1 Before Red Book: Early Video Game Music and Technology

JAMES NEWMAN

The Rise and Rise of Video Game Music

There can be no doubt that interest in video game music has grown considerably in recent years. It is notable that this passion is not reserved only for the latest releases, but extends back to the earliest days of the form, with as much praise heaped upon 1980s Commodore 64 or 1990s Amiga 500 music as on the high-profile symphonic soundtrack *du jour*. Contemporary game developers, such as Terry Cavanagh, underscore games like *vvvvvv* (2010) and *Super Hexagon* (2012) with soundtracks that draw directly on the early home computing and console instrumentation and form. In doing so, they demonstrate their own fandom, as well as a desire to draw on the distinctive aesthetics of early gaming and their association with an era of putatively simple yet unabashedly complex gameplay. We further note exhaustive online archival collections such as the HVSC (High Voltage SID Collection) or VGMrips that gather together music files extracted from original game code. For their part, enthusiastic programmers expend extraordinary effort on the production of software applications such as VGMPlay and Audio Overload, which are solely dedicated to the task of playing back these music files with no care or support for reproduction of graphics or gameplay.

The influence and impact of video game music is felt beyond the community of game fans: contemporary artists such as Tokyo Machine are fully fluent in the language of early video game music, while Grandaddy's 'A.M. 180' (1998) and Beck's 'Girl' (2005) demonstrate a bilingual mix of indie rock guitar instrumentation and 1980s video game consoles. In addition, a litany of tracks directly sample from titles as diverse as David Wise's 1995 Super Nintendo *Donkey Kong Country 2* (on Drake's 2015 '6 God') and David Whittaker's 1984 Commodore 64 *Lazy Jones* (Zombie Nation's 1999 'Kernkraft 400').

To ease the process of accessing this sonic territory, contemporary synthesizer and sampler manufacturers draw on the distinctive sounds of

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

early video game music in their banks of presets, programs and raw waveform data. Plogue's Chipsounds software instrument offers painstakingly detailed emulations and recreations of a host of home and arcade-game sound systems, while Korg's Kamata instrument explicitly references Namco's 1980s game music, even going as far as integrating presets designed by legendary composer and sound designer Junko Ozawa, as it 'reconstructs the C30 custom sound engine that swept the world in the 1980s . . . [and] lets you play the classic video game sounds of the past'.[1]

To palpably extend the impact of early video game sound into the present, some manufacturers build new machines explicitly influenced by the architectures and sonic fingerprints of specific systems, such as Mutable Instruments' Edges module that mimics the Nintendo NES, or Noise Engineering's Ataraxic Translatron that nods to the Atari VCS console. Some go further still in creating instruments that eschew emulation or simulation and instead are built around the physically extracted innards of old home computers and gaming consoles. For instance, Elektron's SIDStation and ALM Busy Circuits' SID GUTS repurpose the Commodore 64's Sound Interface Device (or SID chip) sound chip, while Twisted Electrons' AY3 module has at its heart two of General Instrument's AY-3–8912 chips, variants of which provided the sonic foundation of countless computers, consoles and arcade systems, including the Spectrum 128, Mattel Intellivision and Capcom's *1942* arcade game (1984).

But notwithstanding the enormity and creativity of the labour that continues to see these sounds celebrated and embedded into contemporary electronic music and gaming cultures, there remains little detailed historical work on early video game music. The few popular and scholarly histories of early game music that do exist are frequently based around discursive formulations and conceptualizations of technology that reinforce totalizing teleological narratives rather than add nuance to the picture.

This chapter suggests how we might arrive at a nuanced discussion of the role and function of technology in relation to video game sound and music. In doing this, the chapter will address some of the limitations of extant approaches, and outline some alternative ways of thinking that reconfigure the relationships between hardware, software and, crucially, the creative endeavour of composers, coders and sound designers.

[1] Korg, 'Kamata Wavetable Synthesizer', *korg.com*, 2016, accessed 8 April 2020, http://gadget.korg.com/kamata/index_en.php.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

Early Video Game Sound: When or What?

Before we proceed with this analysis, it is essential to address what we mean by 'early' game sound and music. Surprisingly, this is a difficult task, with no universally agreed definition. While there are distinctive practices, tools, technologies and techniques that are characteristic of game music making and reproduction throughout the 1970s, 1980s and 1990s, these ultimately continue to be evident, and are in use to this day, albeit in different contexts. For instance, central to the definition of early video game music is the sound chip. Whether they took the form of a TV-connected console or handheld device, a general-purpose home computer or a dedicated arcade cabinet, gaming systems created in the first three decades of the medium's existence all generated their sound in real time via a sound chip. These specially designed pieces of silicon were effectively synthesizers capable of generating waveforms played back at specific pitches, with timing and durations specified by code created in advance by a composer and programmer. Just as a graphics chip takes data about colour, x-y coordinates and scrolling and processes them in real time to create images on screen, so too the sound chip responds to data in order to generate the game's aural components.

The references to data are important here, and the methods and working practices for dealing with sound chips are particularly noteworthy. Contemporary video game composers likely expect to work with a Digital Audio Workstation for composition and sound design, might have vast libraries of hardware or virtual instruments and effects and use a plethora of software tools like Wwise to facilitate the integration of sound, music and interactivity. Each of these tools typically offers a graphical user interface, or a physical set of controls in the case of hardware, which are typically assessed in terms of their 'user friendliness', as Pinch and Trocco note.[2] Composers working with 1980s sound chips found themselves in an altogether different situation. Crudely speaking, there were no tools save for those that they created themselves.

Commenting on the situation in the early 1980s when he began composing for the Commodore 64, Rob Hubbard observes,

> There were no MIDI sequencers, no Trackers. We coded everything just in an Assembler. I used to load up a machine code monitor and literally display the bytes in real time. The music was all triggered on the raster interrupt and I would start

[2] Trevor Pinch and Frank Trocco, *Analog Days* (Cambridge, MA and London: Harvard University Press, 2002), 309–13.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

changing the numbers in real time to alter the synth settings and musical notes. So, I would tend to work on four bar chunks that I would tend to repeat and I would sit on that Hex editor, changing things. I would sit and tweak all those numbers until I had the four bars pretty much the way that I wanted them to sound and that would let me continue on for another 16 bars . . . [.][3]

Consumer-facing music-making hardware and software did exist, such as the *Commodore Music Maker* (1982), but these were too inefficient for use in game development, meaning that working with code remained the only viable solution. The experience of writing for the NES was very similar, as Manami Matsumae, who composed the soundtrack for the first *Mega Man* game, explains.

[N]owadays it's a lot simpler to get the data that you want to create, and if you need to make any changes, it's not that difficult. Back then, in order to put the musical data into the ROMs, and you had to convert the musical notes into numbers . . . there were times when you'd have to put the entire game into ROM and test it, but there are also times when only the music had to go in.[4]

As Collins notes, the lack of any readily available or suitable tools, and the necessity to work either directly with machine code or via a code-oriented interface,

meant that most early games composers were in fact programmers working on other aspects of a game, or at best in rare cases, in-house programmer-musicians who had to work closely with programmers.[5]

Even when such tools were later developed, they still demanded a combination of computing and musical literacy. The 'Tracker' to which Hubbard refers, for instance, is a sequencing application that takes its name from Karsten Obarski's *The Ultimate Soundtracker*, developed in 1987 at German developer Rainbow Arts. The Tracker arose from the frustrations Obarski felt writing music for the Commodore Amiga, and derived its visual interface from the vertical piano roll.[6] Yet music and sound information was still entered as a series of hexadecimal values rather than as traditional musical notation and the Tracker's resemblance to a code listing

[3] Rob Hubbard, 'The Golden Days of Computer Game Music', *Assembly 2002 Helsinki*, 2002, accessed 8 April 2020, www.youtube.com/watch?v=DiPdjbsiQqM.

[4] Jeremy Parish, 'Manami Matsumae, the Maestro of Mega Man', *USGamer*, 20 January 2016, accessed 8 April 2020, www.usgamer.net/articles/manami-matsumae/page-2.

[5] Karen Collins, 'Flat Twos and the Musical Aesthetic of the Atari VCS', *Popular Musicology Online* 1 (2006a).

[6] Kevin Driscoll and Joshua Diaz, 'Endless Loop: A Brief History of Chiptunes', *Transformative Works and Cultures* 2 (2009).

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

was unavoidable. Of perhaps even greater significance than the interface was Obarski's implementation of the 'Module' or 'MOD' file format, which defined a collection of 'instruments' in the form of constituent samples, patterns that specified when and how the samples were played, and a list that set out the order in which those patterns would be played.

We will return to the importance of the interface between composer and sound chip later, but here we should note that the programming of sound chips to replay music in real time sits in stark contrast to systems such as the PlayStation 4 and Xbox One, for instance, which stream music from a hard disk, CD or from an online source. In these systems, music is not performed in real time by a sound chip responding to coded instructions but rather is prerecorded and played back in much the same manner as it might be played back on a CD player or portable music player. Accordingly, the scope of the musical creation is not defined by the capabilities of the sound chip, and might make use of any form of instrumentation available to the modern musician, composer, producer or recording engineer.

The utilization of prerecorded music in video games had been greatly precipitated by Sony's 1995 PlayStation console, which brought the first commercially successful mainstream implementation of audio streaming from a built-in CD drive. In addition to creating and commissioning original music, CD playback brought the possibility of licensing pre-existing tracks. Psygnosis's *Wipeout* (1995) led the charge, and serenaded players with the sounds of the Chemical Brothers and Orbital. Beyond this, some games even allowed players to swap out the game CD and its supplied soundtrack and replace it with a disc of their own, as with NanaOnSha's *Vib-Ribbon* (1999), in which the game's landscape was derived from an audio analysis of the CD audio content. The disc swapping possible with games like Namco's *Ridge Racer* (1993), and central to *Vib-Ribbon*, arose because the entirety of the game's program data could be loaded into the PlayStation's memory (rendering the disc is no longer necessary), and because the PlayStation console was able to decode standard audio CDs. Indeed, the PlayStation game discs were often supplied as 'Mixed Mode' discs, meaning there was one partition with game data and one with CD audio, making it also possible to playback game music in a consumer CD player. It was this adoption of the CD audio format, or what is technically termed the 'Red Book' audio standard, that represents perhaps the greatest shift in game music technology. It also serves as a clear pivot point for 'early' game music, as prior to this moment, music was generated and created in a markedly different manner, with different working practices and tools and audibly different results.

However, while this might seem like a neat delineation point, the launch of the PlayStation, Sega Saturn and subsequent home, portable and arcade systems employing prerecorded audio streaming did not mark the end of sound-chip-based music making or reproduction. Countless handheld systems such as Nintendo's Game Boy Advance and DS series as well as their Nintendo 64 home console continued to offer no optical disc streaming and, therefore, relied still on the use of sound chips not merely as an aesthetic choice but as a technical necessity.

If we use the widespread adoption of prerecorded music as a historical watershed, the definition might seem to be based as much around what it isn't as what it is; how it isn't produced, and which technologies and creative practices aren't deployed in its production, distribution and reception. Perhaps more accurately and more positively, however, we can define the era of 'early' game music as the period during which the use of real-time sound chips was the only way to create and hear game music, rather than the Mixed Mode discs and mixed economy of forms of streaming and sound chips that we identify from the mid-1990s onwards.

'Chiptunes' and the Flattening of Early Video Game Sound

One of the consequences of the increased popularity of early video game music has been its contemporary designation as 'chiptunes'. As McAlpine notes, the term appeared in the early 1990s, has grown in popularity since, and is now often used for any music created with sound chips (or the sounds of sound chips), whether produced today or thirty years ago.[7] Given the slippery nature of definitions based around temporality, perhaps this chronological ambiguity might be useful. However, even though 'chiptunes' has the distinct advantage of being in popular parlance, it brings with it some significant issues. From the perspective of the historian, the term 'chiptunes' has an unhelpful flattening effect. Quite simply, there is such a wide variety of 'chips' with which video game music is, has been, and can be made, that such a designation is helpful only in distinguishing it from music from disk or online streaming. 'Chiptunes' tells us nothing of the differences between the affordances of the Commodore 64, NES or *Space Invaders* arcade cabinet, or how composers and sound designers might investigate, harness and extend their potentialities.

[7] Kenneth B. McAlpine, *Bit and Pieces: A History of Chiptunes* (New York: Oxford University Press, 2018), 6.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

This flattening effect is particularly troublesome given the tendency of current accounts of early video game music to deploy the language of 'bleeps' and 'blips' in their often pun-heavy titles (e.g., BBC Radio 4's *While My Guitar Gently Bleeps*, 2017).[8] The origin of this construction of early video game music is understandable as many of the very earliest home and arcade sound chips utilized in 1970s systems offered few of the sound-shaping tools such as filters, or fine-grained control over pitch and amplitude, that would become commonplace from the early 1980s. As such, raw waveforms were typical, and particularly square waves which, requiring only binary on–off instructions to generate, are the simplest to create. Certainly, such sounds are inescapably electronic, frequently associated with the sounds of sci-fi control panels burbling, and are unlikely to be heard outside the most experimental and avant-garde academic music-research labs. Indeed, if we take the Atari 2600 (1977) as an example, we find that the sound capabilities of the market-leading home video game console were decidedly challenging.[9] The system's TIA (Television Interface Adapter) made use of a 5-bit frequency divider, which generated a limited number of mathematically related, but often musically unrelated, pitches. As the opening lines of the *Atari 2600 Music And Sound Programming Guide* have it, 'It is difficult to do music on the Atari 2600 due to the limited pitch and two voices . . . many of the pitch values are not in-tune with others.'[10] Were we only to consider examples such as the 1984 Atari 2600 *Gyruss* soundtrack, we might well be forgiven for building a view of game music as bleeps and bloops (and fairly discordant ones at that). To compensate for the TIA's esoteric tuning, Garry Kitchen, the developer of Activision's 1983 *Pressure Cooker*, 'determined a set of pitches that the Atari TIA could reliably reproduce. He then hired a professional jingle writer to compose theme music using only those available pitches'.[11] Similarly, PC-compatibles and home computers like the ZX Spectrum were equipped with just a 'beeper' circuit primarily intended for system-alert sound effects. Yet, music and sound design of some sophistication and complexity was coaxed from the Spectrum, which speaks to the ingenuity of composers and programmers, and further complicates the idea of bleeping and blooping.[12]

[8] See www.bbc.co.uk/programmes/b07dlx8y (last accessed 15 October 2020).

[9] Nick Montfort and Ian Bogost, *Racing the Beam: The Atari Video Computer System* (Cambridge, MA: The MIT Press, 2009), 131.

[10] Paul Slocum, *Atari 2600 Music and Sound Programming Guide*, 2003, accessed 8 April 2020, http://qotile.net/files/2600_music_guide.txt.

[11] Driscoll and Diaz, 'Endless Loop'.

[12] See Kenneth B. McAlpine, 'The Sound of 1-bit: Technical Constraint and Musical Creativity on the 48k Sinclair ZX Spectrum', *GAME: The Italian Journal of Game Studies* 6 (2017).

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

Yet, so pervasive is the idea that early video game music is no more than a series of bleeps, it has become necessary to tackle it head on. Question 3 on the HVSC's FAQ list is telling: 'Isn't Commodore C64 music just silly beep-blop music?' The answer, unsurprisingly, is an emphatic 'no!' Surely the palpable sonic differences between the TIA, ZX Spectrum, NES and Commodore 64 demand a distinction between their 'bleeps'? In discussing the absence of fine-grained analysis, Altice notes that,

> The output of the GameBoy, NES and Commodore 64 are now subsumed under the chiptune moniker, but the sonic character of those machines are far more unique than the Xbox 360, PlayStation 3, or Nintendo Wii. Games ported across those platforms will exhibit visual differences, but their soundtracks will remain the same. There is no 'sound' of the Xbox 360 any more than there is a 'sound' of an Onkyo CD player.[13]

One possible solution entails adopting an approach that focuses on the specificities of the particular sound chips in the Commodore 64, NES and Atari 2600 and conceiving of them not merely as musical instruments or constituents of gaming systems but as distinctive platforms for music and sound design in and of themselves. A 'platform studies' approach recognizes the significance and character of the base technological systems and the contribution their underlying design and capabilities make in defining the sound of the platform. As Arcangel et al. put it, 'Computers have personalities, shapes and architectures like a canvas that influence what we make.'[14]

Conceptualizing early video game music as 'chiptunes' conflates and smooths out differences between platforms, but a game converted from NES to Commodore 64 will sound as immediately and profoundly different as it will look and feel different. The NES's sound chip is recognizable, characterful and, most importantly, unlike that of the Commodore 64. The platforms contrast sonically in precisely the same way that Nintendo's joypad is unlike Commodore's keyboard and the NES's colour palette and display resolution are unlike the C64's. Indeed, if we look at the specification of the NES and Commodore 64 sound chips (see Table 1.1), we immediately begin to appreciate how these differences come to be so pronounced and why an approach that recognizes and foregrounds the specificity of design is so essential.

[13] Nathan Altice, *I Am Error: The Nintendo Family Computer/Entertainment System Platform* (Cambridge, MA: The MIT Press, 2015), 277.

[14] Cory Arcangel, Paul B. Davis and Joseph P. Beuckman, 'BEIGE as an Expert', (interview with Dragan Espenschied), *Post-Data*, 2001, accessed 8 April 2020, www.post-data.org/beige/beige_make.html.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

Table 1.1 A comparison of five early video game sound chips

System	Chip	Channels	Waveforms	Notes
Nintendo NES/ Famicom (1983)	**RP2A0X**	4	2 x pulse voices; 1 x triangle; 1 x noise	Additional 'DMC' channel can be used for sample playback. For pulse channels, pulse width can be set to 25, 50 and 75 per cent duty cycle. Amplitude variable between 15 fixed values (no amplitude over triangle channel).
Commodore 64 (1982)	**Sound Interface Device (SID Chip)**	3	All voices offer saw, triangle, pulse and noise waveforms (that can be combined). Waveforms can be varied per clock cycle. Pulse width is continuously variable.	Multimode Resonant Filter; per-voice amplitude ADSR envelopes; ring modulation, oscillator sync.
Sega Mega Drive/ Genesis (1988)	**YM2612** **Texas Instruments SN76489**	6 FM voices 4	Four Operator Frequency Modulation (FM) 3 square-wave generators and 1 noise generator	The Mega Drive uses two sound chips. The Texas Instruments SN76489 was also used in Sega's earlier Master System console.
PC-Compatibles (ALMSC released 1987)	**YM3812** **AdLib Music Synthesizer Card (ALMSC)**	9 (or 6 plus 5 percussion instruments)	FM with sine waves; three other waveforms inc. pseudo sawtooth via waveshaping	PC-compatible sound card with dual mode operation offering either 9-voice melodic playback or 6 melodic voices plus 5 percussion instruments.
Commodore Amiga (1985)	**Paula**	4	8-bit PCM samples (techniques were developed to replay 14-bit samples by combining 2 channels)	Separate to the Paula chip, the Amiga contains an analogue low-pass filter. The filter can only be applied globally to all four channels. The Amiga has stereo audio output – the four channels are mixed in pairs and hard-panned to the left and right audio outputs.

Examining the specification in this way, as in Table 1.1, we begin to reveal not only that there are differences between the capabilities of sound chips but also that the differences can be of great significance. For example, the SID chip offered variable waveform oscillators, Multimode Filter (buggy and unpredictable as it was), Ring Modulation and per-voice ADSR amplitude envelopes unavailable on the NES's RP2A0X chip. On the other hand, the NES had more voices and a particularly characteristic and 'lo-fi' triangle waveform, so it would be unproductive to designate one platform better than another. It makes no sense to discuss these sound chips or sound-generation techniques as being better than one another, any more than it would make sense to have that debate about a guitar, flute or acoustic drum kit. That these instruments are markedly and audibly different is the key to understanding the call to disentangle the chips from the tunes. Examining them in this way, we begin to reveal not only some of the significant ways in which the gaming sound chips differ from one another but also how these differences materially affect the sonic fingerprint of the games created for the host system.

'Chiptunes' and the Oversimplification of the Sound of Sound Chips

While it is essential to recognize the distinctive nature of different sound chips and move away from the monolithic nature of *chip*tunes, it is important that we do not overstate the role of the chip in affecting the sound of a given gaming system. Of course, the SID chip's oscillators sound different to those of the NES. For that matter, different revisions of the SID chip sound different to one another, with the filters varying depending on the month of manufacture, just as the different iterations of the Game Boy and Mega Drive vary, with one version of the latter being labelled 'The Stinker'![15] However, although the SID Chip, RP2A0X, YM2612 and Paula all shape the sonic landscape of C64, NES, Mega Drive and Amiga respectively, the chips do not write the tunes. And many of the compositional, stylistic and sound-design techniques most commonly associated with chip music, or even with specific sound chips, are really better understood as interactions and improvisations between person and technology.

[15] Ace, 'GUIDE: A Complete Overview of the Many Sega Genesis/MegaDrive Models, Sega-16', *SEGA-16 Forum*, 2 July 2009, accessed 8 April 2020, www.sega-16.com/forum/showthread.php?7796-GUIDE-Telling-apart-good-Genesis-1s-and-Genesis-2s-from-bad-ones.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

By way of example, the ability of the Amiga's sound chip, Paula, to play back samples, is certainly a distinctive, if not wholly unique feature of the platform. Paula's ability to replay samples certainly affords a different kind of compositional and sound-design style, but it is only one aspect of the characteristic sound of the Amiga. Those snippets of audio are replayed at a specific sample rate and bit depth, which gives them a characteristic, grainy, subjectively 'lo-fi' quality, especially when they are pitched away from the root note. This is also affected by the quality of the computer's DAC (Digital Audio Converter), which is responsible for outputting the sound in a form capable of being replayed to a TV set, speakers or headphones (indeed, it is here that the audiophile might take exception with Altice's point about CD players not having a distinct sound). Each step of the audio pathway imparts its own distortions and colouration, some pleasing, some intended, but all present.

But even this extension of the 'platform' to encompass the audio format conversions fails to recognize the brute fact that Paula doesn't create music. Paula's ability to replay samples might strongly imply a compositional workflow based around small snippets of sound acquired from original recordings, sound libraries or from other pieces of music, and this workflow might have much in common with the cut-up techniques of Steve Reich or the burgeoning hip-hop scene. However, whether Paula plays back the sound of a barking dog or a series of samples derived from Roland's (1987) D-50 synthesizer (as in the case of Andrew Barnabas's *SWIV* soundtrack, 1991) is the result of a complex series of aesthetic and technical decisions arrived at through the dialogue between composer, programmer and chip. The compositional decisions, solutions and compromises are negotiated with the affordances of the sound chip, the architecture of the system as a whole, and the amount of memory and processing resources afforded to the sound program by the development team. Interestingly, the availability of samples gives the video game composer access not only to a broader sonic palette but, as we hear with *SWIV*'s 'Decimation' theme, access to a sound palette immediately familiar from popular music. Laced with the sounds of the Roland D-50, *SWIV* is audibly connected to the industrial soundscapes of Jean-Michel Jarre's influential 1988 *Revolutions* album, for instance.

Yet, there is considerably more to explore than spec sheets and more to chiptunes than chips. Not only does each chip have an identifiable musical/technical 'character', but also each composer/programmer bestows upon it a unique and equally identifiable personality born of specific compositional, sound design and technological interactions and intentions. In

discussing her work on early Namco arcade games, Junko Ozawa notes how foundational this personalization can be, by pointing to her own library of hand-drawn waveforms meticulously laid out on squared paper.[16] Like the Amiga composer's *bricolage* of samples drawn from countless sources, these waveforms operate in dialogue with the sound chip's playback capabilities to define the sonic territory available to the composer. Ozawa's stamp of individuality is evident in her personalized waveforms.

Other composers forged their own distinct paths by rejecting the typical approaches to writing music for a particular chip. The NES's triangle waveform is often used to take on bassline duties, as we hear in the case of the overworld theme in *Super Mario Bros.* (1985). The triangle wave is suited to this function because of a lack of control over its amplitude (thereby offering fewer opportunities for dynamics) and its pitch range extends down to 27.3 Hz, unlike the two pulse voices that bottom out at 54.6 Hz. Yet, some composers challenge convention by utilising the triangle wave in a wholly different way. By rapidly modulating the pitch of the triangle, the single voice can do double duty: it can alternate between a pitched bassline note and a kick drum. When the drum note is reinforced with a burst from the chip's noise channel, the result is heard as combination of bass, kick and percussion. The effect was used extensively by Neil Baldwin (*Hero Quest*, 1991) and Tim Follin (*Silver Surfer*, 1990). Composers devised their own innovative ways of writing for the technology, resulting in a diversity of approaches to composing for even one type of chip.

Some composers took this approach even further. Some SID chip composers, such as Rob Hubbard, would use one chip channel to play multiple voices, and/or blend chip channels to create new hybrid sounds. For example, in *Commando* (1985), to create his percussion sounds, Hubbard changes the waveform of a channel every fiftieth of a second to create a single sound that combines a burst of noise followed by a swooping pitched 'tom' effect. Even though the SID chip only has three channels, Hubbard's *Commando* score does not limit itself to three discrete musical parts. Where the listener may hear the rhythm track as a single 'part', it is actually written and performed across multiple voices. Sometimes voice two performs a 'drum' hit and elsewhere the same sound is played on voice three, for example. This happens because, depending on the nature of the composition, voice two is

[16] RBMA (Red Bull Music Academy), 'Diggin' in the Carts: Episode 1: The Rise of VGM', *Red Bull Music Academy*, 2014, accessed 8 April 2020, http://daily.redbullmusicacademy.com/2014/10/diggin-in-the-carts-series.

not always available and may be 'busy' playing the bassline or countermelody. It is clear from analysing the *Commando* theme that it is not a piece of music simply transcribed for the SID chip but, rather, is written with the specific capabilities and affordances of the chip in mind.[17] The piece's thick sonic texture is further enhanced by the frequent use of ornamentation characteristic of Hubbard's work. *Commando* only uses three channels, but by continually shifting the musical material across the three voices, making space to pack in other musical elements, it gives the listener the impression of more than three voices being present.

Martin Galway's soundtrack to *Wizball* (1987), by contrast, uses similar processes for different ends. Here, Galway doubles and offsets melodies across voices to simulate a delay or echo effect (common in recording-studio outboard racks but absent from the SID's specification).[18] What we hear in Galway and Hubbard's work, then, is as much a function of the SID chip as it is a reflection of their individual compositional characters. And so, when Guay and Arsenault note a tendency in 'chiptunes' towards busy 'baroque' ornamentation, this holds true only for some composers of early video game music.[19]

If we look to the NES, we see similar variations in musical style and form that are best read as reflections of differing artistic and aesthetic intentions and sensibilities. Koji Kondo's work on the *Super Mario Bros.* and The Legend of Zelda series takes a very particular approach to the RP2A0X sound chip. Unlike the work of Hubbard, whose extreme multiplexing sought to create the impression of a greater number of voices than were actually sounding, Kondo's NES themes are stripped back. Not bare or minimalistic, by any means, but neither disguising nor shying away from the presentation of a four-part composition of bassline (triangle), melody and countermelody (pulses) and percussion (noise). For McAlpine, Kondo's technique in *Super Mario Bros.* has its antecedents in the 'shell voicing' technique used by Art Tatum and Bill Evans that implies complex harmonies with just a few notes, and the syncopations and polyrhythms that are not entirely dissimilar to the rhythmic devices employed in Debussy's first *Arabesque*.[20]

[17] James Newman, 'Driving the SID Chip: Assembly Language, Composition, and Sound Design for the C64', *GAME: The Italian Journal of Game Studies* 6 (2017).

[18] Neil Baldwin, 'James Bond Jr (Eurocom/THQ 1991)', *DutyCycleGenerator*, 29 March 2009, accessed 8 April 2020, https://web.archive.org/web/20200325074309/http://dutycyclegenerator.com:80/.

[19] Louis-Martin Guay and Dominic Arsenault, 'Thumb-Bangers: Exploring the Cultural Bond Between Video Games and Heavy Metal', in *Heavy Metal Generations*, ed. Andy Brown and Kevin Fellezs (Oxford: Interdisciplinary Press, 2012), 105–15.

[20] McAlpine, *Bits and Pieces*, 120–1.

Of course, music is typically not the only sonic output of a game, but has to co-exist with sound effects in the game's soundscape. One question we might raise in relation to Kondo's *Super Mario Bros.* theme is why the jumping sound effect, triggered as Mario takes flight and heard with some frequency, should cut out the main melody line. The technical answer is simple. With only four channels available for music and sound effects, something has to give. Kondo's was but one of a number of strategies deployed throughout the 1980s and 1990s to combine music and 'interactive non-diegetic sound' as Collins puts it.[21] Some games reserved voices on the chip exclusively for sound effects (*The Human Race*, Mastertronic, Commodore 64, 1985), while others asked players to choose between music and sound effects (*Delta*, Thalamus, Commodore 64, 1987). But why replace the lead line rather than the countermelody? One answer, favoured by McAlpine, connects the pitch of the jump sound effect with the key of the game's soundtrack.[22] As such, the harmonic integrity of the soundscape (if not the melody) is preserved and the sound effect is integrated into the music, joining the visual, sonic, haptic and ludic aspects of the game into one experiential whole.

What we see in these examples is not simply the result of different musicians imprinting themselves on an instrument through their differing sensibilities. Rather, the different approaches become encoded in the distinct software routines that are used to create and craft music.

Although it is tempting to view the 1980s or 1990s console, home computer or arcade-board sound chip as a synthesizer, if we consider the interfaces by which these sonic potentialities were accessed, they have little in common with the commercially available instruments used by musicians on stage and in studios at the time. The SID chip, for instance, presents itself as a series of data registers that can be read and written to. There are no inviting faders, sliders, potentiometers, or pitch and modulation wheels to be seen. All sound design is undertaken by accessing these data registers in hexadecimal. Similarly, in the 1980s, no visual compositional tools existed to help write, arrange or edit sequences of game music. Any tools had to be created for the task. Some musicians created their own bespoke sound design, composition and playback software routines – or 'drivers' as they are typically called – while others relied on the services of coders. Either way, these sound drivers play an immense role in defining the sound of the sound chip. These routines reveal, or make available, facets of the

[21] Karen Collins, 'In the Loop: Creativity and Constraint in 8-bit Video Game Audio', *Twentieth Century Music* 4, no. 2 (2007): 209–27, at 212.

[22] McAlpine, *Bits and Pieces*, 123.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

sound chip's capability.[23] As Hubbard notes of his *Monty on the Run* (1985) composition, 'The middle section was an excuse to use the new pitch bend code that I wrote for this project.'[24] Similarly, Martin Galway, in-house composer at Ocean Software, describes how he was 'mastering the C64' as he was developing his drivers.[25]

Technical capabilities are transformed into musical utilities by the software driver. For instance, the SID chip's technical capability to stably control the pitch of its oscillators over a wide frequency range or to continuously vary the duty cycle of its pulse waves become the musical operation of *portamento*. A driver includes and omits musical and sound design features in accordance with the aesthetic judgement and predilections of the composer and sound designer. As new affordances are revealed, these features can be added to the driver to present them in a musically useful manner. Sometimes these are features unintended and unanticipated even by the creators of the sound chips themselves, as in the case of the SID chip's sample playback capability.

To speak of the sonic characteristic of a given sound chip is to tell only part of the story. Without sound driver routines written in software and acting as a mediating interface between the composer and silicon, the sound chip's features remain musically inaccessible data points on a specification sheet. As Chris Abbott observes, 'Rob Hubbard sounded very different from Martin Galway because they had to write their own synthesizer engine, as well as the music.'[26]

Perhaps the most persuasive example of the significance and personality of the sound driver in mediating and shaping the interface between musician and chip is found in the rapidly arpeggiating pseudochord. This musical device must surely rank as one of the most instantly recognizable features of 'chiptunes'. It consists of an arpeggio of two or more notes played at a speed so high that the individual tones almost appear to meld into a single chord. Almost, but not quite, which is what gives the figure a warbling or chirping effect a little like a mobile telephone ring, as the individual tones and their transients are still identifiable.

[23] For an analysis of Hubbard's early driver routine see Anthony McSweeney, 'Rob Hubbard's Music: Disassembled, Commented and Explained', *C=Hacking* 5 (1993), accessed 8 April 2020, www.ffd2.com/fridge/chacking/c=hacking5.txt.

[24] Andreas Wallström, 'Rob Hubbard Interview', *C64.com* (n.d.), accessed 8 April 2020, www.c64.com/interviews/hubbard.html.

[25] Andreas Wallström, 'Martin Galway Interview', *C64.com* (n.d.), accessed 8 April 2020, www.c64.com/interviews/galway_part_2.html.

[26] Flat Four, *Programme 3: Commodore Music*, Flat Four Radio, 2005, accessed 8 April 2020, www.mcld.co.uk/flatfour/chiptunes/commodore/.

The key here is that the design and operation of the sound driver for a video game are intimately related to those of the system processor, game program and TV standards. The sound driver is just one of the processes requiring access to the finite computing resources of the system. If we take the case of the typical 1980s console or home computer game, as graphics need to be drawn on screen fifty or sixty times a second (depending on the exact system and international TV specification) to create the impression of continuous, smooth motion, other elements of the program, including the sound driver, can run only when the processor is not busy undertaking these tasks. With the driver written to issue new instructions to the sound chip fifty times a second, this sets the maximum rate of change for the opening or closing of a filter, the adjustment of a waveform duty cycle or the pitch of a note. The effect of this is that the rapidity of the rapid arpeggiations that comprise the pseudochord is dictated not by any musically significant or even musically derived division, but rather by the driver's maximum speed of operation. Importantly, the maximum speed of arpeggiation does not necessarily reflect the extent of a sound chip's capability, and serves to underscore the significance of the driver in shaping the available features.

Yet, while arpeggiation has become almost emblematic of an entire genre of music, it is not endemic to any particular chip nor, indeed, is its use a product of or limited to sound chips. Rapid arpeggiation has been evident in monophonic music for many centuries, but here it is accelerated beyond the ability of even the most dexterous human player. More importantly, because this rapid changing of pitch is not a feature of the sound chip that is simply enabled or disabled, but rather is a function of the sound driver routine, its design and implementation is as personal and distinctive as any other sound-design or compositional device we might identify. So, where some composers use three- or even four-note arpeggios, some, such as Hubbard, often use just two. Indeed, where a composer writes just music with no need for graphics, game logic or user input, a sound driver may be written to run at a higher rate thereby offering increased resolution of sound manipulation and note retriggering. As such, though the pseudochord is characteristic of early game music, its implementation is always coloured by the driver and composer in question.

In addition to its flattening effect, then, the chiptune soubriquet is also problematic because it goes beyond setting focus on technology to giving it primacy. As such, the interactions and creative endeavours of the people who design the chips and the sound drivers and those who ultimately shape

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

the technology are silenced. A focus on the machinery of music making is key to helping us avoid the tendency to generalization rightly identified by Altice. However, we must also exercise caution by ensuring that we do not separate the sound chip from the contexts of its design and use and inadvertently create a deterministic account that is deaf to the ways in which technologies are shaped. Similarly, by focusing on the ways in which the particular combinations of design decisions, technical and musical opportunities, quirks and affordances are revealed, harnessed and exploited in different ways by different composers and programmers, we offer ourselves an opportunity to account for the often chaotic and haphazard manner in which these investigations, innovations and musical inventions are arrived at.

Teleology, Linearity and the Tyranny of Limitations

Both popular and scholarly accounts of game music history display a tendency to centre on the inevitability of technological progress or evolution. These teleological approaches are based upon two interrelated principles. First, that there is an implied perfect or ideal state which this inexorable forward motion is heading towards. Second, that less advanced stages along this journey are characterized by limitations, which are progressively eradicated and rendered immaterial.[27]

Histories of early video game music typically present an evolutionary narrative that is both teleological and demonstrably written from a backward-facing perspective in which the limitations are evident given the bounty of what is now known to have followed. As such, rather than offer a study of early video game music and sound chips in terms of the ways in which foundational practices of design, composition and technique were established, they typically serve the purpose of a (brief) contextual 'prehistory' for contemporary practices. Popular accounts are more explicit in their conceptualization of progression in their titles. Both BBC Radio 3 (2018) and NPR (2008) have produced programmes entitled *The Evolution of Video Game Music*, for instance.

The orthodoxy of the technological timeline begins with the silence of early video games such as Steve Russell's *Spacewar!* (1962) before indefatigably

[27] Damian Kastbauer, 'Using Memorable, Iconic Sounds in Video Games', *Gamasutra*, 30 April 2013, accessed 8 April 2020, www.gamasutra.com/view/news/191426/Using_memorable_iconic_sounds_in_video_games.php.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

moving along the pathway to the present, noting the series of significant milestones along the way. Early milestones typically include games such as Taito's coin-operated *Space Invaders* (1978), noted for including the first continuously playing soundtrack. This consisted of an ostinato pattern of four descending notes played in the bass register, and the tempo increased as the player cleared more invaders, making it the first example of interactive, or perhaps more accurately reactive, audio. For the first instance of a continuous melodic soundtrack, our attention is drawn to Namco's 1980 *Rally-X*, while Konami's 1981 *Frogger* added introductory and 'game over' themes linking music to the game state. The evolution of arcade sound is often articulated in terms of new chips (including the introduction of the numerous variants based around Yamaha FM or Frequency Modulation synthesis method), or the inclusion of greater numbers of sound chips working in parallel to provide increased polyphony (Konami's 1983 *Gyruss* included five of General Instrument's AY-3–8910 chips, for instance).

For home gaming, the evolutionary narrative is typically mapped to the successive 'generations' of consoles which move from the discordance of the Atari 2600's TIA, through the SID chip's synthesis, FM (e.g., Sega Mega Drive/Genesis which actually used a combination of Yamaha YM2612 along with the Texas Instruments SN76489 from the earlier Master System console), and sample playback (the Nintendo SNES/Super Famicom's Sony-designed S-SMP allowed 8 channels of 16-bit samples).

In the domain of general-purpose home computers, while it is ironic to note that the Atari ST was a fixture of commercial music recording studios,[28] it is the Commodore Amiga that is more commonly identified as key in the evolution of game music. This accolade arises from the development of the 'Tracker' software sequencers we noted above and the associated 'MOD' file format, which brought some degree of compatibility and portability.

In contrast, the PC-compatible games marketplace was characterized by a variety of competing sound cards such as the AdLib, Creative's Sound Blaster and peripherals such as the Roland MT32, offering differing playback quality, sound libraries and available effects (the MT32 included reverb, for instance). Where consoles provided a stable platform, the variety of potential PC-compatible sound cards created a situation whereby music might sound markedly different from one system to the next. In the

[28] Matt Anniss, 'Instrumental Instruments: Atari ST', *Red Bull Music Academy Daily*, 6 October 2017, accessed 8 April 2020, http://daily.redbullmusicacademy.com/2017/10/atari-st-instrumental-instruments/.

case of a game like LucasArts' *Monkey Island* (1990), although the PC speaker version is recognizably the same melody, when this is compared with its playback on a Roland SCC-1 card, it is rather easier to note the differences than the similarities.

Aside from the inevitable magnitude of the task, one issue we might immediately have with the timeline approach arises from the fact that, while games and systems unquestionably provide useful sites for investigation, their milestones entangle aesthetic, creative and technological endeavour and innovation. Nonetheless, such narratives remain powerfully seductive, not least because they chime neatly with the dominant discourses identifiable elsewhere in writing on video game history and, indeed, as deployed by the video games industry through its marketing and advertising messages.

As I have suggested elsewhere,[29] this narrative is not merely one that implies the existence of a continuum with its imaginary end point of gaming perfection, but is also one predicated on the supersession of successive generations of hardware. Simply put, each platform generation embarks on a journey beginning with its construction as mythical object of speculation, anticipation and desire through to an ultimate reconstruction as a series of weaknesses, limitations and failings that are rectified by its replacement. It becomes the baseline from which the next generation's performance is measured. Given the prevalence of this narrative and the ways in which new platforms and games are seen to improve upon their predecessors and render them obsolete, it is not surprising to find so much history of game music reproducing these formulations.

However, it is worth probing the efficacy of the conceit of limitations, particularly as its use in relation to early video game music is so prevalent that it seems almost to have become an uncontested truism. Much of what we have seen above, whether it be the use of the distinctive pseudochord or the waveform manipulation/multichannel drum design, can be explained as a response to the limitations of the SID chip and RP2A0X polyphony, just as Mario's occupation as a plumber can be explained as a response to the limitations of the NES colour palette and sprite resolution and the decision to clad him in overalls. On the other hand, these examples can equally be framed as manifest examples of the creativity and inventiveness of composers, programmers and character artists working within specific resource windows and working in dialogue with the technologies at their

[29] James Newman, *Best Before: Videogames, Supersession and Obsolescence* (London: Routledge, 2012).

disposal to shape their outcomes. That their creations, whether auditory or visual, appear to confound and exceed the boundaries of the 'limitations' is a testament to their ingenuity.

The incidence of looping is another area worthy of study in this regard. The construction of early video game music either as a single looping phrase, or as a series of nested loops, is for many commentators a defining feature of the form.[30] The reliance on looping is typically explained as a necessary compositional response to the limited availability of memory within which music data and driver code had to be stored. However, the tendency towards heavily looping sequences might also be read as a result of the iterative nature of compositional/coding practices.

> This repetitious methodology is reflected in the unique needs of game scoring. Unlike a film score, background music for computer games in this era was designed to loop endlessly along with highly repetitious game play.[31]

In fact, in this final point there is a hint that, like Kondo's integration of the jump sound effect into the melodic top line, repetition in early video game music might be read as aesthetically and functionally matched to the often repetitive, iterative nature of the gameplay it supported. The four notes of *Space Invaders* are undoubtedly efficient, but they also suit the relentlessly cyclical nature of the gameplay just as *Super Mario Bros.*' repeating phrases suit gameplay and levels comprising repeating graphical and structural elements.

More than this, however, we might also re-evaluate whether some of what are presented as limitations truly were so, or have ever been truly remedied. Returning to drum design, achieving this through the rapid manipulation and alteration of an oscillators' waveform is a remarkable feat of ingenuity in sound design and creative thinking. It is also made possible because the SID chip's design allows the waveform of its oscillators to be altered at audio rate. It is possible to achieve this because the sound driver software provides an interface to that data register and enables the composer and sound designer to manipulate the SID's output accordingly. Is this example best understood as limited polyphony, with voices and waveforms accessed through an unforgiving interface? We might well argue that the SID chip was far from limited. Indeed, while modern replications and emulations of sound chips (such as the SID Guts, AY3

[30] See Karen Collins, 'Loops and Bloops: Music of the Commodore 64 Games', *Soundscapes* 8 (2006b) and Collins, 'In the Loop'.

[31] Driscoll and Diaz, 'Endless Loop'.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press

and Edges) provide altogether more user-friendly and tactile interfaces to the underlying hexadecimal data registers, the nature of these interfaces and protocols means that they actually offer considerably less precise control. The sound driver's interface may have lacked the tactility of the contemporary hardware synthesizer, but it did enable the composer, programmer and sound designer to effectively write directly to the silicon. If for no other reason than this, our discussion of the role of technology would greatly benefit from a radical rethinking of the centrality of limitations and the teleological, evolutionary process that implicitly frames early video game sound as a prototype or underdeveloped iteration of what would inevitably later come.

In this chapter, I hope to have demonstrated the importance of keeping an awareness of technology at the heart of our consideration of early video game music while simultaneously guarding against the rehearsal of the deterministic, teleological discourses so dominant within scholarly, popular and industry accounts. In particular, I hope to have foregrounded the ways in which contemporary analytical frames such as the centrality of overcoming or working with 'limitations' or the construction of 'chiptunes' actually serve to hinder our ability to create nuanced accounts of early video game sound and music.

https://doi.org/10.1017/9781108670289.003 Published online by Cambridge University Press