УДК 004.4'6

А.А. Овчинников, А.В. Уткин

Применение .NET Micro Framework для разработки встраиваемых информационных (в т.ч. RFID) систем

Современные информационные системы невозможно представить без использования микроконтроллеров. Тем не менее, разработка логики работы таких систем может стать довольно трудоёмкой и долгой задачей. В качестве решения предлагается использовать среду выполнения Microsoft .NET Micro Framework, позволяющую повысить эффективность и упростить процесс разработки информационных систем, в том числе RFID систем.

программирование микроконтроллера, цифровое устройство, среда выполнения, радиочастотная идентификация, RFID система

Разработку современных интеллектуальных устройств довольно трудно представить без использования микроконтроллеров. Цифровые устройства, управляемые центральным процессором, применяются во всём многообразии разрабатываемой в настоящее время техники. Микроконтроллеры определяют основные параметры современных устройств, которые отличаются простотой управления, небольшой массой и объёмом, повышенной точностью и многофункциональностью. Кроме того, такие устройства выгодны и в экономическом плане: микроконтроллеры являются достаточно дешёвой альтернативой аналоговым многокомпонентным устройствам. Это позволяет снизить стоимость производства систем с микроконтроллерным управлением и, как следствие, их цену для конечного потребителя.

Современное развитие микроконтроллеров позволяет применять их практически в любом цифровом устройстве с высокой степенью эффективности. Однако для создания даже простейшей системы, управляемой микроконтроллером, требуется выполнение специальных требований по программированию того или иного центрального процессора. Причём эти требования могут зависеть как от самой архитектуры, так и от модели процессора. Например, для описания логики работы процессора используется язык ассемблера, набор команд которого для каждой архитектуры процессора может быть свой, не говоря уже о мощности и скорости исполнения команд.

Первой эту проблему попыталась решить компания Sun Microsystems, выпустив программную платформу под названием Java. Особенностью Java было наличие так называемой виртуальной машины, части исполняющей среды Java (Java Runtime Environment, JRE). Разработчик писал программный код на языке Java, который затем компилировался в кроссплатформенный (не зависящий от платформы) байт-код, а виртуальная машина интерпретировала этот байт-код, то есть выполняла его непосредственно на имеющемся процессоре. Такой способ позволял описать логику выполнения один раз и впоследствии использовать части написанного кода в других программах, для других устройств и процессоров, применяемых в самой различной технике, в том числе и бытовой.

Корпорация Microsoft разработала свою стратегию Microsoft.NET, предполагающую создание платформы, которая должна работать не только на персональных компьютерах, но и в портативных устройствах. Технология .NET Framework, являющаяся частью стратегии Microsoft.NET, преследует ту же цель, что и Java: программный код пишется один раз и исполняется затем на разных программных и аппаратных платформах. Архитектура .NET Framework в целом очень схожа с архитектурой платформы Java. Разработчик пишет программный код (логику программы) на одном из поддерживаемых языков программирования, затем этот код компилируется в байт-код CIL (Common Intermediate Language – общий промежуточный язык, ранее MSIL – Microsoft Intermediate Language), который в конечном итоге интерпретируется виртуальной машиной CLR (Common Language Runtime – общеязыковая среда выполнения). Как и в Java, .NET Framework использует JIT-компиляцию (Just-In-Time – «на лету»): байт-код интерпретируется по мере выполнения программы. Ввиду такой архитектуры, программный код, написанный разработчиком, называется также управляемым кодом, т.к. за его выполнение полностью отвечает CLR.

.NET обладает некоторыми преимуществами по отношению к Java. Например, платформа Java ограничена использованием лишь одноимённого языка программирования, а технология .NET Framework предоставляет разработчику достаточно широкий выбор инструментария. Помимо этого, среда .NET Framework предоставляет возможность взаимодействия между несколькими поддерживаемыми языками программирования. Так, программа, написанная на языке Visual Basic .NET, может легко взаимодействовать с компонентами, написанными с использованием языка C#, и так далее. Более того, .NET Framework предоставляет широкие возможности взаимодействия сборок (assembly) в байт-коде CIL с нативными (native, в машинном коде) программами.

Отличительной особенностью технологии .NET Framework является наличие версии .NET Micro Framework, специально разработанной для встраиваемых устройств с малым объёмом памяти и невысокой производительностью и способной обеспечивать работу без операционной системы. Совместно с набором Porting Kit для портирования среды .NET Micro Framework на различные устройства .NET Micro Framework является тем самым решением, которое позволяет создавать микроконтроллерные системы, используя высокоуровневые языки программирования для описании логики работы микроконтроллера.

Архитектура .NET Micro Framework разделена на несколько слоёв. Нижний слой – это аппаратное обеспечение: центральный процессор и периферийные устройства. .NET Micro Framework поддерживает 32- и 64-разрядные процессоры различных архитектур, таких как ARM7, ARM9, Cortex-M. На данный момент существует несколько готовых решений для разработки встраиваемых устройств с использованием .NET Micro Framework. В основном это отладочные платы Netduino от компании Secret Labs, а также решения компании GHI Electronics. Кроме того, .NET Micro Framework может быть запущен и в среде операционной системы, например, Microsoft Windows.

Следующий слой, слой компонентов среды исполнения (Runtime Component Layer), включает в себя TinyCLR, слой аппаратных абстракций (HAL – Hardware Abstraction Layer) и слой абстракций платформы (PAL – Platform Abstraction Layer). TinyCLR является той самой уменьшенной версией виртуальной машины CLR, которая выполняет байт-код CIL на текущем аппаратном обеспечении. HAL и PAL непосредственно контролируют это аппаратное обеспечение. По сути, они содержат функции, которые вызывает TinyCLR, когда интерпретирует байт-код. Реализация PAL не зависит от типа аппаратной части, в то время как HAL для каждого аппаратного обеспечения свой. HAL также используется загрузчиком, который инициализирует устройства и запускает TinyCLR при включении питания.

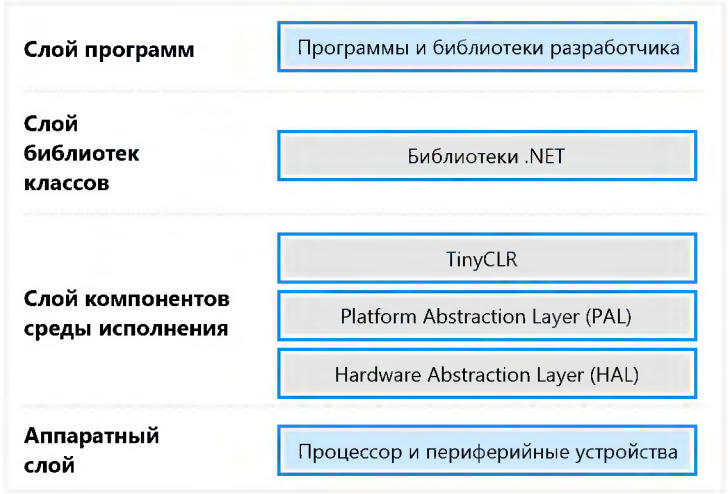

Рис. 1. Архитектура .NET Micro Framework

Дальше располагается слой системных библиотек классов .NET Micro Framework. Он включает стандартные наборы классов и функций, используемых программистом при разработке программного обеспечения микроконтроллерной системы. Эти библиотеки не зависят от аппаратной платформы, поскольку выполняются полностью в среде TinyCLR и могут быть использованы в любых устройствах, построенных с использованием .NET Micro Framework. Стандартные библиотеки содержат часть функций «взрослого» .NET Framework, а также предоставляют возможности шифрования, отладки, работы с графикой и так далее.

Наконец, верхний слой архитектуры .NET Micro Framework – это слой программ, создаваемых программистом данной микроконтроллерной системы в управляемом коде, то есть слой управляемых программ. .NET Micro Framework на данный момент поддерживает написание логики только на языке C#, о котором было упомянуто чуть выше. Разработчик может использовать существующие библиотеки классов .NET Micro Framework, создавать свои и использовать уже существующие классы, функции, писать драйвера (управляющие программы) для подключаемых к микроконтроллерной системе устройств и многое другое. Весь этот программный код не зависит от аппаратного обеспечения системы, поскольку выполняется в среде TinyCLR, и может быть использован при описании логики любых микроконтроллерных систем, поддерживающих .NET Micro Framework. Например, однажды написанный код протокола обмена информацией между управляющим устройством и центральным процессором системы может быть использован при создании других устройств управления.

Для портирования .NET Micro Framework на новые устройства, разработчик должен реализовать слой аппаратных абстракций (HAL) и загрузчик, соответствующие используемому аппаратному обеспечению. Кроме того, возможно изменение реализации TinyCLR в соответствии с нуждами разработчика. Слой абстракций платформы (PAL), как уже было сказано, не зависит от аппаратного обеспечения и не нуждается в портировании, а верхние слои (системных библиотек и управляемых программ) работают на TinyCLR и также не нуждаются в изменении до тех пор, пока текущая реализация TinyCLR этого не потребует (например, если разработчик решает сократить число функций TinyCLR в целях оптимизации использования памяти). Таким образом, архитектура .NET Micro Framework является переносимой на огромное количество различных устройств, что делает ее удобным инструментом для создания качественно новых информационных, например, RFID систем.

В частности, технология .NET использовалась для разработки Системы Аутентификации и Идентификации Продукции «ИСАИП». ИСАИП предназначена для аутентификации и идентификации маркированной радиочастотными метками продукции при помощи мобильных и стационарных радиочастотных считывателей, а также обеспечения защиты маркировки продукции от ее воспроизведения. Определение подлинности осуществляется путём идентификации и аутентификации уникального, защищенного электронной подписью, контента на радиочастотной метке, находящейся непосредственно на самом продукте или его упаковке.

Для того, чтобы проверить подлинность продукта, который покупатель держит в руках, он может поднести его к мобильному телефону, имеющему NFC (Near Field Communication) модуль, или к RFID терминалу с соответствующим программным обеспечением. ПО для такого терминала написано с использованием .NET Framework для операционной системы Windows. .NET позволяет выполнять все необходимые операции с самой меткой, а его возможности межъязыкового взаимодействия широко используются при подключении к устройствам чтения RFID контента, управляемым средствами операционной системы.

В дальнейшем предполагается развитие ИСАИП путём создания специального автономного устройства, выполняющего проверку соответствующих RFID меток. Оно может представлять собой некий «чёрный ящик» с двумя светодиодами, построенный на базе микроконтроллера, поддерживающего .NET Micro Framework. В таком случае логика функционирования данного устройства будет использовать достаточно обширную кодовую базу на языке C#, созданную для терминала ИСАИП, таким образом позволяя избежать написания новой логики, а также более эффективно разрабатывать и тестировать ПО для всех устройств системы ИСАИП. К таким алгоритмам относится, например, проверка данных RFID меток, которыми помечена продукция. Эта же кодовая база сможет быть задействована и в приложениях ИСАИП для некоторых мобильных платформ.

Таким образом, можно сказать, что использование .NET Micro Framework позволяет существенно повысить эффективность разработки микроконтроллерных систем за счёт сокращения трудозатрат программиста на описание логики работы систем, а также снизить затраты на создание новых устройств и время их разработки благодаря использованию отлаженных общих компонентов.

Список литературы

1. Шилдт, Герберт. C# 4.0: полное руководство: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011. – с. 33–35.
2. Троелсен, Эндрю. Язык программирования C# 2010 и платформа .NET 4.0.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011. – С. 48, 49, 53.

3. Understanding .NET Micro Framework Architecture. – MSDN [Электронный ресурс]. / URL: http://msdn.microsoft.com/en-us/library/jj646623

Открытое акционерное общество «Инженерно-маркетинговый центр Концерна «Вега»
(ОАО «ИМЦ Концерна «Вега»)
Статья поступила 01.03.2013.

Ovchinnikov A.A., Utkin A.V.

Using .NET Micro Framework for developing embedded information systems, including RFID systems

It's hard to imagine a modern information system without the use of microcontrollers. However, developing execution logic of those systems can become a time-consuming task. As a solution it is suggested to use a Microsoft .NET Micro Framework runtime which allows to increase efficiency and simplify the process of development of information systems, including RFID systems.

Microcontroller programming, digital device, runtime, radiofrequency identification, RFID system

Joint-stock Company «Engineering and Marketing Centre of Corporation «Vega»