

Программное обеспечение устройств сопряжения мультиплексного канала обмена

*Д. Е. Гурьев, ВМиК МГУ, Москва, gouriev@oit.cmc.msu.ru,
В. Е. Лызлов, ЗАО НТЦ «Модуль», Москва, lve@module.ru,
Н. Ю. Миронов, ЗАО НТЦ «Модуль», Москва, mironov@module.ru,
В. А. Харин, ЗАО НТЦ «Модуль», Москва, vharin@module.ru,
Д. А. Чихичин, ЗАО НТЦ «Модуль», Москва, dch@module.ru*

Этот доклад посвящен программному обеспечению (ПО), обеспечивающему эксплуатацию линейки устройств сопряжения мультиплексного канала обмена (МКО) [1]. Рассматриваемое ПО обеспечивает программное и интерактивное управление устройствами в наземных комплексах отладки целевых систем. Этот проект развивается уже 6 лет. В докладе впервые представлены основные результаты, а также обсуждаются некоторые планы ближайшей перспективы.

Особенности протокола и устройств сопряжения

Протокол «Интерфейс магистральный последовательный системы электронных модулей», известный также как ГОСТ Р 52070-2003 [2] и MIL-STD-1553B [3], применяется для организации бортовых сетей летательных аппаратов различного назначения, а также для бортовых сетей кораблей и судов или сетей управления технологическими процессами.

MIL-STD-1553B – сравнительно несложный протокол. Он имеет несколько особенностей из которых для разработчиков программного обеспечения существенны следующие:

- предсказуемость в соответствии с требованиями «жесткого» реального времени, что обеспечивается наличием специального абонента – «контроллера шины» (КШ), управляющего передачей данных между всем абонентами в соответствии с заранее заданным планом; план пересылок в терминах MIL-STD-1553B называется «кадром»,
- наличие еще 2х типов абонентов: «оконечное устройство» (ОУ) – обычный управляемый абонент, – и «монитор шины» (МТ), подслушивающий пересылки в целях диагностики и отладки).

В отличие от самого протокола, устройства сопряжения [1] достаточно сложные. Прием и передача сообщений, исполнение кадров реализованы в них на аппаратном уровне. Элементами интерфейса с управляющим вычислителем (УВ) являются сущности канального уровня: сообщения, кадры, журналы приема-передачи (в терминологии разработчиков устройств – «стеки команд»). В составе устройств имеется

собственное ОЗУ и локальная магистраль. Каждое из устройств поддерживает режимы КШ, ОУ и МТ, в каждом из которых имеются дополнительные подрежимы и функции.

Конфигурирование устройств – с одной стороны – происходит на достаточно высоком уровне – почти на уровне канального протокола, – а с другой стороны – связано со очень значительным числом технических деталей. Ситуация содержит определенный вызов разработчикам ПО, ответ на который: детали по возможности скрыть, а уровень интерфейса – еще больше приблизить к протокольному.

Программное обеспечение, поставляемое с устройствами, состоит из комплекта драйверов, библиотеки времени выполнения, обеспечивающей программное управление устройствами, и прикладного ПО с графическим пользовательским интерфейсом, позволяющим интерактивно конфигурировать устройства и тестировать передачу данных в бортовой сети или ее макете.

Библиотека времени выполнения

Основой системного ПО является библиотека времени выполнения (RTL), которая обеспечивает:

- доступ к управляющим регистрам и встроенному ОЗУ устройства;
- конфигурирование устройства для работы в каждом из режимов КШ, ОУ или МТ, включая конструирование кадров, а также управление подрежимами и дополнительными функциями;
- задание пользовательской подпрограммы обработки прерываний от устройства;
- предоставление единого процедурного интерфейса (API) ко всем устройствам линейки, скрывая их технические различия;
- сокрытие значительного числа других технических деталей;
- дополнительный уровень абстракции, приближенный к терминам протокола, в том числе – API для работы с объектами «сообщение», «кадр», «стек», размещенными в ОЗУ устройства;
- автоматизированное распределение памяти в ОЗУ устройства;
- контроль параллельного доступа к разделяемым объектам в ОЗУ устройства со стороны протокольной логики устройства и управляющего вычислителя;
- согласование темпа передачи данных из и в устройство (путем искусственного введения холостых циклов шины УВ) и другие действия, обусловленные аппаратными особенностями устройств;
- одновременную работу с любым числом устройств;
- и многое другое.

В состав библиотеки также входит специальный модуль для управления устройством «тестер протокола» [1] для исполнения тестов соответствия протоколу MIL-STD-1553B.

При разработке библиотеки предъявлялись и были выполнены следующие технологические требования:

- возможность переноса в различные ОС и среды исполнения, в том числе такие, в которых отсутствует полноценная ОС и стандартные библиотеки;
- поддержка всех устройств линейки и постепенное накопление «знаний» об особенностях их работы и применения в едином комплекте исходных кодов;
- функциональная расширяемость и единый стиль построения API при функциональных расширениях.

Язык реализации – С. Предусмотрены специальные «прослоечные» модули для обеспечения независимости от ОС, компилятора и стандартной библиотеки С. При проектировании API произведен тщательный выбор независимых и свободно сочетаемых примитивных операций. Доступ к объектам в ОЗУ устройства осуществляется при помощи дескрипторов, между типами которых установлено отношение наследования; имеются полиморфные функции API. Библиотека в эксплуатации с 2000 года.

Драйверы

Сложность устройства и управления им, а также возможное отсутствие ОС (следовательно, и драйверов) на перспективных целевых платформах, привели к решению сосредоточить основную функциональную нагрузку в библиотеке времени выполнения и максимально разгрузить драйвер устройства. В текущих реализациях функции драйвера ограничены:

- обеспечением и контролем монопольности доступа к устройству со стороны приложений;
- картированием управляющих регистров и встроенного ОЗУ устройства в адресное пространство приложения;
- при возникновении прерываний от устройства – при необходимости – передачей управления подпрограмме обработки прерываний, определенной приложением (и исполняющейся в адресном пространстве приложения).

Разработаны драйверы для ОС MS Windows 98, ОС MS Windows 2000 и ОС MS Windows 2000 с расширением реального времени RTX [4]. В ближайших планах – разработка драйверов для ОС Linux: для стандартного ядра и ядра с расширением реального времени RTAI [5].

Прикладное программное обеспечение

Прикладное программное обеспечение, поставляемое вместе с устройствами, по природе вещей является «нецелевым». Его задача – помочь разработчику целевой системы убедиться в работоспособности отдельных устройств-абонентов и сети (или ее макета) в целом. В настоящее время с устройством поставляются 3 интерактивные графические программы, обеспечивающие управление устройством в режимах КШ, ОУ и МТ соответственно, и выполняющиеся на платформе MS Windows. Каждое из приложений позволяет сконфигурировать устройство, задать тестовые данные для передачи, получать в реальном времени информацию об ошибках передачи данных, приложение для режима МТ – сохранять трассу передачи данных для последующего анализа. По историческим причинам эти приложения разрабатывались в разное время независимыми разработчиками.

На основе опыта использования этих программ было осознано, что прикладное ПО, поставляемое с устройствами, должно:

- эффективнее способствовать разработки целевых систем, быть точнее нацеленным на задачи, решаемые разработчиком целевых систем;
- быть удобнее в использовании;
- быть теснее интегрированным внешне, с точки зрения пользователя (одно многофункциональное приложение, а не три разных; единообразные интерфейсные решения для управления аналогичными сущностями);
- быть теснее интегрированным внутренне (повторное использование функционального кода, форматов данных и самих данных).

Кроме этого, необходимы версии интерактивных программ и для других ОС, в первую очередь – для ОС Linux.

Для конкретизации этих пожеланий был проведен анализ типичных областей применения интерактивных программ, сформирован перечень типовых задач и разработана типовая архитектура таких приложений.

Областями применения интерактивных программ являются стенды разработчиков целевых систем и наземные комплексы отладки. Для целей разработки важно иметь возможность быстро смонтировать будущую бортовую сеть и проверить ее работоспособность, убедиться в работоспособности своего устройства в ней. К сожалению, после этого макет сети приходится реализовывать заново в своем оборудовании или ПО уже без помощи интерактивных программ. Более эффективным был бы процесс разработки, при котором части макета можно было бы постепенно переносить в собственное оборудование или ПО.

На более глубоких стадиях разработки необходимо создавать для целевого устройства среду, имитирующую работу других абонентов сети, в том числе и содержание передаваемых сообщений.

Для диагностики и отладки сетевых отказов необходимы развитые средства сбора и анализа трасс сетевых взаимодействий.

Резюмируя сказанное, задачи интерактивного ПО должны быть следующими:

- быстрое и эффективное макетирование бортовых сетей:
 - графическое проектирование структуры сети,
 - интерактивное управление конфигурациями нескольких устройств из одного приложения,
 - возможность сохранения конфигураций с управлением версиями,
 - поддержка устройства MB26.17 [1], специально предназначенного для имитации нескольких абонентов сети;
- возможность статического задания тестовых передаваемых данных;
- возможность задания расчетного правила для тестовых передаваемых данных, с учетом полученных данных (фактически – модели поведения абонента);
- возможность экспорта в том или ином виде конфигурации устройств для последующего встраивания в целевое аппаратное или программное обеспечение;
- возможность сбора и анализа трасс передачи данных, включая отбор по сложным условиям при записи и в постобработке;
- возможность воспроизведения записанных трасс.

В основу типовой архитектуры прикладного ПО положен принцип нескольких уровней:

- уровень взаимодействия с устройствами представлен библиотекой времени выполнения;
- уровень данных обеспечивает сохранение, чтение, объединение/разбиение и другие операции над данными; данных в этих приложениях достаточно много (конфигурации, трассы, тестовые данные для передачи и др.) и они достаточно сложно организованы; в качестве основного формата данных предложен XML;
- функциональный уровень, к которому относится код, реализующий назначение приложения;
- уровень пользовательского интерфейса.

Естественно, предполагается, что уровни реализуются по возможности независимо друг от друга, а компоненты и решения каждого уровня должны быть унифицированы и использоваться повторно в разных приложениях.

Отдельного внимания заслуживает проблема создания приложений для разных платформ. Идеальным было бы решение, позволяющее

скомпилировать один и тот же комплект исходных кодов под разные платформы. Функциональный уровень практически не зависит от платформы, уровень данных также можно реализовать как независимый при аккуратном выборе вспомогательных библиотек (в частности, для разбора и генерации данных в формате XML), в библиотеке времени выполнения проблемы мобильности решены. Наиболее существенную проблему создает уровень пользовательского интерфейса, но и здесь возможны приемлемые решения. В настоящее время производится изучение нескольких платформно-независимых библиотек графических компонентов с целью выбора в качестве внутривнепроектного стандарта разработки, в частности рассматриваются Trolltech QT [6] и Borland CLX [7].

Изложенные цели и принципы будут положены в основу разработки новых и модернизации имеющихся интерактивных прикладных программ управления устройствами сопряжения мультиплексного канала обмена, которые будут осуществлены в ближайшей перспективе.

Заключение

Рассмотрены основные компоненты программного обеспечения устройств сопряжения мультиплексного канала обмена [1], функции, технологические особенности и принципы разработки этих программных компонентов. Предложена архитектура для интерактивных прикладных программ управления устройствами сопряжения. Обсуждены задачи, решение которых планируется в ближайшей перспективе.

Литература

1. Гурьев Д.Е., Демьянов П.Ю., Миронов Н.Ю., Харин В.А. Разработка устройств сопряжения мультиплексного канала обмена (MIL-STD-1553). Наст. сборник.
2. ГОСТ Р 52070-2003. Интерфейс магистральный последовательный системы электронных модулей. Общие требования.
3. MIL-STD-1553B NOTICE 4. DIGITAL TIME DIVISION COMMAND/RESPONSE MULTIPLEX DATE BUS. DOD, 1996.
4. Ardence RTX Is The Real-Time Performance Solution For Microsoft Windows. <http://www.ardence.com/embedded/products.aspx?ID=70>
5. RTAI. <http://www.rtai.org/>
6. Qt Overview. <http://www.trolltech.com/products/qt/index.html>
7. Why Develop Applications with Borland Kylix 3? Borland Software Corporation, 2002. http://www.borland.com/resources/en/pdf/white_papers/why_develop_applications_with_borland_kylix3.pdf