- ➢ Load the VB into memory
- ➢ Transfer it into sound RAM

PlayStation TM

# Getting samples into sound RAM

```
#define VH_ADDR  0x80025000
#define VB_ADDR  0x80030000

short gVAB;
/* open VAB header */
gVAB = SsVabOpenHead (VH_ADDR, -1);

if (gVAB < 0)
   printf ("SsVabOpenHead : failed\n");

if (SsVabTransBody (VB_ADDR, gVAB) != gVAB)
   printf ("SsVabTransBody : failed!\n");

   SsVabTransCompleted (SS_WAIT_COMPLETED);
```

# Saving Main RAM

- Not enough RAM to store entire VB file ?
- Use SsVABTransBodyPartly
  - Uses a small buffer to store parts of the VB file
  - Fill the RAM buffer, transfer it to SPU RAM, and start again
  - Until whole VB is transferred

# Transfer in background

- SsVabTransCompleted (SS_WAIT_COMPLETED) blocks until the transfer is complete
- SsVabTransCompleted (SS_IMMEDIATE) returns immediately.
- Returns 1 if the transfer is completed, 0 if it is ongoing

# Initialising the sound system

- Set NTSC/PAL with SetVideoMode
- SsInit() to start the sound system
- Set the ‘tick mode’ - use SS_TICKVSYNC
- Use SsStart() to begin sound processing
- Use SsEnd() to stop it
- Finish with SsQuit()

PlayStation TM

# Sound Initialisation

```
/* Initialise sound system (libSnd) */
  SsInit();

/* Set ‘tick mode’ to work regardless of NTSC/PAL
  settings */
  SsSetTickMode(SS_TICKVSYNC);

/* Begin Sound Processing */
  SsStart();

/* Set main volume */
  SsSetMVol(leftVOL, rightVOL);
```

# VAB file

- ➢ Bank file for sample data
- ➢ Contains VAG files which are ADPCM samples
- ➢ Has a header and a body (just sample data)
- ➢ Body goes into sound RAM
- ➢ VAB contains programs, which are made up of tones (samples) and ADSR atributes

# Playing a sample (libsnd)

- Easiest way is with SsUtKeyOn
- Specify program number
- Tone number
- Specify Note - 0 to 127
- Specify fine tuning - 0 to 127
- Returns a channel number

# Playing MIDI files

- LibSnd provides MIDI playback functionality
- Use SMF2SEQ to convert your MIDI files
- No aftertouch - minimise continuous controllers
- Sequence playback in SoundDeli did not work
- Check MIDI playback on the DTL-H2000 instead

# MIDI commands

- Must set ‘tick mode’
  - SS_TICK60 / SS_TICK240 - change resolution
  - SS_TICKVSYNC should work for NTSC or PAL
- SsSeqOpen( addr, vab_id ) - Must do this first
  - Give it the address of the SEQ file in memory
  - And the ID number of the VAB it is associated with

# Sequence Playing

- SsSeqPlay
  - Can tell it to play the sequence once, or repeatedly
  - Can set sequence playback to ‘pause’
- SsSeqStop( seq_access_num )
- SsSeqPause / SsSeqReplay

# Sequence Tempo

- SsSeqSetAccelerando / SsSeqSetRitardando
- Allow slowing / speeding up of a sequence
- Specify a new 1/4 note resolution
- Specify a delta time - in ticks
- Basically the same function

# Sequence Volume

- SsSeqSetCrescendo / SsSeqSetDecrescendo
- Raise / lower volume over a period of time
- Volume is added (crescendo) or subtracted (dec.) from current volume
- Takes a delta time - in ticks

# Other MIDI functionality

- SsSetLoop - Sets the number of repititions of the song
- SsSeqSetNext - Specify the next sequence to be played
- SsSetTempo - Sets tempo explicitly
- SsSetMarkCallback - put markers in sequences
    - Perhaps synchronise animation to a beat

# Manipulating a sample

- Adjust Volume - SsUtSetVVol
- Adjust pitch - SsUtPitchBend, SsUtChangePitch
- Adjust ADSR - SsUtChangeADSR
- Adjust other attributes - SsUtSetProgAttr
- Autopanning / autovolume
  - SsUtAutoPan
  - SsUtAutoVol

PlayStation TM

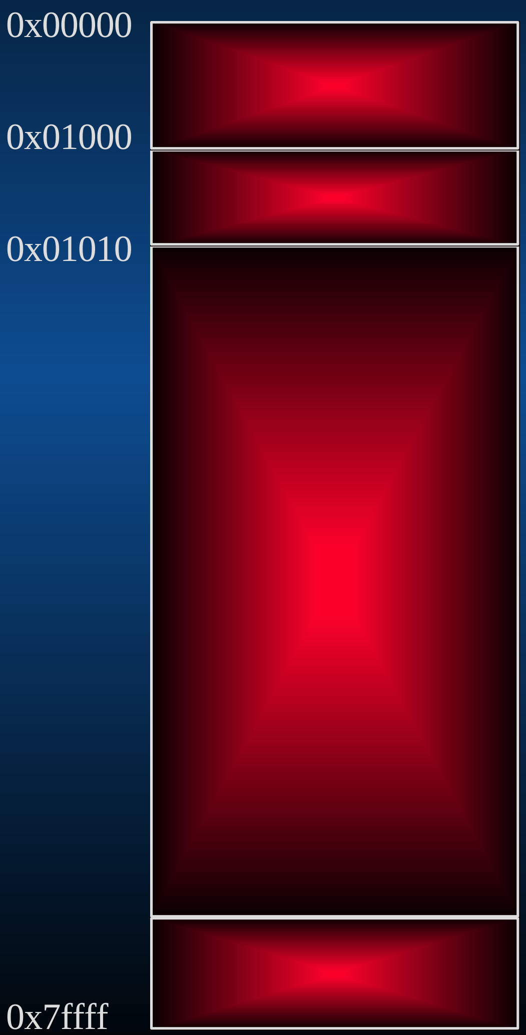
# Using Reverb

- PlayStation DSP offers several reverb algorithms
- Reverb uses SPU RAM as a buffer
- The longer the reverb, the more memory is required
- You set:
  - Algorithm - SsUtSetReverbType
  - Depth - SsUtSetReverbDepth
  - Delay (for delay / echo) - SsUtSetReverbDelay
  - Feedback (for delay) -

# Effect Algorithms

| Effect | Memory | Description |
|---|---|---|
| Room | 9,920 | Short Reverb |
| Studio A | 8,000 | Small Studio Reverb |
| Studio B | 18,496 | Medium Studio Reverb |
| Studio C | 28,640 | Large Studio Reverb |
| Hall | 44,512 | Large Hall reverb |
| Space | 63,368 | Huge spatial reverb |
| Echo | 98,368 | Single echo |
| Delay | 90,368 | Repeating delay |
| Pipe | 7,072 | Metallic pipe |

# SPU RAM layout



0x00000

SPU Decode Data Region

0x01000

Additional Loop information

0x01010

Waveform Data
Transferrable region

0x7ffff

# Using multiple VAB files

- Use SsVabOpenHeadSticky
- Must specify the VB address in sound RAM
- Load it above 0x01010
- Careful not to move into reverb work area

# Using Sound Artist Tools

- Nubus card for Apple Macintosh
- Provides sound functionality of PlayStation
- Has optical digital out
  - Requires external DAC or DAT
- Software converters and bank builders

# AIFF2VAG

- Converts AIFF files into VAG (PlayStation sample) format
- Loops on 28 sample boundaries
- Can batch process them in ‘interactive’ mode

# VAG Compression modes

- Standard - for general sound sources
- High band - for sound sources with high-band components
- Low band - For sound sources with low-band components
- 4-bit straight - Four bit straight compression

# RAW2DA

- Converts raw sample data into DA format
- Use when you need CD-DA audio
- Requires 44.1kHz RAW stereo data
  - Sound Designer II format
- Has an ‘auto’ mode for batch processing

# RAW2XA

- Converts raw sample data into XA-ADPCM
- For interleaving with video, other sound or any data
- Requires 18.9kHz or 37.8 kHz mono or stereo RAW sample data
  - Sound Designer II format
- Also has an ‘auto’ mode
- Extract subeaders for BUILDCD

# SMF2SEQ

- Converts MIDI file data into PlayStation SMF format
- Don’t use Aftertouch
  - Lots of continuous controller data ruins playback
- Can use SEQ2SEP on PC to build SEQs into a bank of sequence files

# Vag Player

- Allows playback of VAGs
- Test sample is looping smoothly
- Check it fits in sound RAM

# Sample looping

- Demo sample looping on Mac
- Show looping points at 28 sample boundaries
- Do a crossfade
- Show playing on VAG player

# 3D sound

- Make sounds appear behind player
- Can give a cinematic feel
- Pre-recorded
  - Done in a studio
  - Specialist hardware encoders
- Real-time
  - Allows movement of samples in 3D space
  - More interactive and suited to games
- Don’t overdo it

# QSound

- Get 3D sound from stereo speakers
- Uses ‘image file panning’
- Relys on phase shifting to fool ear
- Requires 2 copies of a sample to be stored
- Costs around £2000 per game in Europe

# Dolby ProLogic

- Requires additional speakers
- Plus a Dolby ProLogic Decoder
- Many hi-fi systems now come with ProLogic
- No license fee (?)
- Here’s how.....

# Dolby ProLogic (contd)

- Must vary amplitude and phase relationships in the left/right channels

| Encoded Channel | Left Output | Right Output |
|---|---|---|
| Left | 0dB | off |
| Center | -3dB | -3dB |
| Right | off | 0dB |
| Surround | -3dB | -3dB |

# SPU streaming

- New feature of libspu
- Play VAG files of any length
- Can stream VAG files from main RAM into SPU RAM
- Unlike interleaved XA-ADPCM, can overlay VAGs

# Interactive streaming

- Allow music to behave interactively
- But keep the production values high
- More cinematic
- More relevant

# Saving SPU RAM space

- Use low sample rates for low-frequency sounds
- Use MIDI and pitch shifting to allow samples to sound longer
  - Demo Total NBA crowd
- Stream sound effects/speech from CD
- Use Multiple VAB files, loading in as necessary

# LibSPU or LibSND ?

- LibSnd good enough for most purposes
- Has reasonable MIDI functionality
- Does not allow for much manipulation of sounds
- LibSpu has no MIDI functionality
- Provides lower level functionality

# Sound RAM Interrupt

- Can set a callback function to play when an area of sound RAM is ‘hit’
- Can use this to detect end of samples
- SpuSetIRQ
- SpuSetIRQAddr
- SpuSetIRQCallback
- VABSPLIT -v
  - Outputs a VAG address table

# Example VAG address table

```
#define VAGS_engine 1
unsigned long engine[] =
   {
   0x0,
   0x5540,
   }; /*vag table from engine.vab" */
```

# ‘Free running’ problem

- Interrupts were getting triggered for loops after the loop had been stopped
- Due to internal ‘virtual pointers’ continuing to loop after loop end
- AIFF2VAG version 1.6 onwards should cure this
- Use latest libraries

# Finally.....

- DON'T leave the sound until the end of a project
- DO include sound technologies in your technical designs
- Be as INNOVATIVE with sound as you are with graphics
- IF IT SOUNDS GOOD, IT IS GOOD