Применение ARM-процессоров компании «Миландр» под управлением ОС FreeRTOS в приборах учета электроэнергии

Приборы учета электроэнергии компании «Миландр» — «Милур 107» и «Милур 307» — разрабатываются в рамках стартовавшей в 2014 году комплексной программы «Реализация комплексного проекта по созданию высокотехнологичного производства интеллектуальных приборов энергоучета, разработанных и изготовленных на базе отечественных микроэлектронных компонентов, и гетерогенной автоматизированной системы мониторинга потребляемых энергоресурсов на их основе» (шифр «Комплексное импортозамещение» по договору с Минобрнауки на основании постановления Правительства Российской Федерации № 218).

Алексей БОРОЗДИН
borozdin.a@ic-design.ru
Юрий САХНО

Приборы учета «Милур» могут применяться автономно, как обычные счетчики электроэнергии, но их основное назначение — работа в составе автоматизированной системы коммерческого учета энергоресурсов АСКУЭ. Данная система позволяет в автоматическом режиме собирать данные от приборов учета электроэнергии по проводным (PLC или RS-485) или беспроводным интерфейсам (ZigBee), обрабатывать их, хранить и предоставлять в удобной форме для дальнейшего использования. АСКУЭ состоит из многих компонентов, и ее базовым звеном являются приборы учета.

Приборы учета электроэнергии «Милур 107» и «Милур 307» построены на базе специализированных 32-разрядных микроконтроллеров K1986BE23 и K1986BE21, которые являются собственной разработкой компании «Миландр». Микроконтроллер K1986BE23 предназначен для построения однофазных счетчиков электроэнергии и содержит 3 канала 24-битных ΣΔ АЦП. Микроконтроллер K1986BE21 создан для трехфазных счетчиков и содержит семь каналов 24-битных ΣΔ АЦП. Оба микроконтроллера имеют одинаковое ядро ARM Cortex-M0, 16 кбайт ОЗУ, 64 либо 128 кбайт Flash-памяти программ и работают на тактовой частоте 36 МГц [1, 2]. Аппаратные ресурсы микроконтроллеров позволяют решить любые задачи учета электроэнергии. Важнейшим вопросом является программное обеспечение, которое в конечном итоге определяет функциональные возможности приборов.

Программное обеспечение, управляющее приборами учета электроэнергии «Милур 107/307», можно условно разделить на две части — метрологическую и прикладную. Метрологическая часть обрабатывает данные, поступающие с аппаратных блоков АЦП микроконтроллера в соответствии с тарифами и расписанием, формирует интервальные срезы мощности, а также управляет импульсными выходами счетчика. Прикладная часть ПО реализует сервисные функции, отвечает за взаимодействие с пользователем и обмен через внешние интерфейсы. И метрологическая, и прикладная части ПО имеют модульную структуру, в которой каждый программный модуль решает узкоспециализированную часть общей задачи. Большинство программных модулей выполняется асинхронно и параллельно с другими модулями, и важно правильно разделить аппаратные ресурсы микроконтроллера между ними, поскольку от этого зависит общая эффективность и надежность устройства. Особенно это актуально для различных протоколов связи. Традиционно ПО встраиваемых систем реализуют как набор программных модулей, причем каждый модуль содержит конечный автомат, управляющий тем или иным процессом. В главном цикле все конечные автоматы вызываются последовательно, с известным интервалом. В дополнение к автоматам используются обработчики прерываний, работающие непосредственно с аппаратными блоками и обеспечивающие привязку к реальному времени. В самом простом случае обмен данными между обработчиками прерываний, а также между программными модулями идет через глобальные переменные или флаги. Помимо этого, возможны вариации такого подхода, когда дополнительно вводится система сообщений, и обмен данными ведется через диспетчера сообщений. В целом такой подход похож на применение ОС с корпоративной многозадачностью — обеспечивается детерминизм системы, но возникают проблемы с расширяемостью кода. При добавлении новых функций нужно обязательно учитывать максимально допустимое время работы модуля в контексте всей системы. Если решается продолжительная вычислительная задача, которую невозможно выполнить за один квант времени, ее приходится разбивать на более мелкие и выполнять эти части последовательно. Такая архитектура ПО хорошо подходит для относительно простых систем с небольшим числом параллельных задач.

Альтернативой традиционному подходу является применение операционной системы с вытесняющей многозадачностью, что позволяет иначе организовать работу программных модулей. Функции системы распределяются между отдельными потоками, при этом все потоки выполняются псевдопараллельно, обмениваясь данными через механизмы передачи сообщений и объекты синхронизации. Потокам присваиваются приоритеты, в зависимости от которых планировщик ОС распределяет процессорное время. Применение ОС также способствует разделению задач между специалистами при создании кода. Условно процесс разработки ПО на базе ОС можно разбить на два этапа:

1. Разрабатывается базовая структура ПО. Портируется на выбранную платформу ОС и разрабатываются драйверы. Эту задачу выполняет системный инженер,

который хорошо знаком с архитектурой микроконтроллера. В результате данного этапа получается платформо-зависимый код, выполняющий роль слоя абстракций для вышестоящих алгоритмов.

2. Разрабатываются управляющие алгоритмы. За счет отсутствия привязки к конкретному аппаратному обеспечению появляется возможность повторного использования этих модулей.

К негативным аспектам применения ОС можно отнести сложность разработки. Многие проблемы традиционного подхода снимаются, но могут возникнуть другие, присущие всем многопоточным приложениям, в частности, разграничение доступа к разделяемым ресурсам, потенциальные взаимоблокировки. Кроме того, ОС предполагает дополнительные накладные расходы для работы ядра — как процессорного времени, так и памяти. Эти требования варьируются от одной системы к другой.

В большинстве микроконтроллерные системы являются системами жесткого реального времени, то есть невозможность обеспечить реакцию на какие-либо события в заданное время ведет к отказам и невыполнимости поставленной задачи. Для таких систем применяют операционные системы реального времени. В качестве основного требования к ОСРВ выдвигается условие обеспечения предсказуемости или детерминированности поведения системы во всех случаях. Как правило, такие системы не занимают много места в памяти программ (типовые значения 4–20 кбайт), а также нетребовательны к ресурсам памяти [4].

В приборах учета электроэнергии «Милур 107» и «Милур 307» в качестве системного программного обеспечения используется операционная система реального времени FreeRTOS [5] — это ОСРВ с вытесняющей многозадачностью, полностью открытым исходным кодом, имеющая богатый API и при этом предъявляющая минимальные требования к аппаратуре. FreeRTOS поддерживает большое количество архитектур, в том числе ядро ARM Cortex-M0, применяемое в МК К1986ВЕ21 и К1986ВЕ23. Согласно лицензионному соглашению FreeRTOS, при использовании этой ОС в своем проекте нет необходимости открывать проприетарный код, задействующий API ОС. Нужно открыть лишь исходники самой ОС при условии внесения в них изменений. FreeRTOS поставляется как набор исходных С-файлов, которые компилируются вместе с кодом проекта. Функциональные возможности конфигурируются в заголовочном файле, что позволяет исключить ненужный код из компиляции и тем самым уменьшить требуемый объем памяти.

В терминах ОС каждый программный модуль представляет собой задачу (task), реализующую определенный функционал и взаимодействующий с другими задачами. Задача

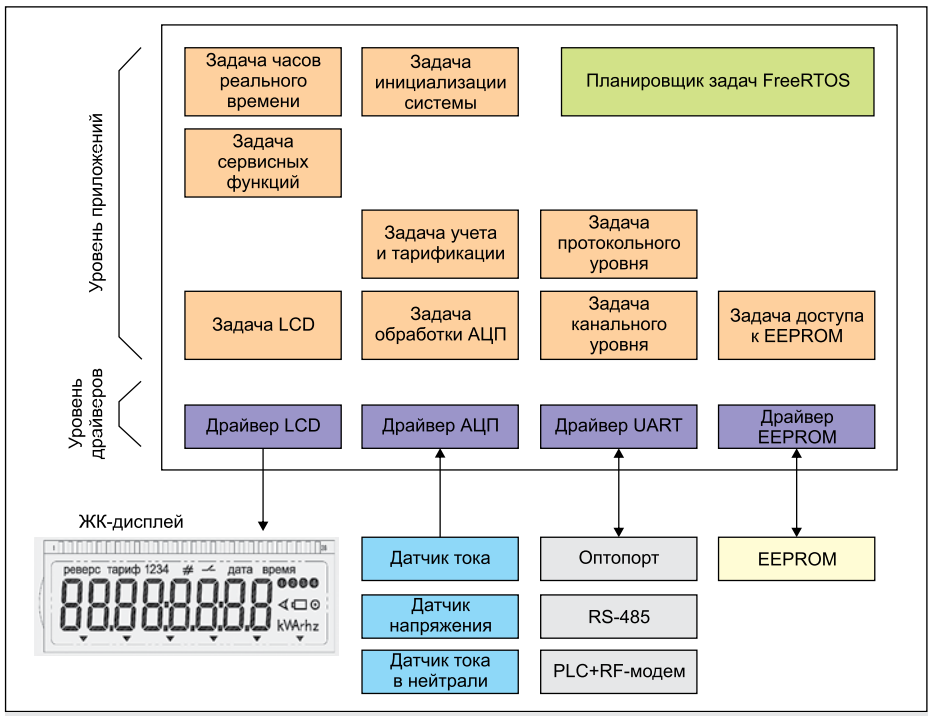

Рисунок. Структурная схема ПО приборов учета электроэнергии «Милур» на базе FreeRTOS

может агрегировать определенный набор функций, выступая как источник событий или данных или быть посредником при доступе к аппаратным ресурсам (так называемый gatekeeper). Количество и выполняемые функции задач выбирают в зависимости от требований к системе так, чтобы максимизировать модульность и автономность кода. Например, в текущей версии ПО приборов учета электроэнергии «Милур» параллельно выполняется восемь задач. При запуске системы дополнительно реализуется задача инициализации системы, которая после работы удаляется (рисунок).

Внутренние данные задач, содержащие детали реализации, сокрыты от других программных модулей. Взаимодействие строится унифицированным образом — на основе механизма сообщений, при этом синхронизация через другие глобальные примитивы сводится к минимуму. Логически обмен данными между задачами может строиться по модели с «проталкиванием» данных (push) или с «вытягиванием» (pull). В зависимости от ситуации применяется как push-, так и pull-обмен данными. Например, задача, обслуживающая часы реального времени, периодически отправляет (push) специальное сообщение задаче LCD, которое инициирует обновление дисплея. Задача LCD, в свою очередь, в зависимости от отображаемого пункта меню и внутреннего состояния запрашивает данные у других задач (pull). Декомпозиция всего функционала ПО должна выполняться так, чтобы по возможности избегать проблем, связанных с многопоточностью и доступом к разделяемым ресурсам.

Очевидно, что наборы программных модулей для трехфазных и однофазных приборов учета схожи. Исходя из этого было принято решение использовать один набор задач для разных моделей приборов учета, заменяя лишь драйверы, непосредственно работающие с аппаратурой, а также внося минимальные изменения в код при помощи директив условной компиляции в зависимости от модели. Это значительно упростило разработку и поддержку ПО, так как объем создаваемого и тестируемого кода существенно сократился. Кроме того, за счет инкапсуляции внутренних данных внутри задач и применения стандартизованного интерфейса обмена данными через сообщения можно легко заменить один модуль на другой, в частности, при переходе на другой тип LCD, при подключении нового внешнего модема или расширении существующей функциональности. При этом такая модификация не затрагивает отлаженные модули, а значит, ускоряется процесс верификации обновленного ПО.

При разработке программного обеспечения для приборов учета «Милур 107/307» применялась стратегия «от общего к частному». На основе существующего решения и требований была проведена декомпозиция на задачи в терминах ОС, проработан интерфейс обмена данными и синхронизации между задачами. В результате проект представлял собой ядро ОС + набор задач, реализующих разработанный интерфейс, но содержащих заглушки на месте реальных вычислений + некоторый базовый набор драйверов, необходимых для минимальной работоспособности. Далее параллельно несколькими инженера-

Таблица. Усредненные показатели потребления ресурсов системы

Задача	Потребление стека, байт	Загрузка ядра, %
Предварительная обработка данных от АЦП	176	0,06
Учет и тарификация электроэнергии	328	0,04
Обслуживание часов реального времени	188	0,7
Обслуживание энергонезависимой памяти	144	0,42
Сервисные функции	368	1,19
LCD-дисплей и пользовательский ввод	160	0,13
Канальный уровень коммуникаций	184	3,71
Протокольный уровень коммуникаций	208	0,11
Системный процесс IDLE (простой системы)	88	93,59

ми велась работа над написанием драйверов аппаратных блоков и наполнением функционалом задач. По мере готовности код объединялся, тестировался в составе системы и при необходимости корректировался.

API операционной системы FreeRTOS позволяет собрать статистику по работе планировщика, потреблению стека и общей памяти задачами, а также по временным ресурсам (таблица).

Поскольку большинство событий, требующих обслуживания, возникает регулярно, но относительно редко, основную часть времени вычислительное ядро простаивает. Суммарный объем кода, занимаемый ОС в итоговом образе, равен 4676 байт, что составляет менее 8% всей памяти программ микроконтроллера К1986ВЕ23 и менее 4% для К1986ВЕ21.

Подводя итог, можно сказать, что выбор ОСРВ для приборов учета электроэнергии «Милур 107/307» повысил модульность, надежность и расширяемость кода, а также улучшил поддержку и сопровождение проекта при приемлемых аппаратных затратах. Имеется большой потенциал для наращивания возможностей на существующей аппаратной платформе.

Литература

1. Спецификация: Микроконтроллер 1986ВЕ23У для применения в однофазных электросчетчиках на базе процессорного ядра ARM Cortex-M0.
2. Спецификация: Микроконтроллер 1986ВЕ21У для применения в электросчетчиках на базе процессорного ядра ARM Cortex-M0.
3. Oshana R., Kraeling M. Software Engineering for Embedded Systems. Methods, Practical Techniques, and Applications, 2013.
4. Бурдонов И. Б., Косачев А. С., Пономаренко В. Н. Операционные системы реального времени. М.: Институт системного программирования РАН, 2006.
5. www.freertos.org