Universität Osnabrück
Seminar: Einführung in die elektronische Musik
Seminar Nr.: 3.312
Dozent: Prof. Dr. Enders

Der Sound des Commodore 64

Elektronische Musik im Heimcomputerbereich

Verfasser: Wolfram Heyer
Studiengang:
Lehramt Musik für Gymnasium
7. Semester (Winter 2001/2002)
Adresse:
Wolfram Heyer, Wörthstraße 56, 49082 Osnabrück
Telefon: 0541 / 982 61 60, eMail: wheyer@uos.de

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Über diese Ausarbeitung

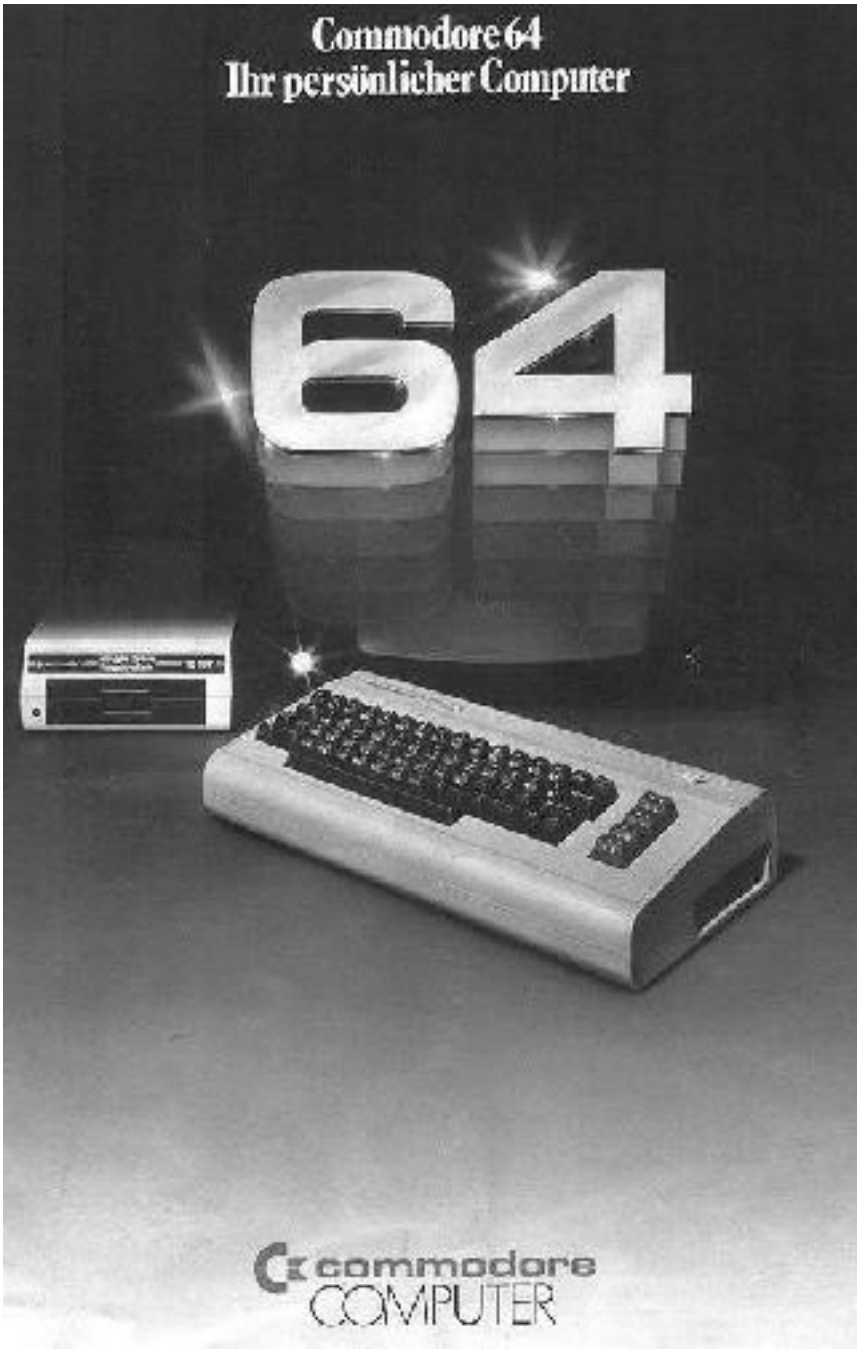

Abbildung 1: Eine Commodore C64 Werbung aus einer Zeitschrift

Diese Ausarbeitung meines Referats, daß ich im Rahmen des Seminars „Einführung in die elektronische Musik“ an der Universität Osnabrück gehalten habe, soll meine Vorführungen zum einen durch eine vollständige Auflistung der grundsätzlichen Möglichkeiten, die der C64 hinsichtlich des Sounds zu bieten hat, untermauern (Kapitel 2), zum anderen möchte ich einige alte und neue Musikprogramme vorstellen, mit denen man auf diesem Computer Klänge editieren oder zusammenstellen kann (Kapitel 3).
Der Commodore 64 war in den achtziger Jahren ein in Deutschland weit verbreiteter Heimcomputer, der zu einem erschwinglichen Preis die Funktionen vereinte, die seine Interessenten, meinst Computerneulinge, ansprachen. In erster Linie waren dieses wohl seine einfache Bedienung und seine für damalige Maßstäbe mehr als ausreichend ausgestatteten Grafik [1] und Soundmöglichkeiten, die die Softwareindustrie anspornte, eine Vielzahl von Computerspielen für den C64 auf den Markt zu bringen. Doch auch für den Anwender gab es immer ausreichend Software, wie zum Beispiel Textverarbeitungs-, Lern- oder Musikprogramme, und außerdem auch ein Betriebssystem mit grafischer Oberfläche, daß sich GEOS nennt.
Als die Firma Commodore schließlich den Verkauf ihres Produktrenners einstellen mußte, blieben, wie so oft, die Fans dem

[1] 160x200 Bildpunkte bei 16 Farben, hochauflösende Grafik mit 320x200 Bildpunkten bei 2 Farben

C64 treu. Was mit kleinen Vorspännen der illegalgen Crackergruppen, den sogenannten „Intros“ (Introductions) vor den Spielen angefangen hatte, begann ein Eigenleben. Die Intros wurden immer länger und aufwendiger gestaltet, sie wurden zu einer eigenen Form, den „Demos“ (Demonstrations). Noch heute konkurrieren neue und alte Gruppen von damals in einem fairen Wettkampf, wer die besten Demos in der „Scene“ herausbringt, denn die Hardwarevoraussetzungen sind unabänderlich für alle gleich. Aus diesem Grund gibt es auch heute noch eine Vielzahl von neuen Demos, Spielen und vor allem auch Anwendungsprogrammen, den sogenannten „Tools“, die dem Programmierer, Grafiker oder Musiker die Arbeit erleichtern. In der heutigen Zeit der PCs und des Internets wurden natürlich auch Mittel und Wege gefunden, mit denen man in kurzer Zeit Daten vom PC zum C64 und umgekehrt transferieren kann.
Da es also von der C64-Hardwareseite nichts Neues zu berichten gibt, habe ich mich bemüht, die Voraussetzungen und Möglichkeiten, die der Commodore 64 hinsichtlich des Klanges bietet, möglichst genau und vollständig zu beschreiben. Im dritten Kapitel, in dem ich auf die Software eingehe, möchte ich alte und neue Programme vorstellen, mit denen ich gearbeitet habe oder heute noch arbeite. Ich wünsche viel Vergnügen!

2 Beschreibung des SID

2.1 Einführung

Der C64 verfügt über einen für damalige Maßstäbe hochwertigen Soundbaustein, den SID (Sound Interface Device) [2]. Dieser Baustein besteht aus 3 getrennten Tongeneratoren mit einem Frequenzbereich von 0 bis ca. 4 kHz. Die Wellenform sowie auch die Hüllkurve jedes Tongenerators können getrennt gewählt werden. Bei der Wellenform hat man die Wahl zwischen Dreieck, Sägezahn, Rechteck und Rauschen. Für jeden Generator stehen sieben Register zur Verfügung, die bei jedem Tongenerator die gleiche Bedeutung haben. All diese Register, bis auf die des Tongenerators 3, der eine Sonderstellung einnimmt, sind „write only“ (nur beschreibbar). Zusätzlich steht noch ein Filter zur Verfügung, mit dem es möglich ist, sowohl die Tongeneratoren als auch eine externe Signalquelle zu verändern. Mit dem dritten Tongenerator ist es unter anderem möglich, Zufallszahlen zu erzeugen, weil er als einziger auch auslesbar ist.

2.2 Registerüberblick

Basisadresse = $D400 [53272]

- Stimme 1:
  00 : Frequenz Low
  01 : Frequenz High
  02 : Pulsbreite Low
  03 : Pulsbreite High
  04 : Steuer-Register [z.B. Wellenform]
  05 : Attack / Decay
  06 : Sustain / Release
- 07 - 14 : Stimme 2 (siehe 00-06)
- 15 - 20 : Stimme 3 (siehe 00-06)
- 21 : Filter Low
  22 : Filter High
  23 : Filter-Resonanz-Schalter
- 24 : Gesamtlautstärke / Filterart
- 25 : A/D Wandler 1
  26 : A/D Wandler 2
- 27 : Oszillator 3
- 28 : Hüllkurvengenerator 3

2.3 Frequenz

Die ersten beiden Register geben die Frequenz des Tons an. Sie wird als 16-Bit Zahl gespeichert. Zur Erzeugung der gewünschten Frequenz muß diese in eine an den C64 angepaßte Zahl umgerechnet werden. Diese Zahl hängt von der Taktfrequenz des Computers ab, die bei dem amerikanischen Modell höher ist als bei dem europäischen. [3]
Die Taktfrequenz läßt sich errechnen, indem man die Anzahl der Schwingungen des Quarzes durch 18, beziehungsweise durch 14, teilt:

- Europäische Version: 17734472 Hz / 18 = 985,2484 kHz
- Amerikanische Version: 14318180 Hz / 14 = 1022,7271 kHz

Der vorerwähnte 16-Bit-Wert setzt sich wie folgt zusammen:

- WERT = Frequenz * 2^{24} / Taktfrequenz

Der so errechnete Wert muß jetzt lediglich in Low- und High-Byte (beides in Hexadezimal) aufgespalten werden:

- HI = INT (WERT/256)

[2] Der SID mit der Nummer 6581 wurde in den frühen C64 eingebaut. Die neueren Modelle (flaches Gehäuse) enthielten den SID 8580, der eine niedrigere Spannung benötigt. Der SID 6581 wurde außer im C64 noch in anderen Computern wie z.B. dem C128, C720 oder natürlich dem portablen C64, dem SX64, eingebaut.

[3] Da im amerikanischen Raum NTSC und im europäischem Raum PAL als Standart für Fernsehapparate gilt, wurde dem C64 die Möglichkeit eingebaut, auf Softwareebene zwischen diesen Bildwiederholfrequenzen umzuschalten. Aus diesem Grund muß die Software immer dem jeweiligen System angepaßt werden.

- LO = WERT - 256 * HI

Durch Änderung des Wertes um eins erreicht man eine Tonfrequenzänderung von ca. 0.06 Hz. Die Variable „Frequenz“ wird folgendermaßen berechnet:

- Frequenz = $2^{(TonNummer/12)}$ * 16,35

Die Tonnummern sind die fortlaufenden Halbtonschritte ausgehend vom tiefsten Ton („C-0“, siehe Abbildung 2), der Faktor 16,35, auf den Bezug genommen wird, ist die Frequenz dieses tiefsten Tons.

In der Praxis hat es sich durchgesetzt, daß man vorkalkulierte Tabellen (Abbildung 2) für die Frequenzzuweisungen benutzt, da eine Berechnung in Echtzeit die ohnehin begrenzten Resourcen des C64 nur zusätzlich belasten würde.

Werte für Musiknoten				
Nr.	Note-Oktave	Frequenz	High-Byte (Hex)	Low-Byte (Hex)
0	C-0	16.4	$01	$16
1	C#0	17.3	$01	$27
2	D-0	18.4	$01	$39
3	D#0	19.4	$01	$4b
4	E-0	20.6	$01	$5f
5	F-0	21.8	$01	$74
6	F#0	23.1	$01	$8a
7	G-0	24.5	$01	$a1
8	G#0	26.0	$01	$ba
9	A-0	27.5	$01	$d4
10	A#0	29.1	$01	$f0
11	H-0	30.9	$02	$0e
12	C-1	32.7	$02	$2d
13	C#1	34.6	$02	$4e

Abbildung 2: Ausschnitt aus einer Frequenztabelle

2.4 Wellenformen

Für die Auswahl einer der möglichen Wellenformen (Dreieck, Sägezahn, Rechteck und Rauschen) sind die obersten 4 Bits des Registers 4 jedes Tongenerators zuständig. Es ist nicht möglich, mehrere Wellenformen zu mischen. Auf die anderen Funktionen dieses Registers werde ich später noch eingehen.

Durch Setzen eines der Bits wird die entsprechende Wellenform ausgewählt:

- Bit 4 [%00010000]: Dreieckwelle

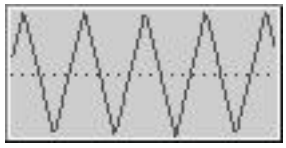

Abbildung 3: Dreieckwelle [triangle]

- Bit 5 [%00100000]: Sägezahnwelle

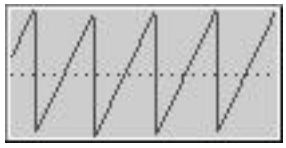

Abbildung 4: Sägezahnwelle [sawtooth]

- Bit 6 [%01000000]: Rechteckwelle

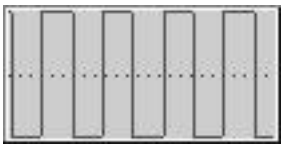

Abbildung 5: Rechteckwelle [pulse]

- Bit 7 [%10000000]: Rauschen

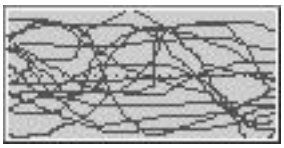

Abbildung 6: Rauschen [noise]

Wird die Rechteckwelle gewählt, so ist es zusätzlich möglich, die Pulsbreite zu variieren. Zur Festlegung der Pulsbreite dienen die Register 2 und 3 des jeweiligen Tongenerators. Von dem sich aus den zwei Registern ergebenden 16-Bit-Werte werden jedoch nur die untersten 12 Bits benutzt. Hieraus ergibt sich, daß der höchste Wert, der auf die Pulsbreite Einfluß hat, $0fff (4096) ist. Extrem kleine oder extrem große Werte erzeugen keinen Ton. Wenn man in diese Register den Wert $07ff (2047), der der Hälfte des Maximums entspricht, schreibt, erhält man eine vollkommen regelmäßige Rechteckschwingung.

Die Dreieckwelle entspricht in etwa einer Sinusschwingung und eignet sich zum Imitieren sämtlicher Zupfinstrumente.

2.5 Hüllkurve

Mit Hüllkurve (Abbildung 7) ist nicht anderes als der Lautstärkeverlauf des zu erklingenden Tones gemeint. Durch dessen Veränderung ist es möglich, den Ton noch weiter zu beeinflussen. Der Lautstärkeverlauf ist in vier verschiedene Abschnitte unterteilt. Es wird unterschieden zwischen:

- Attack

  Das Spielen eines Tones beginnt mit der Attack-Phase. Es ist jedoch nötig, dem SID mitzuteilen, wann er beginnen soll, den Ton zu spielen. Hierfür existiert Bit 0 des Registers 4 des jeweiligen Tongenerators. Sobald dieses sogenannte Key-Bit gesetzt wird, fängt der Ton an zu spielen und beginnt mit der Attack-Phase. In dieser Phase steigt die Lautstärke von Null auf die im Low-Nibble des Registers 24 angegebene Gesamtlaustärke. Die Zeit, in der dies geschieht, wird im High-Nibble des Registers 5 der jeweiligen Stimme angegeben. Der Zeitraum kann zwischen 0.002 Sekunden und 8 Sekunden liegen. Hohe Werte ergeben eine lange Attack-Phase. Die Attack-Phase wird immer eingeleitet, auch wenn der Ton noch nicht am Ende der weiter unten besprochenen Release-Phase war.

- Decay

  Nach Abschluß der Attack-Phase kommt die Decay-Phase. In ihr ändert sich die Lautstärke von der eben erreichten bis auf die im High-Nibble des Registers 6 eingestellte. Die dafür benötigte Zeit wird im Low-Nibble des Registers 5 angegeben. Sie ist einstellbar zwischen 0.006 Sekunden und 24 Sekunden.

- Sustain-Spanne

  Nachdem die neue Lautstärke erreicht wurde, bleibt sie konstant. Ohne einen Eingriff würde der Ton ewig mit der gleichen Lautstärke spielen. Es wird erst die nächste Phase erreicht, sobald das Key-Bit, das wir zu Beginn der Attack-Phase gesetzt hatten, gelöscht wird. Die Länge der Sustain-Spanne ist nämlich in keinem Register einstellbar. Sie muß durch Ablaufen einer Zeitschleife festgelegt werden. Sobald das Key-Bit gelöscht wird, wird umgehend die Release-Phase erreicht, selbst dann, wenn der Ton sich erst in der Attack-Phase befand.

- Release

  In dieser Phase fällt die Lautstärke von der bei Decay angegebenen Lautstärke auf Null zurück. Die dafür benötigte Zeit wird im Low-Nibble des Registers 6 angegeben. Sie ist wie bei Decay zwischen 0.006 Sekunden und 24 Sekunden wählbar.

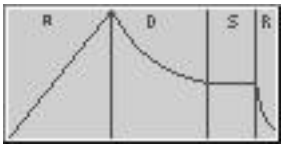

Abbildung 7: Beispiel für eine ADSR Hüllkurve

2.6 Tonerzeugung

Will man die Erzeugung eines Tones programmieren, so muß als erstes eine Grundlautstärke, die für alle drei Stimmen gilt, gewählt werden. Sie liegt im Bereich von 00 (nicht hörbar) bis 15. Dieser Wert muß in den Low-Nibble des Registers 24 geschrieben werden. Daraufhin müssen alle die anderen bereits aufgeführen Parameter wie Frequenz, Hüllkurve und Wellenform festgelegt werden. Nachdem dies geschehen ist, muß dem SID noch mitgeteilt werden, daß er den eben gewählten Ton zu spielen beginnen soll. Dies wird erreicht, indem Bit 0 des Registers 4 (Key-Bit) gesetzt wird. Dadurch schaltet der SID die Attack-Phase (Anklingphase) ein, und es erklingt ein Ton, der nach der angegebenen Zeit (Register 5) die angegebene Gesamtlautstärke (Register 24) erreicht. Da dieses Bit [4] also in demselben Register zu finden ist, das auch für die Wellenform zuständig ist, wird beides - am Ende der Parametereinstellungen - am geschicktesten zugleich erledigt.

Ein BASIC-Programm zur einfachen Tonerzeugung ohne Filter könnte so aussehen:

```
10 SID = 53272     : REM BASISADRESSE SID
20 POKE SID+ 5,194 : REM ATTACK / DECAY
30 POKE SID+ 6, 90 : REM SUSTAIN / RELEASE
40 POKE SID   ,180 : REM FREQUENZ LOW (C-3)
50 POKE SID+ 1,  8 : REM FREQUENZ HIGH (C-3)
60 POKE SID+ 4, 33 : REM SAEGEZAHN EIN (KEY-BIT)
70 FOR N = 1 TO 1400:NEXT N : REM WARTEN
80 POKE SID+ 4, 32 : REM SAEGEZAHN AUS (KEY-BIT)
```

Erläuterungen:

- Der Wert 194 (%11000010) in Zeile 20 wird aufgeteilt. Attack wird das High-Nibble (%1100 = 12) und Decay das Low-Nibble (%0010 = 2) zugewiesen. Das Resultat ist ein langsames Anklingen des Tones gefolgt von einer sehr abrupt einsetzenden Decay-Phase.
- Gleichermaßen wird der Wert 90 (%01011010) in Zeile 30 auf Sustain und Release aufgeteilt. Die neu zu erreichende Lautstärke (Sustain = %0101 = 5) und die Release-Zeit (Release = %1010 = 10) sind somit gesetzt.
- Die Warteschleife in Zeile 70 dient zur Überbrückung der Zeit, die der Ton aufgrund der angegebenen Parameter braucht, um die Gesamtlaustärke zu erreichen und um daraufhin auf die Lautstärke, die in Register 6 angegeben ist, abzufallen.
- Die restlichen Anweisungen erklären sich durch die Anmerkungen im Quelltext.

2.7 Filter

Der SID verfügt, wie bereits erwähnt, über einen Filter, mit dem es ermöglicht wird, die Tongeneratoren und eine externe Quelle zu verändern. Das externe Signal wird über die Audio-In Leitung an den Computer übergeben.

Die Filterfrequenz ist in den Registern 21 und 22 festgelegt. Von den insgesamt 16 Bits der beiden Register werden lediglich 11 benutzt, denn von Register 21 sind nur die Bits 0 bis 2 vom SID ansprechbar. Der Wert, der in die beiden Register geschrieben wird, muß erst aus der Filterfrequenz erechnet werden, die sich wie folgt zusammensetzt:

- Wert = (F-30)/5.8182 Hz

F ist dabei die Filterfrequenz angegeben in Herz.

Nachdem die Frequenz angegeben wurde, muß festgestellt werden, welcher Teil der Tonfrequenz gefiltert werden soll. Dies kann man mit den Bits 4 bis 6 des Registers 24 einstellen:

- Bit 4: Lowpass

  Ist dieses Bit gesetzt, so werden alle Frequenzkomponenten über der eingestellten Filterfrequenz mit 12 dB je Oktave abgeschwächt.

[4] Key-Bit: Bit 0 des Registers 4

- Bit 5: Bandpass

  Ist das Bit gesetzt, so werden alle Frequenzen ungleich der eingestellten Filterfrequenz mit 6 dB je Oktave vermindert.

- Bit 6: Highpass

  Dieses Bit entspricht Bit 4 des Registers, nur daß all Frequenzen unterhalb der Filterfrequenz mit 12 dB je Oktave verringert werden.

Es ist möglich, verschiedene Filterarten zu kombinieren. Durch eine Kombination des Low- und Highpassfilters kann die Tonfrequenz zwischen den Werten eingeschachtelt werden (Bandsperre).

Abschließend sei noch die Veränderung der Filterresonanz erläutert. Sie wird im Register 23 festgelegt und kann einen Wert von 0 bis 15 annehmen, wobei ein hoher Wert eine hohe Resonanz hervorruft und ein niedriger Wert eine niedrige Resonanz. Somit erreicht man eine Betonung der Frequenzkomponenten, mit der sich durch Veränderungen während des Abspielens vielfältige Effekte erzielen lassen.

2.8 Tongenerator 3

Der Tongenerator 3 nimmt eine Sonderstellung im Vergleich zu den übrigen Tongeneratoren ein. Seine Register sind als einzige lesbar und daher sind bei ihm der Verlauf der Hüllkurve und der Zustand des Oszillators feststellbar.

- Hüllkurve

  Die Hüllkurve des Tongenerators 3 kann im Register 28 ausgelesen werden. Ihr Wert gibt die momentane Lautstärke des Tons an. Der Wert steigt bis auf 255 an, wenn die im Register 24 gesetzte Gesamtlautstärke erreicht ist. Danach fällt der Wert wieder und erreicht nach Beendigung der Release-Phase den Wert 0.
  Diesen Umstand kann man sich zu Nutze machen, wenn man beispielsweise eine fest eingestellte Sustain-Spanne unabhängig von der Länge der Attack-Phase haben will. Man liest das Hüllkurven-Register aus und wartet, bis er den Wert 255 erreicht. Von diesem Punkt an läßt man eine Zeitschleife ablaufen, um nach ihrem Ablauf das KEY-Bit zu löschen.
  Mit Hilfe dieses Registers kann auch erkannt werden, wann die Release-Zeit verstrichen ist.
  Weiterhin besteht die Möglichkeit, in Abhängigkeit des Zustands der Hüllkurve die Frequenz oder die Pulsbreite während des Tonsignals zu verändern und somit eine Reihe von wirkungsvollen Effekte zu erzielen.

- Oszillator

  Dies ist das Register 27, mit welchem es möglich ist, die 8 höchstwertigen Bits des Oszillators 3 abzufragen. Je nach Wahl der Wellenform ändern sich die Werte dieses Registers im Verlauf des Tonsignals.
  Wurde die Dreieckwelle gewählt, so nimmt der Inhalt des Registers gleichmäßig von 0 bis 255 zu und gleichmäßig wieder ab.
  Bei der Wahl der Sägezahnwelle nimmt der Wert ebenfalls gleichmäßig zu, fällt jedoch nach dem Erreichen von 255 sofort auf 0 zurück und beginnt alsdann wieder erneut zu steigen.
  Bei der Rechteckwelle pendelt der Wert zwischen 0 und 255. Die Länge der beiden Phasen hängt von der Pulsbreite [5] ab.
  Wurde Rauschen als Wellenform gewählt, dann nimmt das Register zufällige Werte an. Es kann somit verwendet werden, um Zufallszahlen zu erzeugen [6].

Alle diese Funktionen des Tongenerators 3 können auch dann verwendet werden, wenn dieser mit Hilfe von Bit 7 des Registers 24 stumm geschaltet wurde.

2.9 Der Analog-/Digitalwandler

Der SID 6581 enthält zwei Analog-/Digitalwandler (A/D-Wandler). Aus technischen Gründen eignen sich für eine Anwendung der Wandler nur veränderlich Widerstände in irgendeiner Form, z.B. Photowiderstände, Heißleiter, Kaltleiter oder ähnliches. Sollen Spannungen gemessen werden, so müssen diese zuvor in eine geeignete Form umgewandelt werden.
Die Meßanordnung sieht so aus, daß an das eine Ende des Meßwiderstandes +5V angelegt werden, die am Controllerport des C64 verfügbar sind [7]. Das andere Ende des Meßwiderstands wird mit dem Analogeingang des SID (ebenfalls am Controllerport verfügbar, POTX und POTY) verbunden. Der aus den Registern 25 und 26 ausgelesene Wert ist ein Maß für den Widerstand.

Um die ganze Skala von 8 Bits auszunutzen, muß sich der Widerstand im Bereich von 200 Ohm [8] bis 200 KOhm bewegen.

Programme, die digitalisierte Sounds enthalten, sind selten zu finden. Das liegt vor allem daran, daß diese Sounds eine große Menge an Speicherplatz beanspruchen, von dem der C64 nichts zu verschenken hat. Geschickt eingebaute kurze Loops wie z.B. gesamplete Schlagzeugteile oder kurze Ansagen erzielen aber immer den größtmöglichen Effekt.

[5] Die Pulsbreite steht in den jeweiligen Registern 2 und 3.

[6] Dieses ist meines Wissens nach die einzige Möglichkeit, echte Zufallszahlen mit dem C64 in Assembler zu erhalten.

[7] Leider konnte ich dieses Feature noch nie selbst ausprobieren, weil ich damals einen C64 vom ALDI-Markt bekam. Diese Commodore64 wurden leider ohne die benötigte +5V Spannung am Controller-Port ausgeliefert.

[8] Der Mindestwert von 200 Ohm darf nicht unterschritten werden!

2.10 Spezielle Effekte

- Stimme 1, Register 4 ($04, 04)
  - Bit 0, Hüllkurve: Ist das Bit gesetzt, so wird die Attack/Decay/Sustain-Spanne gestartet. Wenn das Bit gelöscht ist, so wird die Release-Phase eingeleitet.
  - Bit 1, Sync: Wenn Bit 1 gesetzt ist, werden die Frequenzen des Oszillators 1 mit der des Oszillators 3 synchronisiert, (der sogenannte „Hard Sync“-Effekt). Damit eine Synchronisation stattfinden kann, darf die Frequenz des Oszillators 3 nicht 0 sein. Keine anderen Parameter außer der Frequenz der 3. Stimme werden beim Synchronisieren berücksichtigt.
  - Bit 2, Ring-Mod: Ist das Ring-Mod-Bit gesetzt ist, so wird eine Dreieckschwingung am Oszillator 1 durch eine Ringmodulation der Oszillatoren 1 und 3 ersetzt. Eine Veränderung der Frequenz in diesem Modus bewirkt eine Vielzahl von nichtharmonischen Obertonstrukturen, die am besten für Glocken- oder Gongsounds und für andere spezielle Effekte geeignet sind. Damit die Ringmodulation hörbar ist, muß zu einen am Oszillator 1 eine Dreieckschwingung gewählt sein und zum anderen am Oszillator 3 eine Frequenz gewählt sein, die größer 0 ist. Wie beim Sync-Bit haben auch beim Ring-Mod-Bit keine anderen Parameter der 3. Stimme als ihre Frequenz eine Bedeutung für die Ringmodulation.
  - Bit 3, Test: Das Test-Bit führt einen Reset am Oszillator 1 aus und sperrt diesen solange, bis dieses Bit wieder auf 0 gesetzt wird. Normalerweise wird diese Funktion zu Testzwecken benutzt, man kann sie aber auch zur Synchronisation des Oszillators 1 mit anderen externen Signalen benutzen, um Effekte in Echtzeit zu erzielen.
- Stimme 2, Register 4 ($0b, 11)
  Die Register $07-$0d kontrollieren die 2. Stimme und sind mit diesen Ausnahmen funktionell identisch mit den Registern $00-$06:
  - Ist das Sync-Bit gesetzt, so wird der Oszillator 2 mit Oszillator 1 synchronisiert.
  - Ist das Ring-Mod-Bit gesetzt, so wird eine Dreieckschwingung an Oszillator 2 durch eine Ringmodulation der Oszillatoren 1 und 2 ersetzt.
- Stimme 3, Register 4 ($12, 18)
  Die Register $0e-$14 kontrollieren die 3. Stimme und sind mit diesen Ausnahmen funktionell identisch mit den Registern $00-$06:
  - Ist das Sync-Bit gesetzt, so wird der Oszillator 3 mit Oszillator 2 synchronisiert.
  - Ist das Ring-Mod-Bit gesetzt, so wird eine Dreieckschwingung an Oszillator 3 durch eine Ringmodulation der Oszillatoren 2 und 3 ersetzt.

3 Software

3.1 „FutureComposer“

Der „FutureComposer“ war das erste Programm, mit dem ich am Commodore 64 Musikstücke erstellt habe, die einfach in Assembler abzuspielen sind [9].
Die Hilfe (Abbildung 9), die man zu Anfang des Programms zu lesen hatte, war etliche Seiten lang und beschrieb recht detailiert die Funktionen des Programms. Die Handhabung ist sehr gewöhnungsbedürftig und zeitaufwändig, was wohl der Grund dafür war, warum ich damals nur wenige Musikstücke wirklich fertigstellte. Die grundlegenden Merkmale dieses Programms, von denen es eine Menge weitere ähnliche gibt, können wie folgt zusammengefaßt werden:

- Das Programm arbeitet ohne hochauflösende grafische Oberfläche.
- Es ist nicht möglich, Notenschrift mit diesem Programm zu erstellen oder zu drucken.
- Der „FutureComposer“ arbeitet nicht nach dem Tracker-Prinzip, d.h., die Dauer der Noten entspricht nicht die visuell dargestellten Form [10].

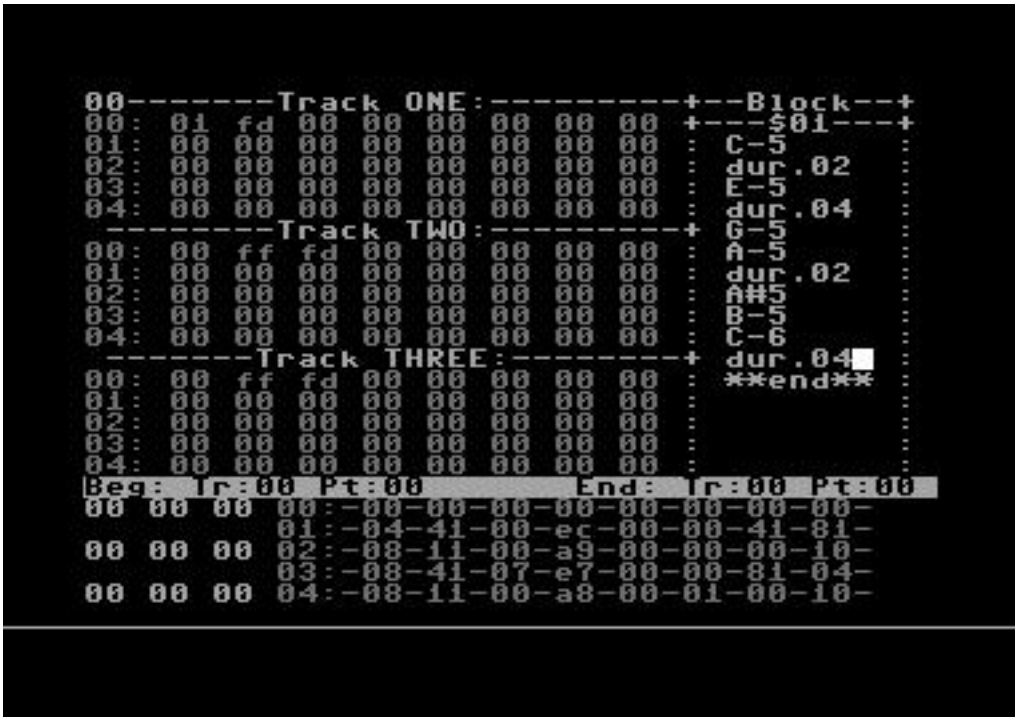

Abbildung 8: „FutureComposer“, Editor

Anhand dieses Screenshots (Abbildung 8) sei kurz der Aufbau und die Arbeitsweise dieses Programms erklärt:

- TrackEditor (oben links): In den drei Stimmen (Track ONE, Track TWO, Track THREE) ist der Aufruf der einzelnen „Blocks“ und somit der Aufbau des Musikstückes festgelegt. Steht zum Beispiel $01 im TrackEditor, so wird Block $01 aufgerufen und dessen Noten abgespielt. Man kann Blocks mehrfach hintereinander aufrufen und somit Speicherplatz sparen. Werte über $80 haben Sonderfunktionen, wie zum Beispiel *Transpose* (ab $81) oder *Repeat* ($fd) und *End* ($ff).
- BlockEditor (oben rechts): In einem Block stehen die eigentlichen Musiknoten. Hier wird die Tonhöhe (z.B. C-5 ist ein C der 5. Oktave), das Instrument und die Tonlänge („duration“, z.B. dur.04 hält den Ton 4 Schläge, je nach Metrum) angegeben.
- SoundEditor (unten): Hier werden die Instrumente definiert. Alle Parameter wie z.B. die Wellenform, die ADSR-Hüllkurve oder die Filter müssen für jedes Instrument einzeln programmiert werden (vgl. Abbildung 9).

[9] Ich meine das im Gegensatz zu anderen Musikprogrammen, die sicherlich auch ihre Vorzüge, wie z.B. Notenschrift haben, sich aber nicht für die Einbindung in Programme eignen.
[10] Mehr zum Tracker-Prinzip kann im Kapitel zu „OdinTracker“ nachgelesen werden.

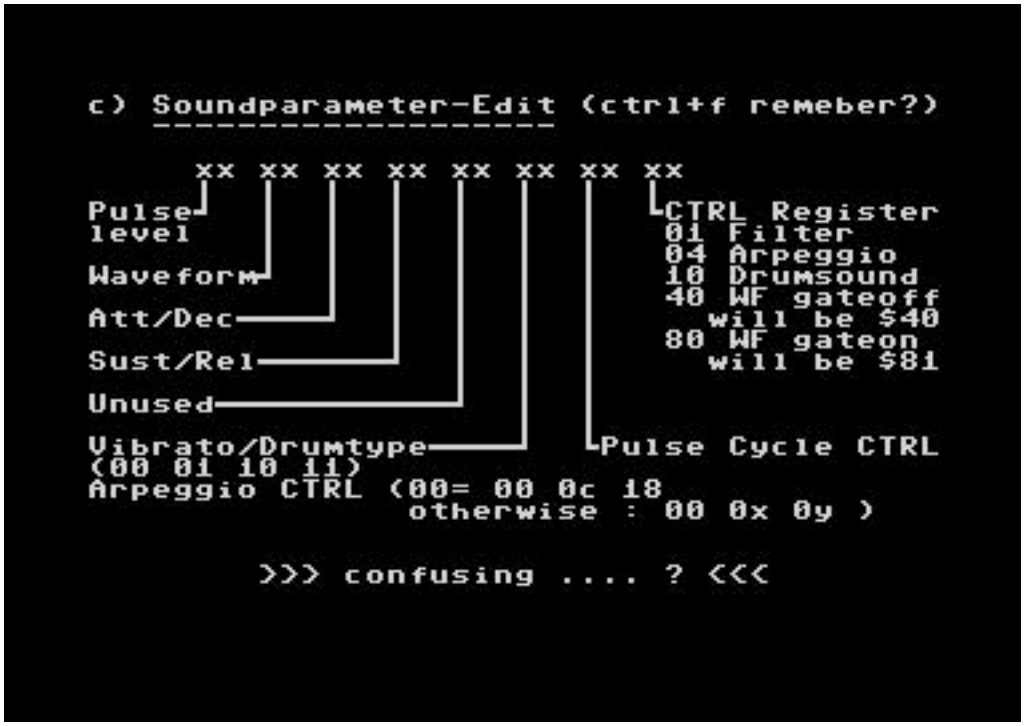

Abbildung 9: „FutureComposer“, Dokumentation

Immerhin lassen sich mit dem „FutureComposer“ Musikstücke erstellen, die äußerst wenig Speicher für sich beanspruchen (ca. 14 Blocks mit Abspielroutine) und die zudem auch eine sehr schnelle Abspielroutine haben (ca. 28 Rasterzeilen). Da dieses die frühste Form der Soundprogrammierung auf dem C64 war, darf angenommen werden, daß die meisten der frühen Spielmusiken wohl mit einem ähnlichen Editor programmiert wurden. Vor solchem Hintergrund gewinnen diese Musikstücke für mich einen besonderen Reiz, wenn man bedenkt, wie aus heutiger Sicht unglaublich aufwendig die Programmierung gewesen sein muß.

3.2 Tracker

3.2.1 „OdinTracker“

„OdinTracker“ ist ein neueres Programm (2000 veröffentlicht) zur Bearbeitung von Musikstücken, den Sid-Files, auf dem C64. Wie der Name schon verrät, ist dieses Programm ein „Tracker“. Die Tracker-Idee war schon vorher z.B. vom „ProTracker“ auf dem Amiga oder dem „FastTracker“ auf dem PC bekannt. Hierbei handelt es sich um eine Darstellungsform der Musiknoten, die zwar keine traditionelle Notenschrift imitiert, die aber immerhin schon einmal übersichtlicher als zum Beispiel die vom „FutureComposer“ ist. Anders als bei Programmen wie dem „FutureComposer“, bei denen man jede Notendauer explizit angeben muß (z.B. „dur.04“), bewegt man sich bei einem Tracker in einem zeitlichen Raster stehts nach unten, wobei gleichzeitig ertönende Noten in den verscheidenen Stimmen nebeneinander aufgereiht sind. Die Länge des Rasters kann für alle Stimmen gleichzeitig eingestellt werden. Betrachtet man einen Raster der vorgegebenen Länge für *eine* Stimme, so spricht man von einem Track. Die Tracks sind durchnummeriert und entsprechen somit in etwa der „Blocks“ im „FutureComposer“. Das einzige, was von den älteren Programmen wie dem „FutureComposer“ übernommen wurde, ist die schreibweise der Notenhöhen (C#5 ist ein Cis in der 5. Oktave).

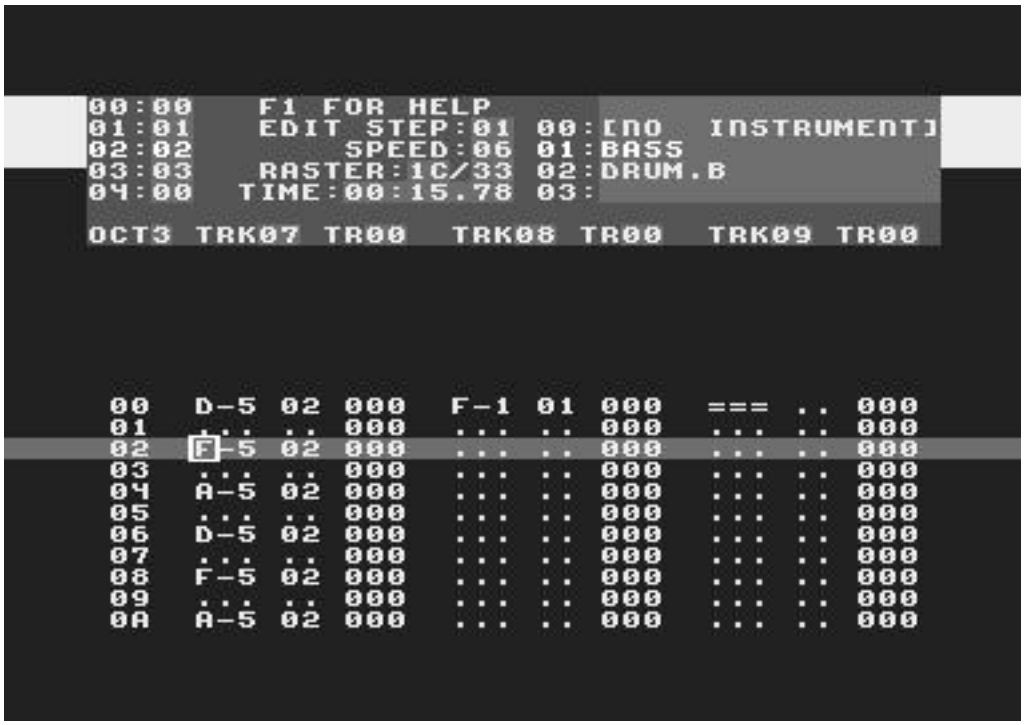

Abbildung 10: OdinTracker TrackEditor

Übersicht des TrackEditors (Abbildung 10):

- PatternEditor (oben links): Ähnlich wie der TrackEditor beim „FutureComposer“ gibt auch der PatternEditor hier die Abfolge des Musikstücks an. Ganz links befindet sich eine fortlaufende Nummerierung, lediglich die Position, an welchem Pattern sich das Stück gerade befindet. Rechts daneben stehen die Patterns, die jeweils die Anordnung dreier Tracks enthalten. In der Statusleiste können die aktuellen Tracks abgelesen werden: In diesem Bild gehören die Tracks 07 [Stimme1], 08 [Stimme2] und 09 [Stimme3] zu einem Pattern, wahrscheinlich Pattern 00. Auf diese Weise kann wieder einmal viel Speicherplatz gespart werden, wie ich kurz erläutern möchte: Hat man z.B. einen leeren Tack 00, eine Baßlinie auf Track 01, eine Schlagzeuglinie auf Tack 02 und eine Melodie auf Track 03, so könnte man die Instrumente nacheinander einsetzen lassen. Die Patterns würden dann beispielsweise so aussehen:

  - Pattern 00: 01 00 00
  - Pattern 01: 01 02 00
  - Pattern 02: 01 02 03

  Hätte man jetzt noch eine weitere Melodie auf Track 04, die der ersten Melodie zuammen mit dem Baß und dem Schlagzeug folgen sollte, dann müßte das Pattern so aussehen:

  - Pattern 02: 01 02 03
  - Pattern 03: 01 02 04

- Instrumentenfenster (oben rechts): Das Instrumentenfenster dient lediglich der Übersichtlichkeit. Die vorher erstellten und somit verfügbaren Instrumente können auf- und abgescrollt und somit abgelesen werden.
- Statusleiste (mitte): In dieser Zeile stehen aktuelle globale Werte des Musikstücks. Von links nach rechts gelesen lassen sich die Basis-Oktave („OCT“) für das Editieren der Notenhöhe sowie die aktuellen Tracks („TRK“) und Transpositionen („TR“) der einzelnen Stimmen ablesen.
- TrackEditor (unten): Wie schon angesprochen stehen hier die eigentlichen Musikwerte des Stückes. Liest man von links nach rechts, so stehen dort die aktuelle Patternposition und daneben für alle drei Stimmen die Tonhöhe nebst Instrument und Effektparametern.

Abbildung 11 zeigt den SoundEditor des (*OdinTracker*), in dem man die Parameter getrennt programmieren kann.

Abbildung 11: OdinTracker SoundEditor

Der obere Teil des SoundEditors bleibt wie im TrackEditor bestehen. Der untere Teil sei hier kurz beschrieben:

- Instrumentenname (mitte): Hier kann der Name für das Instrument eingetragen werden.
- Funktionsbezeichnung (links): Die einzelnen Klangparameter können hier programmiert werden. Die ADSR-Hüllkurve, das Vibrato und die Pulsbreite für Rechteckschwingungen können direkt eingetragen werden, bei der Wellenform, dem Arpeggio und dem Filter werden Wertetabellen („tables“) angesprungen.
- Wertetabellen (rechts): Hier können die Werte für die Wellenform, das Arpeggio und den Filter direkt eingetragen werden. Durch dieses Verfahren können neue, interessante Klänge erzeugt werden, wenn man z.B. mehrere verschiedene Wellenformen hintereinander setzt. Durch die rasche Wiederholung in einer Schleife (Loop) verschmelzen Klänge dann fast ineinander. Gleiches gilt für Arpeggio und Filtereigenschaften.

3.2.2 „Sid Duzz It“

Das Programm „Sid Duzz It“ (kurz: „SDI“) von der Gruppe Shape stellt für mich momentan das beste Tracker-Programm für den C64 dar. Zwar ist die Bedienung und die Übersichtlichkeit meiner Meinung nach nicht so gut gelöst wie bei „OdinTracker“, dafür haben Musikstücke, die mit „SDI“ gemacht wurden, zwei für die Arbeit mit dem C64 entscheidende Vorteile:

1. Die Musikstücke belegen weitaus weniger Platz im Speicher (unter 20 Blocks).
2. Die Abspielroutine von „SDI“ ist mit einem Verbrauch von ca. 30 Rasterzeilen ein wenig schneller als die vom „Odin-Tracker“.

Ausgehend von einem Screenshot (Abbildung 12) möchte ich kurz auf die Handhabung und Arbeitsweise dieses Programms eingehen.

Abbildung 12: Sid Duzz It

Übersicht:

- Ganz oben auf dem Bildschirm ist eine Status-Zeile zu sehen, in der grundlegende Einstellungen immer präsent sind. Hier wird angezeigt, welche Stimmen stummgeschaltet sind und in welchem Track man sich gerade bewegt.
- Darunter befindet sich der TrackEditor in heller Schrift. Hier werden die Abfolge der Tracks, die Instrumente, die Notenwerte und die Effektzuweisungen bearbeitet.
- Wichtig ist auch der darunter befindliche SoundEditor. Hier können die Instrumente benannt und ihnen die einzelnen Parameter zugewiesen werden. Rechts sind wieder die „Tables“ zu sehen.
- Die unteren fünf Zeilen zeigen den aktuellen Status der wichtigsten SID-Register dar. Diese Übersicht ermöglicht ein schnelles Aufspüren von Fehlerquellen.

Auffällig ist bei diesem Programm, daß nicht wie erwartet drei, sondern vier Stimmen im Trackeditor zu sehen sind. Vorerst hatte ich angenommen, daß diese vierte Stimme für digitalisierte Instrumente oder ähnliches gedacht wäre, es ist aber schlichtweg eine Spur zum Ausprobieren und zum Kopieren, die dem Komponisten die Arbeit erleichtern soll.

3.3 „MidiSlave“

3.3.1 Über den Einsatz von MIDI am C64

Gleich vorweg gesagt: Der Einsatz von Midi-Hardware spielt in der Commodore64 Welt, anders als bei anderen uns heute geläufigen Systemen, eine eher untergeordnete Rolle. Ich selbst kann mich nicht daran erinnern, ein midifähiges Programm auf dem C64 gesehen zu haben, bevor ich nicht ausgiebig danach im Internet gesucht hätte.
Von anderen C64-Midi-Benutzern habe ich mich beraten und mir sagen lassen, daß das Programm „MidiSlave“ von der Gruppe Triad zur Zeit das beste wäre, was den Einsatz von Midi am C64 ermöglichen könne. Leider konnte ich dieses Programm nicht selber testen, da mir das erforderliche Midi-Interface fehlt. Immerhin habe ich im Internet Angebote in Online-Auktionen entdecken können, die ein solches Gerät zum Verkauf anbieten. Auch eine Bauanleitung für versierte Eletronikbastler habe ich gefunden sowie einen Midi-Kurs für den C64 in einer Ausgabe des „Go64“-Magazins [11], dem ich einige interessante neue Informationen entnehmen konnte.

Midi-Interfaces wurden einst zu einem hohen Preis verkauft, manche kamen gar komplett mit Sequencer-Software als Modul. Wie bei so vielen Dingen in der Elektronik ist auch hier nicht verständlich, warum ein Gerät, das einen Bauteilewert von ca. 5 Euro hat, für ein vielfaches dieses Wertes verkauft wird, denn eigentlich ist in so einem Interface nicht viel Elektronik enthalten.

Jedes MIDI-Interface besteht im wesentlichen aus einem Sende/Empfangsbaustein (UART [12]) sowie etwas Elektronik drumherum. Diese fungiert als Schutzelektronik (sogenannte Optokoppler) und soll dafür sorgen, daß im ungünstigsten Falle lediglich diese billige Elektronik beschädigt wird und nicht der UART oder gar dahinter liegende Logik, denn bei der Entwicklung des MIDI-Interfaces war Robustheit gefragt.

Alle MIDI-Interfaces für den C64 verwenden den UART6850 als Schnittstellenbaustein. An diesem hängt eine externe 2MHz-Takterzeugung, welche als Schrittmacher für die serielle Schnittstelle des UART fungiert. Der serielle Ausgang wird über Treiberbausteine gepuffert und direkt an die MIDI-Out-Buchse geschickt. Das MIDI-In-Signal wird über einen Optokoppler galvanisch von der Außenwelt getrennt und gegebenenfalls über eine MIDI-Thru-Buchse durchgeschleift [13]. Kommerziell vertriebene MIDI-Interfaces gab es in Europa vornehmlich von den Firmen „C-LAB“, „Steinberg“ und „Jellinghaus“, aus den USA kamen einige Produkte von „Dr. T’s2“ und „Opcode“ zu uns. Selbstbauprojekte wurden seinerzeit in der Zeitschrift „Keyboards“ sowie in diversen „64er“-Sonderheften veröffentlicht.

Auf die Programmierung der UART möchte ich nicht detailiert eingehen, nur soviel, daß ich generell sage, daß sie sehr unkompliziert und einfach ist. Ich habe mir dazu ein passendes Assemblerbeispiel durchgesehen und konnte feststellen, daß die zuständigen Register des I/O1-Ports ($D404 - $D407) bequem im Normalbetrieb wie im Interrupt ausgelesen und beschrieben werden können.

[11] „Go64“ ist das letzte in Deutschland nichtelektronisch erscheinende C64-Magazin.
[12] Ausgeschrieben bedeutet UART: „Universal Asynchronous Receiver/Transmitter“.
[13] Tatsächlich sind fast alle MIDI-Interfaces fuer den C64 identisch, meist unterscheiden sie sich nur in den verwendeten Optokopplern.

3.3.2 Über das Programm

Abbildung 13: MidiSlave Hauptmenü

Das Programm „MidiSlave“ wurde von der Gruppe „Triad“ in den letzten Jaren entwickelt. Mit diesem Programm soll es also möglich sein, durch ein Datel-kompatibles Midi-Interface mit dem C64 zu kommunizieren. Da ich wie gesagt nie selbst die erforderliche Hardware erwerben konnte, kann ich nicht viel zu diesem Programm sagen. Abbildung 13 zeigt das Hauptmenü, die Midi-Einstellungen werden wie in Abbildung 14 vorgenommen. Ein kleiner Editor läßt Klangveränderungen zu (Abbildung 15).

Abbildung 14: MidiSlave Midi-Einstellungen

Abbildung 15: MidiSlave Soundparameter

3.4 Notation und Jukebox

Um meine Zusammenstellung von Musikprogrammen abzurunden, möchte ich in diesem Teil noch zwei weitere Programme vorstellen, die nicht direkt mit der Produktion der für den C64 typischen Klänge zu tun haben.

3.4.1 „MusicShop“

Abbildung 16: MusicShop - Titelbild

Der „MusicShop“ von Broderbund Software ist das ausgereifteste Notatensatzprogramm auf dem Commodore 64. Bis zu drei Stimmen lassen sich in Notenschrift mit dem Joystick editieren, anhören und schließlich in akzeptabler Qualität ausdrucken. Die gesammte Oberfläche ist grafisch gestaltet mit einem Pulldown-Menü am oberen Bildrand. Alle gebräuchlichen Notenzeichen sind sinngemäß einsetzbar, wobei die Notenhalsung leider nicht gebunden werden kann. Der Klang ist nur sehr begrenzt einstellbar und nur zum abhören eventueller Fehler gedacht. Sieht man aber von solchen Details einmal ab, so ist dieses Programm meiner Meinung nach durchaus auch heute noch brauchbar.

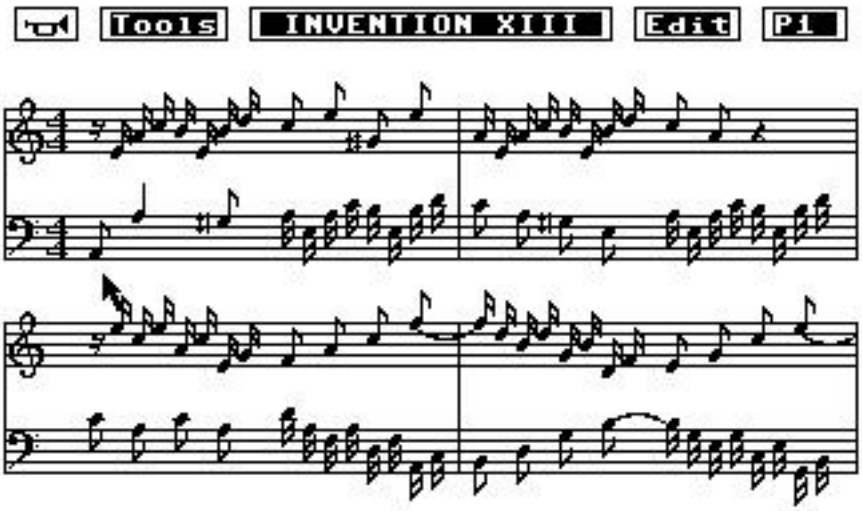

Abbildung 17: MusicShop - Noteneditor

3.4.2 “SidPlayer“

Natürlich gar es auch damals schon Programme für den normalen C64-Benutzer, der einfach nur Musik hören wollte. Dieses Programm heißt schlicht „SidPlayer“ (Abbildung 18) und fungiert als eine Art Jukebox für den C64. Ich selbst habe dieses Programm immer gerne laufen lassen und dabei die drei Stimmen auf der Klaviatur verfolgt. Prinzipiell hätte dieses Programm zu einem vollwertigen Kompositionsprogramm zählen können, wenn auch ein Editor hierfür publik gemacht worden wäre. Der eigentliche Noteneditor allerdings blieb mir immer vorenthalten, so daß ich zu der Annahme kommen muß, daß diese Stücke von einer Firma produziert und dann als „SidPlayer“-Stücke verkauft wurden.

Abbildung 18: SidPlayer

4 Links und Literatur

- Link-Auswahl zum Thema Commodore 64 SID:
  - „Odintracker“ von Zoltán Konyha
    http://www.inf.bme.hu/~zed/tracker/index.html
  - „Sid Duzz It“ (SDI) von Geir Tjelta (SHAPE)
    http://home.eunet.no/~ggallefo/html/sdi.html
  - „High Voltage Sid Collection“, die größte Sammlung von Stücken im original C64 Format
    http://www.hvsc.c64.org
  - SidPlayer für PC und andere Systeme
    http://www.SidPlayer.C64.org
  - SidPlug, der SidPlayer-Plugin für Netscape
    http://www.geocities.com/SiliconValley/Lakes/5147/sidplay/sidplug.html
  - C64 Remixes im MP3-Format
    http://remix.kwed.org
  - Eine SID-basierte PC Karte
    http://www.hardsid.com
  - Ein kommerzieller Synthesizer mit eingebautem SID-Chip
    http://www.sidstation.com
  - Ein Internet-Radio, daß C64 und Amiga Songs spielt
    http://www.kohina.com
  - Die allgemeine Commodore 64 Directory
    http://www.c64.org
  - Die Commodore 64 Literaturliste
    http://www.doc64.de
- Literaturauswahl zum Thema Commodore 64 SID:
  - Dachsel: Das Musikbuch zum Commodore 64. Data Becker, 1984
  - Lawrence, England: Sachbuch Band 13: Programmieren in Maschinensprache auf dem Commodore 64: Grafik und Musik. Commodore, 1984
  - Baloui, Brückmann, Englisch, Felt, Gelfand, Gertis, Krsnik: Commodore 64 Intern. Data Becker, 1990
  - (Unbekannt): Commodore 64 Handbuch. Commodore (Frankfurt, 1988)
  - Lorenz: Grafik und Sound mit dem C-64. Hofacker, 1984
  - Spitzner: Das C64/128 Musik Kompendium. Markt&Technik, 1988