

Внешняя звуковая карта на базе YM2612

Автор: Romanich
e-mail: RomanichApparate@mail.ru

1. О чём эта статья

В данной статье рассказывается о создании внешней звуковой карты (для краткости далее в тексте ВЗК) для IBM совместимого персонального компьютера (IBM PC). Звуковая карта собрана на базе музыкального процессора YM2612 или его аналогов (TA-07, PCS8593). Общение компьютера с ВЗК происходит по LPT-порту, питание берётся от USB-порта. Имеется предварительный усилитель звука, выход которого может подсоединяться к наушникам, активным колонкам или усилителю мощности.

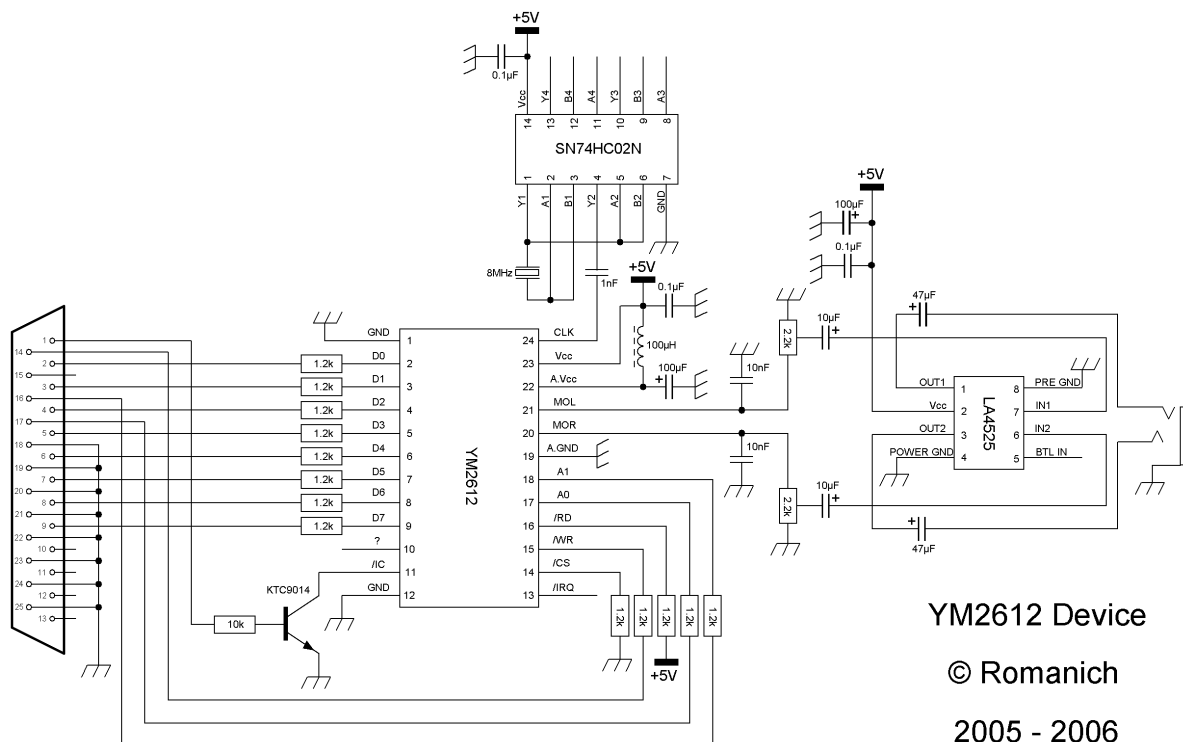
Также в статье рассматривается ряд вопросов, связанных с программированием микросхемы YM2612: общение с регистрами звукового чипа, формат музыкальных файлов *.GYM, *.VGM, а также рассмотрена возможность воспроизведения 8-битных оцифровок с частотой дискретизации 0...44100 Гц.

Приведены все необходимые сведения для создания, программирования и использования ВЗК.

Всё что рассказано в данной статье, сделано, проверено и работает под управлением операционных систем DOS v7.0 и Windows98.

2. Описание принципиальной схемы ВЗК

Схема ВЗК представлена ниже.



Основа схемы – звуковой чип (музыкальный процессор, микросхема,... - кому как больше нравится) YM2612 от японской фирмы Yamaha. Данная микросхема позволяет получить множество звуковых эффектов на основе FM-синтеза. Тип частотного синтеза –

OPN2. Это значит, что каждый инструмент имеет 4 оператора, которые соединяются по типу N. Учитывая то, что это – крайне редкая микросхема, доведу до сведения, что вместо неё можно поставить аналоги: TA-07 или PCS8593, выпаянные из неисправных плат игровой приставки SEGA (MegaDrive a.k.a. Genesis). То есть наша ВЗК будет способна проигрывать музыкальные композиции сеговских игр!

Звуковой чип не имеет встроенного тактового генератора, поэтому просто подсоединить кварц в ножке CLK нельзя! Поэтому на микросхеме SN74HC02N собран тактовый генератор на 8 МГц. Функционально – это генератор, состоящий из вентиля ИЛИ-НЕ, охваченного кварцем по обратной связи и буферный каскад на следующем вентиле ИЛИ-НЕ.

Звуковой чип ещё хорош тем, что он имеет встроенный ЦАП, поэтому с ножек MOL (левый канал) и MOR (правый канал) аналоговый сигнал поступает на вход двухканального (стереофонического) усилителя звука LA4525, который почти не имеет никакого обвеса (кроме разделительных и фильтрующих конденсаторов) и достаточно линейен.

Интерфейс как было сказано уже выше – порт LPT. Он хорош тем, что обеспечивает удобное управление внешними устройствами (периферией) компьютером. Причём все его выходные линии (12 штук) используются: 8 бит данных, /CS, /WR, A0, A1.

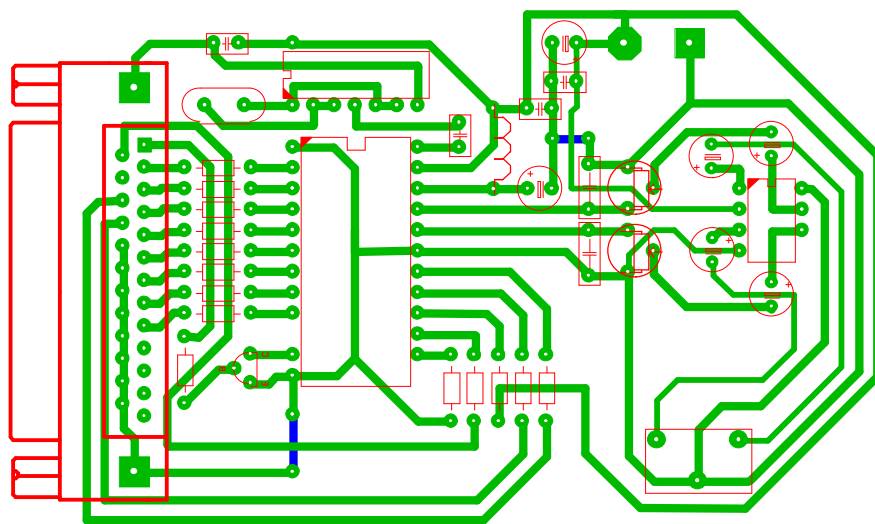
Ключ, собранный на транзисторе КТС9014, нужен для корректного сброса звукового чипа (вывод /CS внутрисхемно установлен в “1”, т.е. он подтянут).

Питание +5V берётся от USB-порта в компьютере. Но предпочтительнее (для избежания помех по питанию при активной работе блоков компьютера) использовать внешний блок питания на напряжение 5V (ток отдачи не менее 500mA).

Чтобы упростить составление комплектации, привожу перечень элементов:

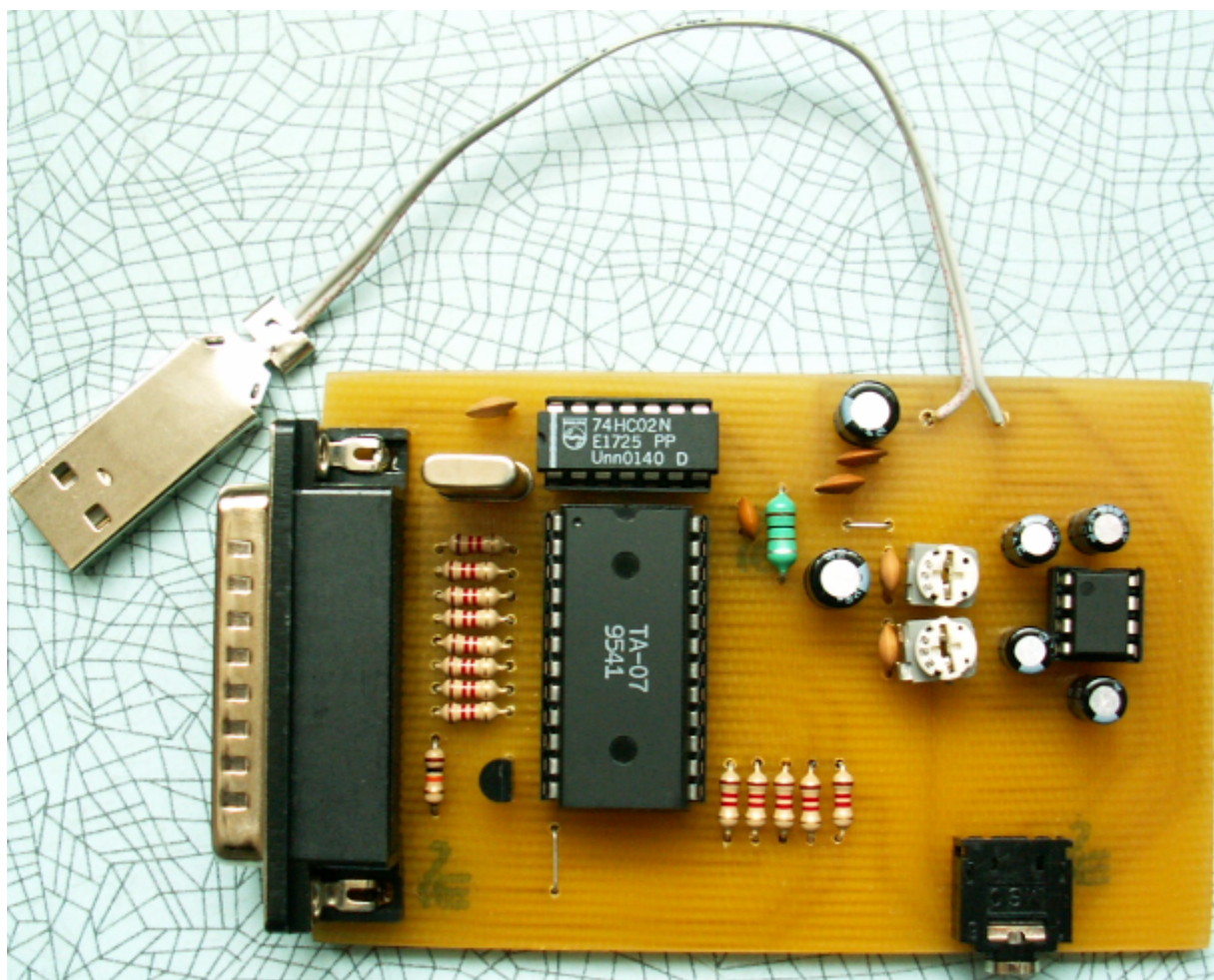
Радиоэлемент	Количество	Примечание
YM2612	1	Аналоги: PCS8593, TA-07
SN74HC02N	1	
LA4525	1	
КТС9014	1	
8MHz	1	
DRB-25MA	1	Разъём LPT порта со штырьками, паяется к плате
Audio	1	Разъём под штекер для наушников, колонок, усилителя
USB	1	Разъём для подачи напряжения питания 5V от USB
1.2 кОм	13	Подстроечные
10 кОм	1	
2.2 кОм	2	
0.1 мкФ	3	
10 нФ	2	
1 нФ	1	
100 мкФ	2	
47 мкФ	2	
10 мкФ	2	
100 мкГн	1	
DIP24	1	Панелька на YM2612, широкая
DIP14	1	Панелька на SN74HC02N
DIP8	1	Панелька на LA4525

Рисунок печатной платы приведён ниже.



Как видно из рисунка – плата односторонняя, имеет две перемычки, монтаж довольно неплотный, поэтому трудностей при изготовлении печатной платы не должно возникнуть.

Фото собранного девайса.



3. Программирование звукового чипа

Прежде чем рассказать о программировании YM2612 нужно подробно изучить все её ножки. Привожу краткое назначение каждой из ножек:

GND – цифровая земля (“минус” напряжения питания)

D0...D7 – шина данных (для установки адреса регистра и его значения)

? – вывод неизвестного назначения (оставить в воздухе)

/IC – вывод сброса микросхемы (обнуляются все регистры), внутрисхемно уже установлен в “1”

CLK – подача сигнала 8 МГц (уровень КМОП или ТТЛ, наличие постоянной составляющей крайне нежелательно, т.е. подавать через конденсатор)

Vcc – подача напряжения питания +5V для цифровой части звукового чипа

A.Vcc – напряжение питания +5V для аналоговой части звукового чипа

MOL – аналоговый выход левого канала (открытый эмиттер)

MOR – аналоговый выход правого канала (открытый эмиттер)

A.GND – аналоговая земля (“минус” питания)

A0, A1 – шина адреса (A0 – адрес регистра/данные, A1 – каналы 1...3/4...6)

/RD – строб чтения (у нас установлен хардварно в “1” – т.е. чтение запрещено)

/WR – строб записи (запись происходит когда сброшен в “0”)

/CS – выбор кристалла (у нас сброшен в “0” – т.е. микросхема всегда выбрана)

/IRQ – для реализации системы прерываний, у нас не используется

Фактически, в нашем случае для управления чипом нужны выводы D0...D7, /IC, /WR, A0, A1. Рекомендую обратиться к даташиту на YM2203 с целью чтоб по-подробнее ознакомиться с ним, так как он аналогичен нашему чипу. Даташита на YM2612 просто нет, так как это – собственность SEGA™ и вряд-ли по Интернету можно накопать что-нибудь про него, связанное с железом, кроме самой SEGA. Есть только описание регистров (что собственно совпадает с YM2203). Если хорошо постараться – можно найти принципиальные схемы SEGA MD/Genesis. Именно из них я взял схему включения YM2612.

А теперь вкратце о самой микросхеме YM2612 с программистской колокольни. Звуковой чип делится на два банка: Bank0 и Bank1. Bank0 отвечает за каналы 1, 2, 3, а Bank1 – за каналы 4, 5, 6. В данном случае – каналов 6, то есть одновременно может звучать 6 инструментов максимум! Каждый канал имеет 4 оператора, которые могут соединяться одним из восьми способов (определены в даташите), создавая при этом различные схемы (алгоритмы) FM-синтеза (для реализации разных классов музыкальных инструментов).

Если нужно работать с Bank0, то перед использованием регистров этой части (каналы 1...3), нужно, чтоб A1=0. Если необходимо задействовать Bank1 (каналы 4...6), то A1=1. То есть нужная часть звукового чипа определяется посылкой в A1 нуля или единицы – микросхема как бы внутренне состоит из двух.

Перед тем как общаться с регистрами YM2612, необходимо её сбросить. Делается это так:

A1=0 – выбираем Bank0

IC=0 - /IC=1

IC=1 - /IC=0 – формируем строб сброса для Bank0

IC=0 - /IC=1

A1=1 – выбираем Bank1

IC=0 - /IC=1

IC=1 - /IC=0 – формируем строб сброса для Bank1

IC=0 - /IC=1

После этого все регистры YM2612 в обеих её частях будут обнулены. Обратите внимание, что мы формируем строб RESET (IC) инверсно!!! Это неизбежно, так как к выводу /IC подключен транзисторный ключ, инвертирующий сигнал, поэтому на базу транзистора посылка должна быть такой (IC): 0, 1, 0, а не 1, 0, 1. С коллектора сигнал-же будет проинвертирован и на вывод /IC пойдёт: 1, 0, 1, т.е. как надо. Сброс делать необходимо, так как при включении микросхемы её регистры запрограммированы случайными значениями – возможна грязь и посторонние звуки.

А теперь об общении с регистрами чипа. Проводится в два этапа – выбор адреса регистра и запись в выбранный адрес значения. Символично это выглядит так:

[A]=D

То есть [A] – это регистр с конкретным адресом, D – это его значение. A0 отвечает за смысл данных – если A0=0, то будет записываться **адрес** регистра, если A0=1, то будут записываться **данные** в уже выбранный регистр. /WR – это строб записи, передёргивается в “0”, когда адрес/данные надо уже писать и обязательно устанавливается в “1”, когда идут манипуляции с A0. Строб чтения /RD должен быть запрещён, т.е. установлен в “1”, а микросхема выбрана - /CS=0. D – это байт шины данных (т.е. D0...D7). В общем алгоритм таков:

/WR=1 – отключаем запись

A0=0 – будем писать адрес регистра

D=Address – пишем адрес регистра (т.е. выбираем нужный регистр)

/WR=0 – включаем запись

Delay – ждём пока микросхема отработает (LPT-порт медленный сам по себе)

/WR=1 – отключаем снова запись (готовимся к формированию данных)

A0=1 – будем в выбранный регистр писать данные

D=Data – пишем данные в выбранный регистр

/WR=0 – активизируем запись

Delay – ждём...

/WR=1 – запрещаем запись

Упомяну ещё раз один важный момент. Если обращение идет к регистрам чипа каналов 1...3 (Bank0), то перед процедурой записи обязательно следует указывать, что работа идёт с Bank0 (A1=0). Если нужны каналы 4...6, то A1=1 (Bank1).

Примечание. Вместо сброса YM2612 можно просто тупо записать нули во все её регистры, но на мой взгляд с точки зрения программистской эстетики – это некрасиво!

После того как разобрались с записью в регистры чипа, можно смело его программировать ☺. Самое первое что приходит на ум – воспроизвести музыку из игр SEGA, о чём рассказывается далее...

4. GYM-файлы

Практически любой нормальный эмулятор приставки SEGA MD/Genesis на IBM PC имеет возможность создавать файлы музыки в двух форматах: WAV – это для Писюка и GYM – это просто дампы регистров звукового чипа YM2612 со служебными данными. Сразу хочу предостеречь, что виндовые эмуляторы создают очень кривые GYMы! Что самое интересное – все плагины (под WinAMP например) или специальные плееры воспроизводят их нормально! А на реальном звуковом чипе при воспроизведении таких GYMов будет смешная какофония! Что-ж – неудивительно, та же ошибка при записи, что и при чтении даёт верный результат! Поэтому я призываю использовать эмулятор SEGA под DOS (Genecyst), он очень качественно подходит к решению вопросов по созданию GYM-файлов.

А теперь о самом формате GYM. Он предельно прост, компактен и очень хорошо сжимается архиваторами! Речь будет идти о GYM-формате, не содержащем ничего кроме служебных данных и дампа регистров.

GYM формат состоит из четырёх инструкций: 0, 1, 2, 3.

0 – ждать 1/60 секунды

1 – запись в регистры Bank0. Далее адрес регистра и его значение

2 – запись в регистры Bank1. Далее адрес регистра и его значение

3 – запись в PSG одного следующего байта данных

PSG – это Program Sound Generator, его обсуждение выходит за рамки этой статьи, поэтому не обсуждается. В YM2612 его нет (в последних версиях приставок SEGA MD/Genesis он интегрирован в ВидеоПроцессор).

Пример (вымышленный):

0x00,0x00,0x00,0x01,0x23,0x55,0x02,0x34,0xFF,0x03,0xFA,0x00

Декодируется так:

ждать 1/60 сек.,

ждать 1/60 сек.,

ждать 1/60 сек.,

Bank0 [0x23]=0x55,

Bank1 [0x34]=0xFF,

[PSG]=0xFA,

ждать 1/60 сек.

Вот собственно небольшая, но исчерпывающая информация про GYM-файлы! Следует отметить, что в 99 % случаев в GYM-файлы к сожалению не входит DAC-

дорожка, то есть ударных в GYMe не будет ☹. Хотя встречаются такие музыкальные композиции, которые не используют DAC! Хотя при этом звучат нехилые басы!!!

5. DAC

Ещё один сюрприз... В YM2612 есть 8-битный встроенный DAC, который программно доступен! Причём он параллельной загрузки, что делает возможным воспроизводить оцифровки звука, посылая их в LPT порт со скоростями от 0 Гц до 44100Гц и выше (хотя это всё зависит от быстродействия IBM PC и оптимизации программы воспроизведения. И конечно же ограничивается предельными характеристиками микросхемы!).

Есть пара регистров: 0x2A и 0x2B. Для того, чтоб работать с DAC напрямую, нужно вначале в регистре разрешения DAC установить старший бит в "1", то есть:

[0x2B]=0x80

А далее в регистр данных DAC класть оцифровку:

for i:=1 to N do [0x2A]=Digital[i]

Вот и всё! И постараться делать цикл как можно быстрее! Все задержки реализуются программно (подбирается на слух, чтоб WAV звучал и не быстро и не медленно).

Примечание. Пока используется DAC, то будет недоступен 6-й канал (Bank1). При этом доступны только каналы 1...5.

6. VGM-файлы

К огромному счастью, есть ещё один формат файла, похожий на GYM, который содержит музыкальные композиции из игр под SEGA MD/Genesis! Это VGM-формат, который может содержать дампы регистров звукового чипа YM2612, а также дорожку для DAC! В отличие от GYM-формата, VGM-формат более качественно и экономично подходит к хранению музыкальных композиций. Например, DAC-дорожки (данные PCM) хранятся в единственном экземпляре, а в музыкальном потоке даны лишь ссылки на эти дорожки. А регистры звукового чипа могут при необходимости заполняться со скоростями выше, чем 50 / 60 Гц, что несомненно повышает качество и неотличимость звучания VGM-файла от звучания на реальной игровой приставке! Следует отметить, что VGM-файлов нет как таковых. Есть файлы с расширением VGZ, которые по сути являются сжатыми VGM-файлами! Для того, чтобы из VGZ получить VGM можно поступить так: VGZ-файл переименовывается в файл с расширением *.RAR, далее разжимается (например программой WinRAR 3.20) и переименовывается в *.VGM.

А теперь рассмотрим кратко спецификацию VGM (v1.50), то есть здесь объяснены только самые необходимые вещи, которые понадобятся для написания простейшего VGM-декодера.

В самом начале VGM-файл содержит 64-байтный заголовок, с различной информацией о нём. Нас интересуют следующие байты:

1) Двойное слово (4 байта) по адресу 0x1C – это смещение точки цикла. Оно равно нулю, если заикливание не предусмотрено или содержит значение, к которому надо прибавить 0x1C (чтобы получить абсолютное (от начала файла) смещение точки цикла). После окончания звукового потока (команда 0xb6) следует перейти к смещению точки

цикла (то есть – прыгнуть на это смещение, по идее близко от начала музыкального потока).

2) Двойное слово по адресу 0x34 – начало музыкального потока, к которому надо прибавить 0x34 чтоб вычислить абсолютный адрес потока в VGM-файле. Музыкальный поток содержит команды и данные.

Команды: 0x30...0x50 пропускаются, при этом указатель музыкального потока каждый раз увеличивается на 2.

Команды 0x51, 0x54...0x5F, 0xA0...0xBF пропускаются, при этом указатель потока увеличивается на 3.

Команды 0xC0...0xDF пропускаются, указатель потока увеличивается на 4.

Команды 0xE1...0xFF пропускаются, указатель увеличивается на 5.

Далее рассмотрим команды, касающиеся только YM2612, DAC, временные задержки и позиционирование.

0x52, [A], D – запись в регистр с адресом A данных D (Bank0). Указатель увеличивается на 3.

0x53, [A], D – запись в регистр с адресом A данных D (Bank1). Указатель увеличивается на 3.

0x61, nn, nn – задержка nnnn сэмплов (nnnn=0...65535). Один сэмпл равен 1/44100 секунд. Указатель музыкального потока увеличивается на 3.

0x62 – задержка на 1/60 секунды (735 сэмплов). Указатель потока растёт на 1.

0x63 – задержка на 1/50 секунды (882 сэмпла). Указатель потока растёт на 1.

0x66 – конец музыкального потока: это значит либо мы можем прекратить воспроизведение или перейти в точку зацикливания (см. выше). Или начать играть музыкальную композицию сначала... Указатель музыкального потока устанавливается в первоначальное положение (начало / зацикливание).

0x67, 0x66, tt, ss, ss, ss, ss – конкретно:

0x67 – маркер блока данных для DAC

0x66 – конец потока (**здесь эта под-команда игнорируется!**)

tt – тип данных (должно =0 – это значит YM2612 PCM data)

ssssssss – объём данных для DAC в байтах

Здесь логично запомнить смещение данных для DAC, то есть начальный указатель на PCM data в VGM-файле. Указатель музыкального потока увеличивается на ssssssss+7.

0x7* - задержка на *+1 сэмплов. Напомню, что 1 сэмпл = 1/44100 секунд. Указатель потока увеличивается на 1.

0x8* - конкретно:

A1=0 – выбираем Bank0

[0x2A]=Digital[Offset]

Задержка на * сэмплов

Увеличиваем Offset на 1 (указатель на PCM data)

Увеличиваем указатель потока на 1

Попросту говоря, данная команда пишет в DAC один байт PCM-данных (указатель на PCM вычисляется командами 0x67 и 0xE0). Далее задержка на * сэмплов и увеличение указателей музыкального потока и PCM data на 1.

0xE0, dd, dd, dd, dd – вычисление увеличения указателя на PCM data. Итоговое значение указателя PCM data в данный момент: Offset=ddddddd+Адрес начала данных для DAC в VGM-файле (адрес байта, следующего сразу после команды 0x67, 0x66, tt, ss, ss, ss, ss). Указатель музыкального потока увеличивается на 5.

Оцените высокое качество воспроизведения VGM-файлов по сравнению с GYM-файлами, а также наличие полноценного DAC ☺

Конечно, для 100%-ной схожести не хватает PSG, но этот недостаток более незаметный, чем отсутствие DAC-дорожки в GYM-файле.

Следует отметить, что декодер VGM-файла получается намного сложнее, чем декодер GYM-файла, но не напрасно!

7. Patches

С большим трудом мои поиски в Интернете увенчались успехом! На сайте <http://www.valsound.com/> мне удалось откопать несколько патчей для OPN2 чипов ☺ По сути – это похоже на General MIDI Instruments для Adlib, но только для OPN2. Автором патчей оказался Takeshi Abo, который выложил их в разделе [FM-Sound Library](#).

Разобравшись с форматом данных патчей, я приступил к их обработке (адаптировал их для YM2612). И вот, после нескольких дней, благодаря моему упорству написаны программы OPN2.exe и OPN2P.exe.

Без этих программ пришлось бы довольствоваться проигрыванием мелодий с приставки SEGA и/или слушать 8-битные оцифровки. А так появляется возможность подобрать 258 разнообразных музыкальных инструментов и звуковых эффектов из следующих групп:

Bass
Bell
Brass
Guitar and Distortion
Lead-Synth and Organ
Percussion
Piano
Sound Effect
Special
Strings and PAD
Wind
World

То есть, OPN2 Patches - программа, которая позволяет экспериментировать со встроенными в неё патчами - менять их параметры (Блок, Частоту) случайным образом (OPN2.exe) или вручную (OPN2P.exe).

Основная задача программы OPN2 Patches - это помочь найти нужные звуковые эффекты, подобрать их параметры (Блок, Частота) и далее - использовать патчи в своих разработках! В поставке программы есть файл патчей (бинарник) - OPN2. Ниже приведён формат патчей, лежащих в OPN2.

Основные положения:

- всего 258 патчей, нумеруемых с 0 до 257
- каждый патч описывается 38 байтами
- лишних данных в файле нет, поэтому его объём 9804 байта

Формат:

AL FB

AR1 DR1 SR1 RR1 SL1 OL1 KS1 ML1 DT1

AR2 DR2 SR2 RR2 SL2 OL2 KS2 ML2 DT2

AR3 DR3 SR3 RR3 SL3 OL3 KS3 ML3 DT3

AR4 DR4 SR4 RR4 SL4 OL4 KS4 ML4 DT4

AL - Algorithm

FB - FeedBack

AR* - Attack Rate

DR* - Decay Rate

SR* - Sustain Rate

RR* - Release Rate

SL* - Sustain Level

OL* - Operator Level

KS* - Key Scale

ML* - Multiple

DT* - Detune

Где * - номер оператора = 1...4

Описание данных программ завершает эту статью. Считаю, что в этой статье исчерпывающе рассказано о возможностях и сферах применения звукового чипа YM2612. Особо полезной эта статья будет для тех людей, которые самостоятельно проектируют свои вычислительные системы – микрокомпьютеры, игровые приставки и прочие маломощные мультимедиа- девайсы! Подключение цифровых устройств к LPT-порту весьма наглядно демонстрирует их возможности и алгоритмы управления!

Творческих успехов Вам в программировании внешней звуковой FM-карты! Не дадим исчезнуть навсегда **железному** FM синтезу!

© Romanich 20.05.06

