Разработка программного обеспечения для моделирования систем передачи данных с использованием распределённых вычислений

Тришин А. А.

Тришин Александр Андреевич / Trishin Alexander Andreevich – студент магистратуры, кафедра вычислительной и прикладной математики, факультет вычислительной техники, Рязанский государственный радиотехнический университет, г. Рязань

Аннотация: *выполнен анализ существующих программных средств, для моделирования систем передачи данных, описаны особенности создания модели, рассмотрены подходы для ускорения работы программы.*

Ключевые слова: *компьютерное моделирование, передача данных, помехоустойчивое кодирование, GPGPU.*

При передаче информации под действием на канал связи различного рода шумов и помех могут появиться искажения в сообщениях. Быстрый рост объемов обработки данных, развитие цифровых систем вещания и вычислительных сетей предъявляют высокие требования к минимизации ошибок в используемых цифровых данных. Поэтому одной из важнейших задач является обеспечение высокой достоверности передачи данных. Исправлением ошибок после передачи по каналу занимается помехоустойчивое кодирование [1].

Исследование помехоустойчивых алгоритмов существенно усложняется почти полным отсутствием доступных программных продуктов, позволяющих проводить их комплексный анализ. В связи с этим актуальной является задача создания прикладного приложения, которое дало бы возможность быстро реализовать ключевые элементы алгоритма, оценить эффективность новых решений в масштабах всей системы.

Схема системы, работу которой должно моделировать описанное приложение, представлена на рис. 1 [2].

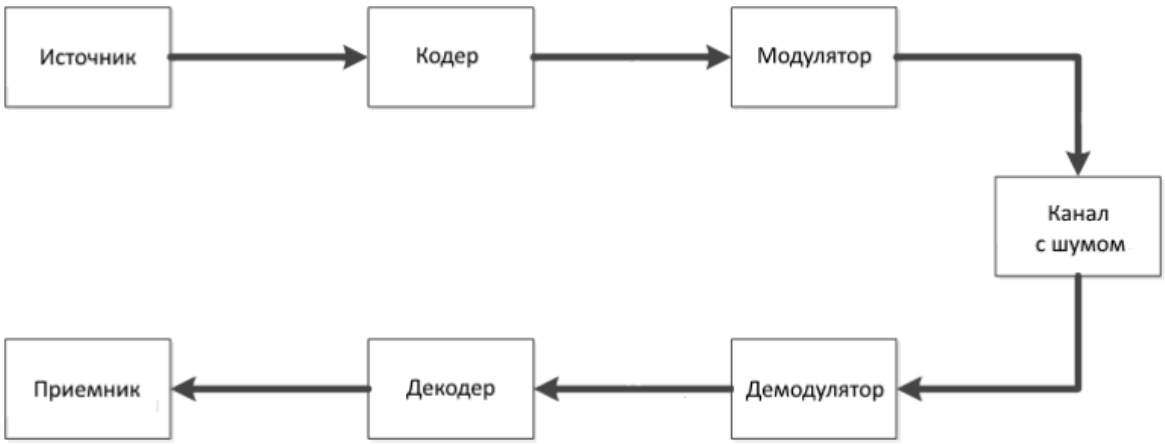

Рис. 1. Схема системы передачи цифровой информации

Модель системы передачи цифровой информации состоит из семи основных блоков.

Источник генерирует сообщение в виде двоичных символов.

В задачи кодера входит внесение в сообщение некоторой избыточности, которую декодер сможет использовать для исправления возникших ошибок.

Модулятор реализует отображение данных с кодера в аналоговый сигнал.

В канале с шумом сигнал подвергается негативным воздействиям, для оценки которых обычно используют отношение *сигнал-шум*, выражаемое в децибелах.

Демодулятор преобразует принятый сигнал в последовательность чисел, представляющих оценку передаваемых данных.

Выход демодулятора поступает на декодер, который, используя внесенную кодером избыточность, определяет переданное источником сообщение.

Основная сложность реализации модели системы передачи данных приходится на реализацию кодера и декодера, поэтому остальные блоки системы в программной реализации были выполнены как

подключаемые модули в целях упрощения хода разработки, а для кодирования и декодирования информации используется Код Хэмминга как наиболее известный и простой в реализации из самоконтролирующихся и самокорректирующихся кодов, применимых к двоичной системе счисления.

Представленную разработку отличает возможность динамического подключения компонентов системы передачи данных, что позволяет в режиме реального времени подключать и тестировать различные библиотеки, описывающие алгоритмы кодирования информации.

Приложение должно удовлетворять следующим требованиям:

- возможности настройки параметров компонентов системы;
- возможности подключения библиотек компонентов системы;
- получению статистики о ходе моделирования.

Общая схема архитектуры приложения представлена на рис. 2.

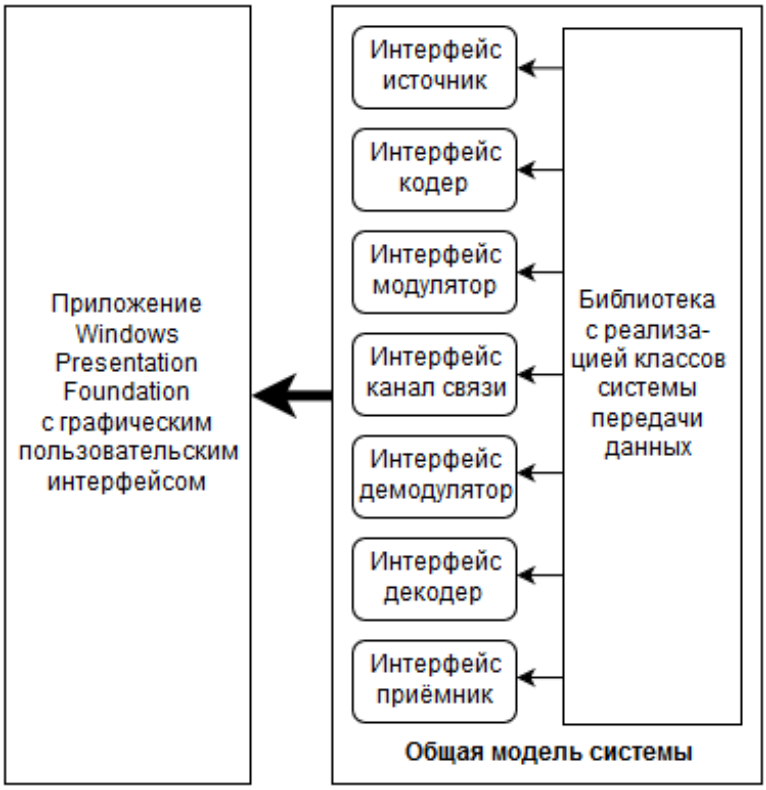

Рис. 2. Общая схема архитектуры приложения

В качестве языка разработки был выбран C#, который полностью реализует идеи объектно-ориентированного программирования, что значительно упрощает реализацию многих решений, имеющих место в разрабатываемом приложении. Также платформа .Net предоставляет обширные функциональные и постоянно расширяемые библиотеки для работы с файловой системой, графическим интерфейсом, и локальной сетью, что позволяет значительно ускорить процесс разработки.

Следующей проблемой на пути исследований являются требующиеся при исследовании систем передачи данных огромные объемы экспериментов. В процессе реализации тестовых моделей были получены результаты моделирования, показывающие, что выполнение всех вычислений только средствами CPU (central processing unit) среднестатистического персонального компьютера не может предоставить достаточных возможностей для проведения насколько-то эффективного и наглядного исследования, результаты которого могут говорить об эффективности алгоритма.

Основной причиной такой медленной работы является то, что CPU – это универсальное вычислительное устройство, нацеленное на выполнение широкого круга задач и не оптимизированное для выполнения операций, характерных для работы кодера и декодера.

Исходя из вышесказанного, необходимо найти возможности по увеличению скорости моделирования систем передачи данных.

Задача ускорения вычислений требует использования принципиально других подходов к организации вычислений. Перспективным направлением ускорения вычислений является использование незадействованных ресурсов гетерогенных компьютерных систем, в частности вычислительных ресурсов графических процессоров GPU (graphic processing unit). В случае достаточного параллелизма

реализуемых вычислений применима техника General-purpose computing forgraphics processing units (GPGPU).

В настоящее время для вычислений на GPU активно продвигаются две технологии CUDA и OpenCL.

- CUDA (Compute Unified Device Architecture) – программно-аппаратная архитектура параллельных вычислений, которая позволяет существенно увеличить вычислительную производительность благодаря использованию графических процессоров фирмы NVidia [3].
- OpenCL (Open Computing Language) – это технология, реализующая параллельные компьютерные вычисления на различных типах графических и центральных процессоров.

Обе описанные технологии представляют схожий функционал для активного и эффективного использования вычислительных ресурсов GPU. Существенным отличием является то, что поддержка CUDA реализована только на видеокартах производства компании NVidia. Данный факт существенно сужает парк компьютеров, который может использоваться для моделирования. В то же время OpenCL поддерживается практически всеми производителями GPU. Казалось бы, выбор очевиден в сторону OpenCL, но компания NVidia разработала библиотеку CUDAfy.NET, которая содержит генератор кода для CUDA и OpenCL [4].

Для экспериментов использовался процессор Intel(R) Core(TM) i5-3230M 2,60GHz (двухъядерный процессор) и видеокарта NVidia GeForce 720M с 96 ядрами в процессоре. Использование технологии GPGPU позволило увеличить скорость моделирования передачи данных по каналу связи при использовании OpenCL с 400кбит/с до 10мбит/с, и до 11.5мбит/с при использовании CUDA.

Таким образом, в процессе исследования была изучена система передачи данных, разработана гибкая модульная архитектура модели системы моделирования данных. Найдены способы увеличения скорости моделирования через задействование вычислительных мощностей GPU. Полученные результаты дают основание считать перспективным использование технологии GPGPU для решения задачи уменьшения временных затрат на ресурсоемкое моделирование системы передачи данных.

Литература

1. *Золотарев В. В., Зубарев Ю. Б., Овечкин Г. В.* Многопороговые декодеры и оптимизационная теория кодирования – М.: Горячая линия – Телеком, 2012. – 239 с.
2. *Золотарев В. В., Овечкин Г. В.* Помехоустойчивое кодирование. Методы и алгоритмы: справочник. М.: Горячая линия – Телеком, 2004. 126 с.
3. *Боресков А. В., Харламов А. А., Марковский Д. А., Микушин Д. Н., Мортиков Е. В., Мыльцев А. А., Сахарных Н. А., Фролов В. А*. Параллельные вычисления на GPU. Архитектура и программная модель CUDA: учебное пособие. – М.: Изд-во Московского университета, 2012. – 336 с.
4. CUDAfy.NET [Электронный ресурс]: Документация и исходный код. URL: https://cudafy.codeplex.com (дата обращения: 15.04.2016).