СБОРКА ИГРОВОГО ДВИЖКА ДЛЯ ОС WINDOWS 95

Вашкевич И.А., студент

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Порхун М.И. – маг. техн. наук, старший преподаватель

В работе описываются необходимые требования для сборки и тестирования игрового движка для компьютеров эпохи 90-ых и операционной системы Microsoft Windows 95 в частности, затрагивая ключевые отличия, с которыми столкнётся современный разработчик.

Вычислительные системы неумолимо устаревают морально и физически, однако год за годом находятся энтузиасты, что дают им новую жизнь. По этой причине перенос (портирование, от англ. *porting*) и написание нового программного обеспечения для официально неподдерживаемых платформ остаётся актуальной задачей. Так как устаревшие персональные компьютеры и их операционные системы не поддерживаются крупными разработчиками, они требуют особого подхода при разработке. Игровые движки же создаются специально для абстрагирования от платформы, что удобно при портировании. По этим причинам особенности разработки и сборки современного программного обеспечения для операционной системы *Windows 95* будет рассмотрено на примере игрового движка.

ОС *Windows 95* была выпущена в 1995-ом году *Microsoft* и привнесла многие функции и возможности, которыми мы пользуемся по сей день, включая поддержку 32-разрядных приложений и вытесняющую многозадачность, что делает её одной из старейших систем, для которых современные компиляторы, в теории, могут собирать приложения. На момент актуальности системы наиболее популярными были процессоры *Pentium*, что лишает нас инструкций *MMX* и *SSE*, представленных позже. Видеоадаптеры того времени умели отрисовывать лишь двухмерную графику, а первые трёхмерные игры полагались на процессор целиком. Однако уже годом позже вышел первый трёхмерный ускоритель, попавший к обычным пользователям: *3Dfx Vodooo* с графическим *API Glide.* Такой ускоритель ставился вместе с двухмерной видеокартой. Звуковые карты почти не изменились.

Таким образом, выбор ПО для портирования ограничен, так как необходимо низкое потребление памяти и процессора, а также *Glide*-совместимый отрисовщик. Однако в 1996-ом появился *OpenGL 1.1*, который распространился благодаря выходу аппаратно ускоренной версии игры *Quake* на нём, что привело к появлению прослоек совместимости *MiniGL* от *3Dfx* и *MesaFX* от проекта *Mesa*. Остаётся лишь найти движок с открытой лицензией, работающий с *OpenGL 1.1.*

Игровой движок *Godot* (читается “Годо”, от фр. *Godot*) изначально был проприетарным программным обеспечением Хуана Линиецкого и Ариэля Манзура и лицензировался другим разработчикам, находясь в таком состоянии с 2007 по 2014 год. После этого он вышел под открытой лицензией *MIT*, благодаря чему оброс большим сообществом. Движок работает с двухмерной и трёхмерной графикой и успел претерпеть множество изменений, однако только версия 1.0 унаследовала *OpenGL 1.3* отрисовщик, благодаря чему идеально подходит под поставленную задачу.

Далее необходимо найти комплект с компилятором, однако большинство готовых решений без модификаций не соберут программу совместимую с *Windows 95.* За основу взяты компилятор *GCC* версии 12 проекта *GNU* и библиотеки для разработки под *Windows* проекта *MinGW* версии 13.

В комплекте разработки для *Windows 95* необходимо предотвратить использования вызовов из более новых версий ядра. Одним из таких является *InterlockedCompareExchange.* Более ранние версии *GCC* включали вариант для старых версий *Windows* использующий *x86* инструкцию *xchg*. С помощью заплатки [1] секция исходного кода возвращается. Также требуется собирать компилятор для работы с *Microsoft Visual C Runtime 6* (*MSVCRT6*) [2], так как новые библиотеки, предоставляющие стандартные функции *C* и *C++*, тоже используют новые вызовы ядра. По этой причине отключается использование функций *at_quick_exit()* и *quick_exit()* в библиотеке *libstdc++-v3* из *GCC* заплаткой на файл конфигурации [1]. Большая же часть подобных вызовов производятся из библиотеки *pthreads* в *MinGW,* реализующей потоки, поэтому применяется реализация из *GCC* под названием *threads_win32.* Также выставляется целевая архитектура *Pentium* с помощью *$TARGET=i586-w64-mingw32*. Конфигурация приведена на рисунке 1.

```
../gcc-*/configure \
 --target="$TARGET" \
 --prefix="$PREFIX" \
 --disable-multilib \
 --enable-languages=c,c++ \
 --enable-threads=win32
make all-gcc -j$(nproc)
make install-gcc
```

Рисунок 1 – Конфигурация *GCC*

Стоит заметить, что данная реализация потоков поменялась в *GCC 13* и больше не подходит под поставленную задачу [3].

Далее собирается заголовочные файлы *mingw-w64-headers* и библиотеки *mingw-w64-crt* из *MinGW* с помощью полученного ранее минимального компилятора *GCC*, затем собирается *GCC* вместе с полученными библиотеками с помощью *make install*. Итоговый комплект разработки находится по пути *$PREFIX* и способен собирать рабочие исполняемые файлы для *Windows 95.*

Подходящим ответвлением *Godot 1.0* от сообщества является проект под названием *The Fusion Engine* [4]. Его разработчики уже выполнили большую часть работы по портированию движка, поэтому на данный момент его можно тестировать. Ниже приведены основные изменения относительно исходного проекта, которые касаются *Windows 95* и должны быть учтены при портировании:

– Заменены, убраны и исправлены некорректные вызовы ядра *Windows*. В старых версиях *Windows* на параметры вызовов ядра было наложено больше ограничений, в то время как на новых системах неправильные параметры могут не приводить к сбоям и оставаться незамеченными;

– Добавлена интеграция с библиотекой *Unicows* [5] от *Microsoft* для поддержки кодировок *Unicode*. Сюда относятся и улучшения работы с файловой системой для поддержки таких кодировок как *Windows-1251*, так как упомянутая библиотека не может добавить поддержку *Unicode* в *Windows* или *MSVCRT* . Например, для доступа к файлам используется *open()* вместе с *WideCharToMultiByte()*, так как *_wopen()* в *MSVCRT* работает только на *Windows 2000* и более новых;

– Добавлен отрисовщик для *OpenGL 1.1*, работающий *с прослойкой MesaFX* [6] (которая, однако, требует *Voodoo2* и новее). Изменения по сравнению с *OpenGL 1.3* сводятся к отключению *MIP*-текстурирования (из-за отсутствия встроенного метода генерации их уровней), работы с текстурными блоками (более одной текстуры за проход не используется) и экспериментальные эффекты, такие как карты нормалей;

– Исправления и изменения связанные с системой сборки, комплектом разработки и необязательными функциями.

Для тестирования применяется эмулятор *86Box* [7]. Заметим, что первые *Voodoo* не поддерживают оконный режим, а при работе наблюдаются аварийные выходы при определённых действиях и искажения текста при программной отрисовке. Демонстрация представлена на рисунке 2.

А) Б)

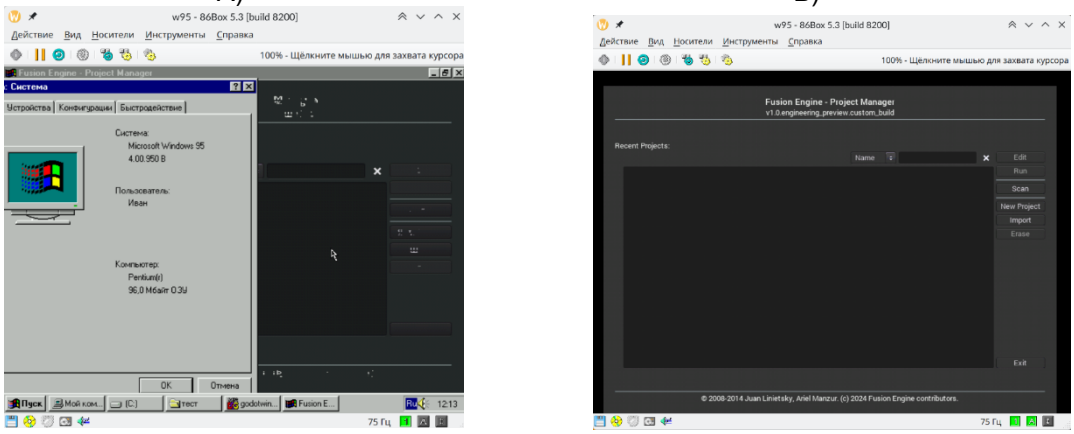

Рисунок 2 – Демонстрация: *а)* программная отрисовка; *б)* аппаратная отрисовка на *3Dfx Voodoo Banshee*

Обнаруженные проблемы к комплекту разработки не относятся и не связанны с версией самой системы, поэтому можно считать, что задача выполнена, а все особенности учтены.

Список использованных источников:

1. Страница проекта mingw-95 [Электронный ресурс] – https://github.com/ati9550/mingw-95

2. Страница загрузки Microsoft Visual C Runtime 6 [Электронный ресурс] – https://web.archive.org/web/20120610063726if_/http://download.microsoft.com/download/vc60pro/update/1/w9xnt4/en-us/vc6redistsetup_enu.exe

3. Статья об изменениях в GNU GCC 13 [Электронный ресурс] – https://gcc.gnu.org/gcc-13/changes.html

4 Страница проекта The Fusion Engine [Электронный ресурс] – https://github.com/TheFusionEngine/FusionEngine

5. Страница Unicows в библиотеке Microsoft для разработчиков [Электронный ресурс] – https://web.archive.org/web/20100116141417/http://msdn.microsoft.com/en-us/library/aa286515.aspx

6. Страница загрузки MesaFX [Электронный ресурс] – https://www.3dfxzone.it/dir/3dfx/mesafx/download/index.php

7. Страница эмулятора 86Box [Электронный ресурс] – https://86box.net/

8. Страница проекта MinGW [Электронный ресурс] – https://www.mingw-w64.org/

9. Страница проекта GNU GCC [Электронный ресурс] – https://gcc.gnu.org/