Умные устройства безопасности

на микроконтроллерах Atmel

Бирюков А. А.

ДМК Пресс
издательство

УДК 621.398:654.924
ББК 32.968.9
Б64

Бирюков А. А.
Б64 Умные устройства безопасности на микроконтроллерах Atmel. – М.: ДМК Пресс, 2017. – 162 с.

ISBN 978-5-97060-558-5

В книге подробно рассматривается разработка устройств на базе микроконтроллеров Atmel. С их помощью предлагается собрать ряд полезных в быту устройств безопасности, таких как датчики света, температуры, кодовые замки и других.

Издание будет полезно как начинающим радиолюбителям, так и профессионалам.

УДК 621.398:654.924
ББК 32.968.9

ISBN 978-5-97060-558-5

СОДЕРЖАНИЕ

1 ЧТО ТАКОЕ МИКРОКОНТРОЛЛЕР ATMEL ATTINY13

На сегодняшний день на рынке присутствует множество различных микроконтроллеров. Помимо Atmel, существуют также PIC, чьи контроллеры более подходят для тиражирования устройств, STM, ориентированны на более сложные, промышленные устройства и другие решения. Atmel из этого ряда выделяют дешевизна, по сравнению с другими микроконтроллерами, простота в освоении. К тому же, на мой взгляд, именно младшие модели контроллеров Atmel из линейки ATtiny наиболее подходят для начинающих радиолюбителей, которые еще не владеют всеми тонкостями работы с микроконтроллерами.

Но, прежде чем начать обсуждение продукции компании Atmel, я хотел бы поговорить о том, что такое микроконтроллер и зачем он нужен.

1.1. Микроконтроллер – мозг устройства

Микроконтроллер – это небольшая микросхема, на кристалле которой реализован функционал крошечного компьютера. То есть внутри одной микросхемы смонтированы процессор, оперативная и энергонезависимая память, периферийные устройства, аналогово-цифровой и цифроаналоговый преобразователи (АЦП и ЦАП). При этом данные элементы работают и взаимодействуют между собой и внешним миром с помощью специальной микропрограммы, которая хранится внутри микроконтроллера.

Основное назначение микроконтроллеров – это управление различными электронными устройствами. В контексте моей книги микроконтроллеры (МК) используются для управления устройствами безопасности. То есть именно он решает, какое значение входного параметра является поводом для выполнения тех или иных охранных действий. Например, именно МК решает, что надо включить сигнализацию, когда значение температуры, передаваемое термодатчиком, превысит некоторую пороговую величину.

Таким образом, можно без преувеличения сказать, что микроконтроллер является мозгом устройства (рис. 1.1).

Так выглядят современные микроконтроллеры.

В свою очередь, за правильность функционирования этого мозга отвечает программа, которая в него зашита. В обычных микросхемах функционал жестко зашивается при производстве и не может быть никак изменен. На микроконтроллер прошивку можно записывать многократно. Но не стоит считать МК «маленькими флешками». У МК гораздо меньше рабочих циклов записи, чем у USB-

накопителя. Не стоит более ста раз перепрошивать микроконтроллер, он может попросту не прошиться. В то же самое время данные на флешку можно записывать более 10 000 раз. Именно поэтому нам потребуется симулятор Proteus, с помощью которого можно будет протестировать работу прошивки и в определенной степени всего устройства, прежде чем начать прошивать живой микроконтроллер.

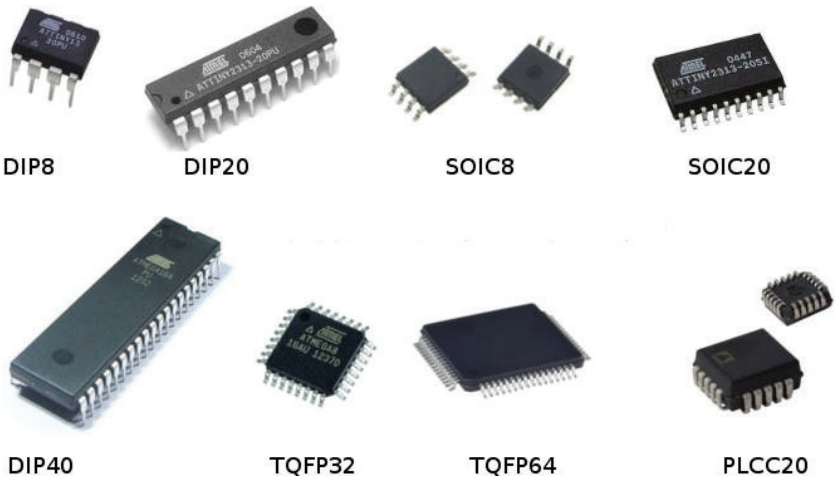

Рис. 1.1. Современные микроконтроллеры

Посмотрим, как логически устроен МК (рис. 1.2).

- Арифметико-логическое устройство (АЛУ) – предназначено для выполнения арифметических и логических операций, на самом деле в совокупности с регистрами общего назначения АЛУ выполняет функции процессора.
- Оперативно-запоминающее устройство (ОЗУ) – предназначено для временного хранения данных при работе микроконтроллера.
- Память программ – выполнена в виде перепрограммируемого постоянного запоминающего устройства и предназначена для записи микропрограммы управления микроконтроллером, так называемая прошивка.
- Память данных применяется в некоторых микроконтроллерах в качестве памяти для хранения всевозможных констант, табличных значений функций и т. д.

Микроконтроллер в своем составе может иметь и другие вспомогательные элементы.

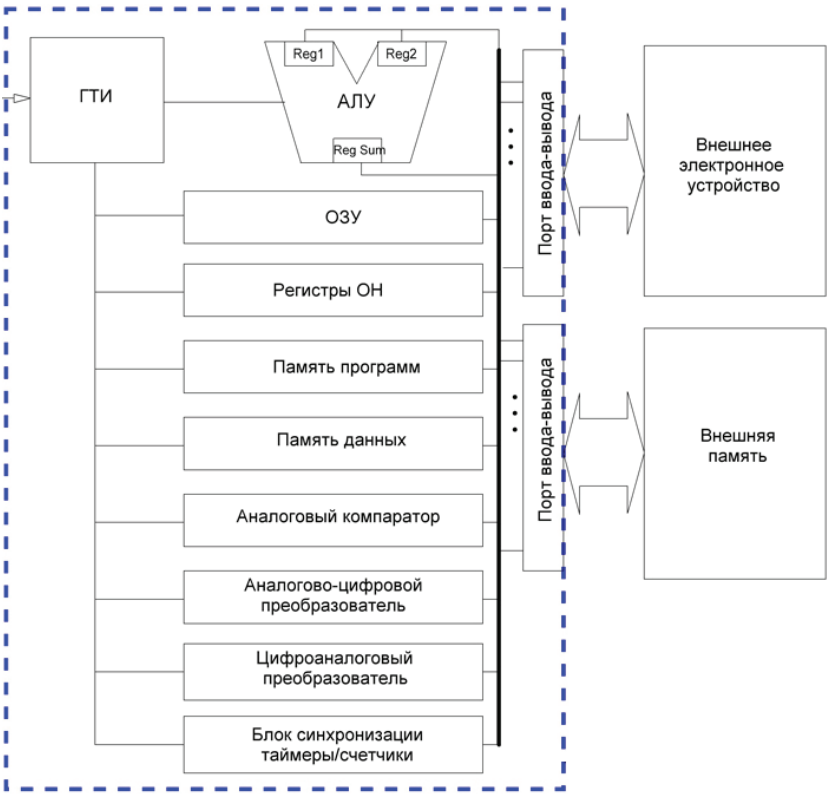

Рис. 1.2. Логическая схема устройства микроконтроллера

- Аналоговый компаратор – предназначен для сравнения двух аналоговых сигналов на его входах.
- Таймеры в микроконтроллерах применяются для осуществления различных задержек и установки различных интервалов времени в работе микроконтроллера.
- Аналогово-цифровой преобразователь (АЦП) необходим для ввода аналогового сигнала в микроконтроллер, и его функция – перевести аналоговый сигнал в цифровой.
- Цифроаналоговый преобразователь (ЦАП) выполняет обратную функцию, то есть сигнал из цифрового вида преобразуется в аналоговый вид.

Программный код, который зашивается в микроконтроллер, отвечает за взаимодействие всех этих элементов в рамках поставленной программистом задачи.

Думаю, что с тем, что такое микроконтроллер, в целом понятно, теперь перейдем к рассмотрению МК от Atmel.

1.1.1. Обзор микроконтроллеров Atmel AVR

Компания Atmel производит несколько типов микроконтроллеров. На их сайте можно найти следующее описание:

32-разрядные микроконтроллеры AVR UC3

Самые эффективные 32-разрядные микроконтроллеры.
Предназначение: системы общего назначения.
Основные параметры:
- флеш-память 16–512 Кб;
- корпуса с 48–144 выводами;
- рабочая частота до 66 МГц;
- производительность 1,5 MIPS/МГц.

Микроконтроллеры AVR XMEGA

8-разрядное решение высокой производительности.
Предназначение: системы общего назначения.
Основные параметры:
- флеш-память 16–384 Кб;
- корпуса с 32–100 выводами;
- рабочая частота до 32 МГц;
- производительность 1,0 MIPS/МГц.

Микроконтроллер megaAVR

Больше функций и периферийных устройств.
Предназначение: системы общего назначения.
Основные параметры:
- флеш-память 4–256 Кб;
- корпуса с 28–100 выводами;
- рабочая частота до 20 МГц;
- производительность 1,0 MIPS/МГц.

8-разрядные микроконтроллеры tinyAVR

Компактность и мощность.
Предназначение: системы общего назначения.
Основные параметры:
- рабочее напряжение 0,7 В;
- флеш-память 0,5–8 Кб;
- корпуса с 6–32 выводами;
- рабочая частота до 20 МГц;
- производительность 1,0 MIPS/МГц.

Управление аккумулятором

Предназначение:

- управление литий-ионными аккумуляторами;
- измерение уровня заряда аккумулятора;
- балансировка элементов аккумулятора;

Основные параметры:

- рабочее напряжение 1,8–25 В;
- флеш-память 8–40 Кб;
- корпуса с 28–48 выводами;
- рабочая частота до 8 МГц;
- производительность 1,0 MIPS/МГц.

Микроконтроллеры AVR для автомобильной электроники

Предназначение: интеллектуальное управление и надежная конструкция, соответствующие жестким требованиям к автомобильной электронике.

Как видно, Atmel выпускает множество различных микроконтроллеров, предназначенных как для общего применения, так и для решения специализированных задач. Для решения задач, описываемых в этой книге, нам вполне подойдут 8-битные МК семейства tinyAVR. Более мощные и, следовательно, более дорогие микроконтроллеры будут избыточны для этого.

1.1.2. ATtiny13 – маленькая, но шустрая

ATtiny – семейство AVR-микроконтроллеров, оптимизированных для приложений, требующих относительно большой производительности (до 1,0 MIPS и способных работать на частотах до 20,0 МГц), энергоэффективности и компактности.

Номенклатура микроконтроллеров ATtiny на сегодняшний день состоит из следующих устройств:

- ATtiny4, ATtiny5, ATtiny9, ATtiny10 – максимум 4 контакта ввода/вывода, рабочее напряжение 1,8–5,5 В, 32 байта SRAM, производительность до 12 MIPS (на частоте 12 МГц), Flash-память для хранения программ (1 Кбайт в ATtiny9/10 и 512 байт в ATtiny4/5), аналого-цифровой преобразователь (в ATtiny9/10).
- ATtiny13 – максимум 6 контактов ввода/вывода, рабочее напряжение 1,8–5,5 В, 64 байта SRAM, 64 байта EEPROM, производительность до 20 MIPS (на частоте 20 МГц), 1 Кбайт Flash-памяти для хранения программ, аналого-цифровой преобразователь (ADC).

- ATtiny24, ATtiny 44, ATtiny 84 – максимум 12 контактов ввода/вывода, рабочее напряжение 1,8–5,5 В, 128/256/512 байт SRAM и 128/256/512 байт EEPROM (соответственно), производительность до 20 MIPS (на частоте 20 МГц), 2/4/8 Кбайт Flash-памяти для хранения программ (соответственно), аналого-цифровой преобразователь (ADC), температурный датчик (на кристалле), универсальный последовательный интерфейс (USI).
- ATtiny25, ATtiny 45, ATtiny 85 – максимум 6 контактов ввода/вывода, рабочее напряжение 1,8–5,5 В, 128/256/512 байт SRAM и 128/256/512 байт EEPROM (соответственно), производительность до 20 MIPS (на частоте 20 МГц), 2/4/8 Кбайт Flash-памяти для хранения программ (соответственно), аналого-цифровой преобразователь (ADC), универсальный последовательный интерфейс (USI).
- ATtiny261, ATtiny 461, ATtiny 8 61 – максимум 16 контактов ввода/вывода, рабочее напряжение 1,8–5,5 В, 128/256/512 байт SRAM и 128/256/512 байт EEPROM (соответственно), производительность до 20 MIPS (на частоте 20 МГц), 2/4/8 Кбайт Flash-памяти для хранения программ (соответственно), аналого-цифровой преобразователь (ADC), универсальный последовательный интерфейс (USI).
- ATtiny48, ATtiny 88 – максимум 24/28 контактов ввода/вывода (в зависимости от корпуса), рабочее напряжение 1,8–5,5 В, 256/512 байт SRAM (соответственно), 64 байта EEPROM, производительность до 12 MIPS (на частоте 12 МГц), 4/8 Кбайт Flash-памяти для хранения программ (соответственно), аналого-цифровой преобразователь (ADC), последовательный внешний интерфейс (SPI).
- ATtiny43U – максимум 16 контактов ввода/вывода, рабочее напряжение 0,7–1,8 В, 256 байт SRAM, 64 байта EEPROM, производительность до 1 MIPS на мегагерц, 4 Кбайт Flash-памяти для хранения программ, аналого-цифровой преобразователь (ADC), температурный датчик (на кристалле), универсальный последовательный интерфейс (USI). Микросхема с низким энергопотреблением, встроенный преобразователь автоматически генерирует стабильное напряжение питания 3 В от низковольтного источника питания (не ниже 0,7 В).

Как видно, ATTiny13 является не самым мощным микроконтроллером в данном ряду. Однако небольшой размер, невысокие требования к энергопотреблению и дешевизна сделали его главным героем этой книги. Кроме того, ATTiny13 оптимально подходит для разработки тех устройств безопасности, о которых шла речь в преды-

дущей главе. Функционал ATtiny43U, к примеру, будет избыточен для большинства из них.

Далее мы немного поговорим о некоторых особенностях работы с данным микроконтроллером и затем перейдем к обсуждению большой темы, связанной с программированием данных МК (рис. 1.3).

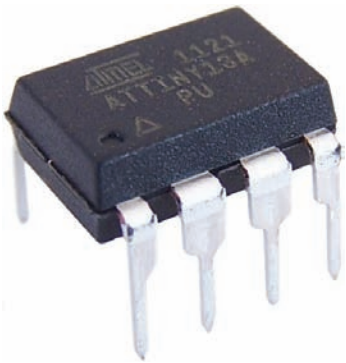

Рис. 1.3. Внешний вид МК ATtiny13

1.1.3. Особенности применения

МК ATtiny13 имеет корпус DIP или SMD. Учитывая небольшой размер самого микроконтроллера, а также наличие у него всего 8 внешних контактов, я бы рекомендовал использовать корпус DIP.

Кроме того, с микроконтроллерами в корпусе DIP очень легко работать благодаря большому шагу между выводами, что очень удобно как при пайке, так и при подключении микроконтроллера к плате через специальный разъем (панелька/кроватка) – для снижения риска повреждения компонента при пайке.

Также использование панелек позволяет при необходимости перепрошивать МК, например в случае модернизации кода программы или при использовании МК в другом устройстве.

В качестве источников питания можно использовать как старые зарядки от телефонов с выходным напряжением 5 В, так и круглые, плоские батарейки типа CR2032, выдающие напряжение 3 В.

В остальном этот микроконтроллер не требует каких-либо дополнительных настроек и компонентов.

1.1.4. Заключение

В этом разделе мы кратко обсудили, что такое микроконтроллер, зачем он нужен. Также поговорили о типах существующих на рынке

решений. Еще мы обсудили все особенности и преимущества ATtiny13. Теперь самое время перейти к описанию средств программного обеспечения для микроконтроллеров.

1.2. Ассемблер Atmel AVR

Для написания прошивок для МК ATtiny существует несколько языков программирования. Автору приходилось сталкиваться с компиляторами для языков высокого уровня C и Basic. Однако в своей книге для написания прошивок я буду использовать ассемблер.

1.2.1. Почему именно ассемблер?

У начинающих радиолюбителей, особенно не слишком хорошо знакомых с программированием, данный язык программирования вызывает страх и трепет, и они предпочитают использовать в своих устройствах код, написанный на языках высокого уровня. Конечно, такая точка зрения вполне имеет право на жизнь. В простых устройствах, где скомпилированная прошивка занимает не более 200–300 байт, использование высокоуровневых языков будет вполне уместно. Однако не стоит забывать, что при компиляции с языка высокого уровня неизбежно теряется часть памяти. Дело в том, что компилятор не всегда использует оптимальные приемы трансляции с языков высокого уровня в машинные коды. В результате один и тот же алгоритм, реализованный на ассемблере и на C, может в откомпилированном виде в первом случае занимать на 5–10 процентов меньше места, чем во втором. Кроме того, программы, написанные на ассемблере, зачастую работают быстрее, чем их аналоги на языках высокого уровня. Это также связано с тем, что компиляторы не оптимально транслируют высокоуровневую программу в машинные коды. Хотя при разработке описываемых в книге устройств вам вряд ли придется столкнуться с проблемами производительности МК.

С другой стороны, сложность ассемблера МК Atmel несколько надуманна. По крайней мере, по сравнению с аналогичными языками программирования в более серьезных процессорах, таких как Intel. Здесь нет такого разнообразия команд и типов регистров, поэтому освоить ассемблер Atmel будет значительно проще, чем, к примеру, программировать 64-битные приложения под Unix.

Так что не все так страшно. Ну а я со своей стороны постараюсь вам помочь, подробно и доходчиво описав основы программирования на данном языке.

1.2.2. Регистры

Для начала работы с ассемблером необходимо усвоить понятие регистра. Регистром является зарезервированная ячейка памяти. В отличие от памяти программ, адресное пространство памяти данных адресуется побайтно 8 разрядам (а не пословно 16 разрядам). Адресация МК Atmel полностью линейная, без какого-то деления на страницы, сегменты или банки, как это принято в некоторых других системах.

МК семейства Tiny (включая Tiny13) памяти данных как таковой не имеют, ограничиваясь лишь регистрами общего назначения (РОН) и регистрами ввода-вывода (РВВ).

Всего в МК 32 регистра общего назначения. В микроконтроллерах AVR все 32 РОН непосредственно доступны АЛУ. Благодаря этому любой РОН с R0 по R31 может использоваться во всех командах и как операнд-источник, и как операнд-приемник. Исключение составляют лишь пять арифметических и логических команд. При этом регистры R26-R31 используются для косвенной адресации. Пары этих 8-разрядных РОН образуют три 16-разрядных регистра X, Y, Z.

У МК есть три регистра ввода-вывода: DDRx, PORTx и PINx. Эти регистры, как следует из названия, предназначены для взаимодействия с портами ввода-вывода. Порты, в свою очередь, могут работать как входы и как выходы. Если порт работает как вход, то, для того чтобы считать значения, необходимо обратиться к регистру PINB или PIND – смотря с какого порта производим считывание. Если порт является выходом, то значения на линиях порта устанавливаются путем записи соответствующего значения в регистр порта PORTB или PORTD.

Самый важный момент работы с портом – это работа с регистром-защелкой, отвечающей за работу линий порта на вход или на выход. Название этого регистра DDRx, где x – буква порта. Для того чтобы сделать ножки выходами, мы должны записать в соответствующие биты «1». Например, мы хотим сделать ножку PB7 порта B входом, а остальные ножки – выходами, тогда для этого необходимо записать в регистр DDRB значение 0b01111111. Приставка 0b означает, что число записано в двоичном виде. При запуске регистры DDRx обнулены, т. е. все ножки являются входами.

Еще одним важным регистром, о котором следовало бы упомянуть, является регистр SREG. Он содержит специальные флаги (по сути, биты), которые могут принимать значения 0 или 1 в зависимости от того, в каком состоянии должен находиться флаг.

Ниже приводится описание всех флагов, входящих в SREG.

Бит 7. Флаг I. Общее разрешение прерываний

Для разрешения прерываний этот флаг должен быть установлен в 1. Если флаг сброшен, то прерывания запрещены независимо от состояния разрядов регистров маскирования отдельных прерываний. Флаг сбрасывается аппаратно после входа в прерывание и восстанавливается командой RETI для разрешения обработки следующих прерываний.

Бит 6. Флаг T. Хранение копируемого бита

Используется в качестве источника или приемника команд копирования битов BLD (Bit Load) и BST (Bit Store).

Бит 5. Флаг H. Флаг половинного переноса

Устанавливается в 1, если произошел перенос из младшей половины байта (т. е. из третьего разряда в четвертый) или заем из старшей половины байта при выполнении некоторых арифметических операций.

Бит 4. Флаг S. Флаг знак

Равен результату операции «Исключающее ИЛИ» (XOR) между флагами N и V. Соответственно, этот флаг устанавливается в 1, если результат выполнения арифметической операции меньше нуля.

Бит 3. Флаг V. Флаг переполнения дополнительного кода

Устанавливается в 1 при переполнении разрядной сетки знакового результата. Используется при работе со знаковыми числами (представленными в дополнительном коде).

Бит 2. Флаг N. Флаг отрицательного значения

Устанавливается в 1, если старший (седьмой) разряд результата операции равен единице. В противном случае флаг равен 0.

Бит 1. Флаг Z. Флаг нуля

Устанавливается в 1, если результат выполнения операции равен нулю.

Бит 0. Флаг C. Флаг переноса

Устанавливается в 1, если в результате выполнения операции произошел выход за границы байта.

Некоторые из этих флагов мы будем активно использовать в дальнейшем.