УДК 004.92

# ОРГАНИЗАЦИЯ ВИДЕОКОНВЕЙЕРА ДЛЯ ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ ПО МУЛЬТИМЕДИА ТЕХНОЛОГИЯМ В ОПЕРАЦИОННОЙ СИСТЕМЕ LINUX

*Абрамов Артур*

Кубанский Государственный Технологический Университет, Россия

**Аннотация**

*Описывается применение пайпинга для направления потоков обрабатываемого видео. Рассматривается возможность применения машины под управлением Linux для кодирования и декодирования видео в классе состоящем из машин под управлением Windows. Предлагается описание технической стороны реализации работы студентов на лабораторном занятии включающем обработку видео.*

*The usage of piping for video stream delivery is described. Paper concerns the question of using Linux controlled machine for encoding and decoding video for group of Windows controlled machines in the classroom. The principal technical backend for students attending laboratory classes involving video editing is offered.*

## Введение

Кодирование и декодирование видео в современные популярные форматы плохо стандартизировано, отличается обилием реализаций, неоднозначных терминов[1], и потому трудно для понимания и обучения. Более того, большинство существующих решений не предполагают возможности вычленения отдельных этапов своей работы, что делает их «чёрным ящиком» для преподавателей и студентов.

Цель данного доклада описать способ, позволяющий встроить программы обработки видео, написанные студентами, между звеньями декодирования исходного видео, и кодирования обработанного видео, с последующим его воспроизведением (рис. 1).

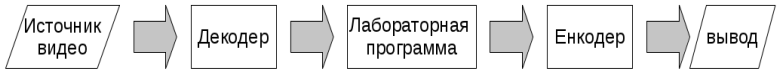


Рисунок 1 - Схема встройки лабораторных программ в видеоконвейер

## Среды, инструменты, идея решения

Реализация предполагает использование открытых библиотек FFmpeg через интерфейс одноимённого приложения в операционной системе Linux. Следует использовать любой дистрибутив поддерживающий Bash-скриты. Лабораторные работы допустимо программировать на любом языке, предоставляющем консольный ввод и вывод, а также возможность подключения сторонних библиотек. Автор рекомендует, и предоставляет консольные примеры на C++ в среде разработки Qt Creator. Воспроизведение видео будет осуществляться программой MPlayer.

Идея организации видеоконвейера заключается в следующем. В программу FFmpeg передаётся поток из любого источника (файл, камера, устройство видеозахвата). FFmpeg декодирует данный поток и передаёт несжатые кадры в лабораторную программу посредством стандартного ввода/вывода . Лабораторная программа производит обработку видео, после чего передаёт видео в обратно в FFmpeg, а затем в MPlayer или видео файл.

## Техническая реализация

Реализуется данная цепочка посредством Bash скрипта:

ffmpeg -i <имя входного файла> -f mjpeg -sameq pipe:|<имя лабораторной программы>|\
ffmpeg -f mjpeg -i pipe: -f avi -sameq pipe:|mplayer -

Использованы опции ffmpeg[2]:

- i – имя файла содержащего видео для обработки;
- f mjpeg – инструкция использовать формат (контейнер) mjpeg;
- f avi – инструкция использовать формат (контейнер) avi;
- sameq – кодировать/декодировать с максимальным качеством;
  pipe: – так ffmpeg обозначает stdin/stdout;

Обращение к стандартному вводу/выводу Линукса в лабораторной программе удобно производить через объекты библиотеки Qt QbyteArray, QtextStream и Qbuffer. Изображения будем хранить в объекте типа Qimage. Пример процедуры для загрузки одного кадра:

int input(QImage *img) QTextStream f(stdin,QIODevice::ReadWrite); img->LoadFromData(f.device,"jpeg"); return 0;

Как видно из листинга, открывается файл stdin, и производится чтение. В качестве формата изображения указан jpeg, что может смутить внимательного читателя, поскольку, как мы помним ffmpeg было указано декодировать в mjpeg. Как упоминалось в начале доклада стандартизация форматов видео оставляет желать лучшего, и это один из подтверждающих примеров. Версия 0.10.2 ffmpeg, также как Mplayer dev-SVN-r34818-4.5-openSUSE Linux 11.4 (x86_64)-Packman, интерпретирует mjpeg как конкатенацию (склейку) обычных jpeg изображений (каждый кадр со своим заголовком jpeg)[2][3]. Однако существуют и другие реализации.

Для вывода изображений в свою очередь используем процедуру подобную следующей:

int output(QImage *img) QByteArray frame(5000,0); QTextStream f(stdout,QIODevice::ReadWrite); QBuffer out(frame); f.flush(); img->save(out,"jpeg"); f«out.data(); return 0;

Данный код создаёт буфер в который записывается кадр. Этот кадр отправляется в stdout.

Используя данную несложную технику, студент может организовать цикл, последовательно обработав все кадры видео (рис. 2).

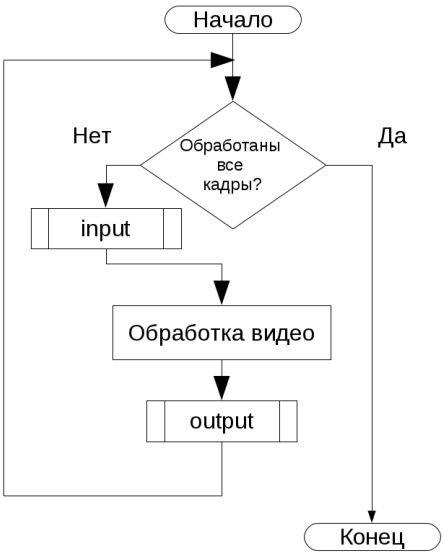


Рисунок 2 — Логика лабораторной программы

**Заключение**

В заключение следует сказать, что продемонстрированный пример с небольшими изменениями может легко стать мультиплатформенным (Windows, Mac OS, Linux), следует только внести синтаксические правки в Bash файл (.bat в Windows). Следует также отметить, что FFmpeg, как и MPlayer, является опенсорсным проектом[4][5], благодаря чему программисты не ограничены только интерфейсами предоставляемыми исполняемыми файлами данных программ, или API их библиотек, но и могут встраивать код в свои собственные проекты, становясь частью большого сообщества, и получая возможность познакомиться с устройством приложений мирового класса.

**Список использованных источников:**

1. Ватолин Д.С. Алгоритмы сжатия изображений: учеб.-метод пособие / Д.С. Ватолин – М.: изд. отдел факультета ВМиК МГУ им. Ломоносова, 1999 г.
2. FFmpeg Documentation // FFmpeg: URL: http://ffmpeg.org/ffmpeg.html
3. Encoding from multiple input image files (JPEG, PNG, TGA, etc.) // The MPlayer Project: URL: http://www.mplayerhq.hu/DOCS/HTML/en/menc-feat-enc-images.html
4. General Documentation // FFmpeg: URL: http://ffmpeg.org/general.html
5. MPlayer - Медиа Проигрыватель. Глава 1. Введение // The MPlayer Project: URL: http://www.mplayerhq.hu/DOCS/HTML/ru/intro.html