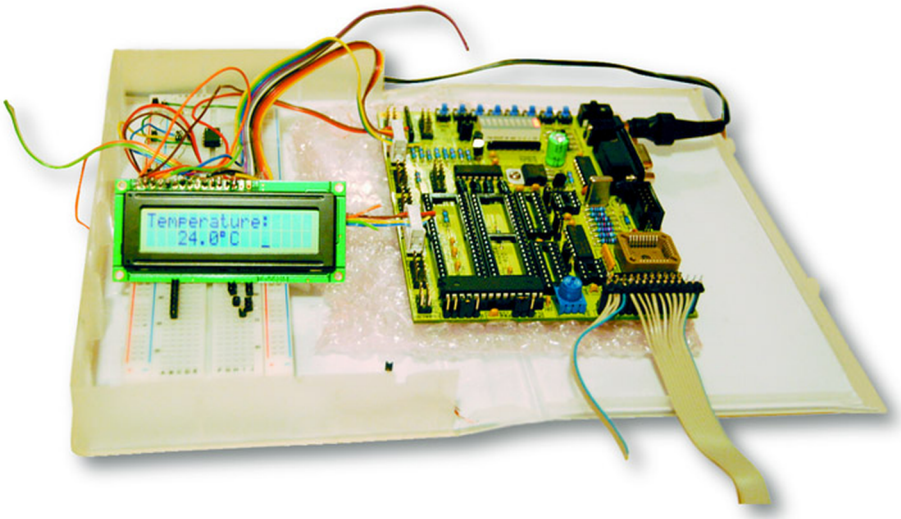
Электроника

Юрий Ревич



практическое программирование микроконтроллеров Atmel AVR на языке ассемблера

2-е издание

Юрий Ревич

практическое программирование микроконтроллеров Atmel AVR на языке ассемблера

2-е издание

Санкт-Петербург
«БХВ-Петербург»
2011

УДК 681.3.068
ББК 32.973.26-018.1
Р32

Ревич Ю. В.

Р32 Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера. — 2-е изд., испр. — СПб.: БХВ-Петербург, 2011. — 352 с.: ил. — (Электроника)

ISBN 978-5-9775-0657-1

Изложены принципы функционирования, особенности архитектуры и приемы программирования микроконтроллеров Atmel AVR. Приведены готовые рецепты для программирования основных функций современной микроэлектронной аппаратуры: от реакции на нажатие кнопки или построения динамической индикации до сложных протоколов записи данных во внешнюю память или особенностей подключения часов реального времени. Особое внимание уделяется обмену данными микроэлектронных устройств с персональным компьютером, приводятся примеры программ. В книге учтены особенности современных моделей AVR и сопутствующих микросхем последних лет выпуска. Приложения содержат основные параметры микроконтроллеров AVR, перечень команд и тексты программ для них, а также список используемых терминов и аббревиатур.

Для учащихся, инженерно-технических работников и радиолюбителей

УДК 681.3.068
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Юрий Рожко*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 01.10.10.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 28,38.
Тираж 1500 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0657-1

© Ревич Ю. В., 2010
© Оформление, издательство "БХВ-Петербург", 2010

Оглавление

Микроконтроллеры, их возникновение и применение 7
Предыстория микроконтроллеров 8
Электроника в греческом стиле 10
Почему AVR? 12
Что дальше? 14

ЧАСТЬ I. ОБЩИЕ ПРИНЦИПЫ УСТРОЙСТВА И ФУНКЦИОНИРОВАНИЯ ATMEL AVR 17

Глава 1. Обзор микроконтроллеров Atmel AVR 19
Семейства AVR 21
Особенности практического использования МК AVR 23
О потреблении 23
Некоторые особенности применения AVR в схемах 25

Глава 2. Общее устройство, организация памяти, тактирование, сброс 27
Память программ 27
Память данных (ОЗУ, SRAM) 29
Энергонезависимая память данных (EEPROM) 31
Способы тактирования 32
Сброс 34

Глава 3. Знакомство с периферийными устройствами 37
Порты ввода-вывода 38
Таймеры-счетчики 39
Аналогово-цифровой преобразователь 41
Последовательные порты 42
UART 43
Интерфейс SPI 46
Интерфейс TWI (I^2C) 50
Универсальный последовательный интерфейс USI 50

Глава 4. Прерывания и режимы энергосбережения 53
Прерывания 53
Разновидности прерываний 57
Режимы энергосбережения 58

ЧАСТЬ II. ПРОГРАММИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ ATMEL AVR 61

Глава 5. Общие принципы программирования МК семейства AVR 63
Ассемблер или С? 63
Способы и средства программирования AVR 67
Редактор кода 67
Об AVR Studio 68
Обустройство ассемблера 70
Программаторы 71
О hex-файлах 75
Команды, инструкции и нотация AVR-ассемблера 78
Числа и выражения 79
Директивы и функции 80
Общая структура AVR-программы 84
Обработка прерываний 85
RESET 89
Простейшая программа 90
Задержка 92
Программа счетчика 94
Использование прерываний 96
Задержка по таймеру 97
Программа счетчика с использованием прерываний 98
О конфигурационных битах 101

Глава 6. Система команд AVR 105
Команды передачи управления и регистр *SREG* 105
Команды проверки-пропуска 111
Команды логических операций 113
Команды сдвига и операции с битами 114
Команды арифметических операций 116
Команды пересылки данных 118
Команды управления системой 122
Выполнение типовых процедур на ассемблере 123
О стеке, локальных и глобальных переменных 125

Глава 7. Арифметические операции 127
Стандартные арифметические операции 128
Умножение многоразрядных чисел 129
Деление многоразрядных чисел 131
Операции с дробными числами 134
Генератор случайных чисел 136
Операции с числами в формате BCD 138
Отрицательные числа в МК 143

Глава 8. Программирование таймеров 147
8- и 16-разрядные таймеры 147
Формирование заданного значения частоты 149
Отсчет времени 153
Точная коррекция времени 158

Частотомер и периодомер ... 160
Частотомер ... 160
Периодомер ... 164
Управление динамической индикацией ... 167
LED-индикаторы и их подключение ... 168
Программирование динамической индикации ... 171
Таймеры в режиме PWM ... 174

Глава 9. Использование EEPROM ... 179
Еще раз о сохранности данных в EEPROM ... 179
Запись и чтение EEPROM ... 181
Хранение констант в EEPROM ... 183

Глава 10. Аналоговый компаратор и АЦП ... 187
Аналого-цифровые операции и их погрешности ... 187
Работа с аналоговым компаратором ... 190
Интегрирующий АЦП на компараторе ... 193
Принцип работы и расчетные формулы ... 194
Программа интегрирующего АЦП ... 198
Встроенный АЦП ... 201
Пример использования АЦП ... 204
Программа ... 206

Глава 11. Программирование SPI ... 215
Основные операции через SPI ... 215
Аппаратный вариант ... 216
Программный вариант ... 218
О разновидностях энергонезависимой памяти ... 219
Запись и чтение flash-памяти через SPI ... 221
Программа обмена с памятью 45DB011B по SPI ... 224
Запись и чтение flash-карт ... 225
Подключение карт MMC ... 225
Подача команд и инициализация MMC ... 228
Запись и чтение MMC ... 232

Глава 12. Интерфейс TWI (I^2C) и его практическое использование ... 237
Базовый протокол I^2C ... 237
Программная эмуляция протокола I^2C ... 240
Запись данных во внешнюю энергонезависимую память ... 241
Режимы обмена с памятью AT24 ... 241
Программа ... 243
Часы с интерфейсом I^2C ... 247
Запись данных ... 255
Чтение данных ... 259

Глава 13. Программирование UART/USART ... 261
Инициализация UART ... 262
Передача и прием данных ... 263
Пример установки часов DS1307 с помощью UART ... 266

Приемы защиты от сбоев при коммуникации ... 271
Проверка на четность ... 271
Как организовать корректный обмен ... 273
Дополнительные возможности USART ... 274
Реализация интерфейсов RS-232 и RS-485 ... 276
Преобразователи уровня для RS-232 ... 280
RS-485 ... 283

Глава 14. Режимы энергосбережения и сторожевой таймер ... 285
Программирование режима энергосбережения ... 286
Пример прибора с батарейным питанием ... 287
Доработка программы ... 289
Использование сторожевого таймера ... 293

ПРИЛОЖЕНИЯ ... 299

Приложение 1. Основные параметры микроконтроллеров Atmel AVR ... 301

Приложение 2. Команды Atmel AVR ... 309
Арифметические и логические команды ... 310
Команды операций с битами ... 311
Команды сравнения ... 312
Команды передачи управления ... 313
Команды безусловного перехода и вызова подпрограмм ... 313
Команды проверки-пропуска и команды условного перехода ... 314
Команды переноса данных ... 315
Команды управления системой ... 316

Приложение 3. Тексты программ ... 317
Демонстрационная программа обмена данными с flash-памятью 45DB011B по интерфейсу SPI ... 317
Процедуры обмена по интерфейсу I^2C ... 321

Приложение 4. Обмен данными с персональным компьютером и отладка программ через UART ... 329
Работа с COM-портом в Delphi ... 329
Установка линии RTS в DOS и Windows ... 335
Программа COM2000 ... 337
Отладка программ с помощью эмулятора терминала ... 339

Приложение 5. Словарь часто встречающихся аббревиатур и терминов ... 341

Литература ... 347

Предметный указатель ... 349

ВВЕДЕНИЕ

Микроконтроллеры, их возникновение и применение

Говорят, что в 1960-е годы, наблюдая за участниками студенческих демонстраций протеста, Гордон Мур заметил: "Истинные революционеры — это мы". Ученик и сотрудник одного из изобретателей транзистора У. Шокли, в числе прочего считающегося основателем знаменитой Кремниевой долины, в свою очередь основатель и лидер компаний, которым суждено было сыграть ведущую роль в развитии микроэлектроники, Мур знал, что говорил. Парадоксальным образом именно изобретениям Мура и его сотрудников было суждено стать основой того мира, в котором впоследствии сконцентрировалась деятельность "бунтующей молодежи" 1960-х. Современные хакеры (не компьютерные хулиганы из газет, а настоящие увлеченные своим делом компьютерщики) — прямые идеологические наследники сорбоннских студентов и американских демонстрантов, сменившие девиз "Make love not war"[1] на "Не пишите лозунги — пишите код". Неслучайно многие известные деятели электронно-компьютерной индустрии, авторы изобретений, сформировавших лицо современного мира, — выходцы из среды, близкой той самой "бунтующей молодежи".

Наша история о микроконтроллерах началась с того, что в 1957 г. Гордон Мур совместно с Робертом Нойсом, ставшим впоследствии одним из изобретателей микросхемы, и еще шестью сотрудниками Shockley Semiconductor Labs (Шокли назвал их "предательской восьмеркой"), основал компанию Fairchild Semiconductor. Ей мы обязаны не только развитием полупроводникового рынка и внедрением микросхем в инженерную практику, но и тем, что она стала своеобразной кузницей кадров и генератором идей для молодой отрасли.

Вот только некоторые из исторических фактов. Сам Мур с Нойсом в конце 1960-х создали фирму Integrated Electronics, которая под сокращенным названием Intel сейчас знакома каждому школьнику. Джереми Сандерс, основатель другой известнейшей компании — AMD, также вышел из Fairchild, где отличился открытием современной экономической модели производства и продаж полупроводниковых компонентов, в которой себестоимость изделия стремится к нулю по мере повыше-

---

[1] "Занимайтесь любовью, а не войной" — лозунг хиппи 1960-х, протестующих против войны во Вьетнаме.

ния объема партии. Чарли Спорк, один из ключевых менеджеров Fairchild, в 1967 г. стал директором National Semiconductor, которой впоследствии руководил четверть века. Половина "предательской восьмерки" — Джин Хоерни, Евгений Клайнер, Джей Ласт и Шелдон Робертс — в 1961 г. основала компанию Amelco, из которой впоследствии выросли всем известные теперь Intersil, Maxim и Ixys. Сотруднику Fairchild Роберту Видлару мы обязаны изобретением операционных усилителей — разновидности микросхем, и по сей день уступающей по популярности разве что микропроцессорам. Мало того, с историей Fairchild связано возникновение известной венчурной (т. е. "рисковой") модели финансирования, сыгравшей определяющую роль в развитии всех отраслей, связанных с электроникой, компьютерами и телекоммуникациями. Недаром Fairchild нередко называют "праматерью всей электроники".

Предыстория микроконтроллеров

Из всего этого урагана событий для нашего повествования важно то, что в числе прочих инноваций сотрудники Fairchild первыми стали продвигать полупроводниковую память. Сейчас, в век CD и DVD, жестких дисков и flash-карточек, нам трудно представить себе, что в начале 1960-х годов программы для компьютеров хранились в основном на картонных листочках (перфокартах), конструкторы ломали голову над дорогущими модулями ОЗУ на ртутных линиях задержки, осциллографических трубках и ферритовых колечках, где каждый бит "прошивался" вручную. Самое компактное в те годы электронное устройство для хранения данных на магнитных дисках под названием RAMAC 305 емкостью 5 Мбайт было размером с промышленный холодильник и сдавалось в аренду за 5 тыс. долларов в месяц.

Единственным "лучом света" в темном царстве этих монстров стало изобретение сотрудника корпорации American Bosch Arma Йен Чоу, который в 1956 г. получил патент на устройство, известное теперь как "однократно программируемое ROM" (OTP ROM[1]). В этом патенте, между прочим, впервые был употреблен термин "прожиг" (burn) — микромодуль состоял из матрицы с плавкими перемычками, которые при программировании пережигались подачей на них большого напряжения. OTP ROM долгое время оставались единственными устройствами для компактного хранения данных, и не потеряли своего значения до самого последнего времени — не меньше четверти микроконтроллеров в мире, особенно из тех, что попроще, до сих пор выпускается именно с такой однократно программируемой встроенной памятью, ввиду крайней ее дешевизны. И лишь в последние годы "прожигаемая" память стала постепенно вытесняться более удобной flash-памятью, когда последняя подешевела настолько, что смысл в использовании OTP ROM почти пропал.

Но вернемся в 1960-е. Компактная полупроводниковая память была нужна абсолютно всем — от военных и NASA до изготовителей бытовых приборов. Сначала Fairchild предложила то, что сегодня называется DRAM, в частности, на таких микросхемах (32 768 чипов емкостью 256 бит каждый) была построена память знаме-

[1] Расшифровку некоторых аббревиатур см. в *приложении 5*.

нитого суперкомпьютера ILLIAC-IV, конкурента отечественной БЭСМ-6. Почуяв, откуда дует ветер, в 1968 г. Мур с Нойсом оставили Fairchild и основали Intel, как специализированную компанию по разработке и производству памяти. Они еще не ведали, что самым популярным детищем Intel станет вовсе не память, а небольшой приборчик (рис. В1), названный микропроцессором, разработка которого первоначально затевалась как вспомогательный этап в проектировании обычного калькулятора.

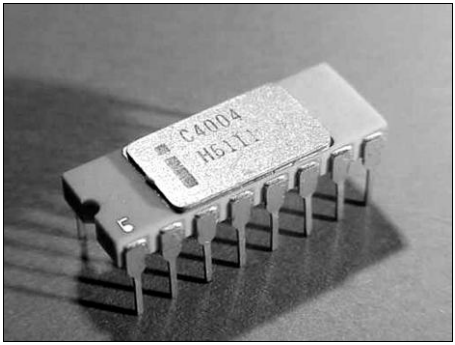


Рис. В1. Микропроцессор Intel 4004

Изобретение микропроцессора

В 1969 г. в Intel появились несколько человек из Busicom — молодой японской компании, занимающейся производством калькуляторов. Им требовался набор из 12 интегральных схем в качестве основного элемента нового дешевого настольного калькулятора. Проект был разработан Масатоши Шима, который и представлял японскую сторону. Тед Хофф (Marcian E. Ted Hoff, р. 1937 г.), руководитель отдела, занимавшегося разработкой применений для продукции Intel, ознакомившись с проектом, понял, что вместо того, чтобы создавать калькулятор с некоторыми возможностями программирования, можно сделать наоборот, компьютер, программируемый для работы в качестве калькулятора. Развивая идею, в течение осени 1969 г. Хофф определился с архитектурой будущего микропроцессора. Весной в отдел Хоффа пришел (все из той же уже известной нам Fairchild) новый сотрудник Фредерик Фэггин (Federico Faggin), который и придумал название для всей системы: "семейство 4000". Семейство состояло из четырех 16-выводных микросхем: 4001 содержал ROM на 2 Кбайта; 4002 — RAM с 4-битовым выходным портом для загрузки программ; 4003 представлял собой 10-битовый расширитель ввода-вывода с последовательным вводом и параллельным выводом для связи с клавиатурой, индикатором и другими внешними устройствами; наконец 4004 был 4-битовым ЦПУ (центральным процессорным устройством). Это ЦПУ содержало 2300 транзисторов и работало на тактовой частоте 108 кГц. 15 ноября 1971 г. было объявлено о создании первого микропроцессора. Busicom приобрела разработку, заплатив Intel $60 000. Но в Intel решили возвратить Busicom эти деньги, чтобы вернуть себе права на микропроцессор.

i4004 обладал вычислительной мощностью, сравнимой с первым электронным компьютером ENIAC (1946). Свое первое практическое применение 4004-й нашел в системах управления дорожными светофорами и анализаторах крови. Этот микропроцессор был использован в бортовой аппаратуре межпланетного зонда Pioneer-10, который поставил рекорд долгожительства среди подобных аппаратов: он был запущен в 1972 г., а к сентябрю 2001 г. Pioneer-10 удалился от Земли на 11,78 млрд км и все еще работал и, вполне вероятно, работает по сей день, хотя в феврале 2003 г. NASA официально с ним попрощалось.

Так началось победное шествие микропроцессоров, которые позднее разделились на несколько разновидностей, в основном относящихся к двум главным группам: собственно микропроцессорам (МП) и микроконтроллерам (МК). Первые предназначены для использования в составе вычислительных систем, самые распространенные из которых — персональные компьютеры (ПК), поэтому их еще часто называют "процессорами для ПК" (хотя к этой группе обычно относят также и производительные МП для серверов и некоторые другие). МК отличаются от МП тем, что они в первую очередь предназначены для управления различными системами, поэтому при относительно более слабом вычислительном ядре они включают в себя много дополнительных узлов. То, что для обычного МП предполагается размещать во внешних чипсетах или дополнительных модулях (память, порты ввода-вывода, таймеры, контроллеры прерываний, узлы для обработки аналоговых сигналов и пр.), в МК располагается прямо на кристалле, отчего их когда-то было модно называть "микро-ЭВМ".

И действительно, в простейшем случае для построения полностью функционирующего компьютера достаточно единственной микросхемы МК с подсоединенными к ней устройствами ввода-вывода. Современные модели рядовых однокристальных МК превышают вычислительные возможности IBM PC AT на 286-м процессоре образца второй половины 1980-х. Есть области, где границу между МП и МК провести трудно: таковы, например, процессоры для мобильных устройств, от телефонов и карманных компьютеров до цифровых камер, в которых процессорный узел должен обладать развитыми вычислительными функциями и управлять многочисленными внешними компонентами.

Электроника в греческом стиле

В 1962 г. в Калифорнии появилась семья Перлегос, греческих эмигрантов, уроженцев города Триполис. Родители занялись, как и на родине, виноградарством, а сыновья, Джордж и Гюст Перлегос, выбрали модную специальность инженера-электронщика: оба окончили вначале университет Сан-Хозе, а затем Стэнфордский университет. В 1974 г. в возрасте 24 лет младший из братьев Джордж Перлегос начал работать в компании Intel, где попал на одно из самых передовых направлений: разработку электрически стираемой памяти для замены "прожигаемой" OTP ROM. Еще до Перлегоса, почти одновременно с изобретением микропроцессора в 1971 г., сотрудник Intel Дон Фрохман изобрел "плавающий" затвор и создал первую УФ-стираемую EPROM объемом 2К (256×8).

Заметки на полях

В "обычной" жизни употреблять сокращение для единиц информации из одной буквы "К" (так же, как и "М") не рекомендуется: очень трудно иногда понять, идет ли речь о килобитах, килобайтах, "килословах" или вообще килобитах в секунду. Тем не менее такие сокращения часто встречаются, в том числе и в техдокументации, и нам придется иногда следовать этому примеру. Для определенности примем следующие правила: одиночная прописная буква "К" означает двоичные килобиты (1024 бита), "М" — двоичные мегабиты (1024 кбита). Хотя в литературе часто еще принято килобайты сокращать, как "КБ", а килобиты, как "Кб", мы постараемся избежать этой путаницы, и во

всех остальных случаях писать полностью: кбайт и Мбайт, кбит/с, Мбайт/с. Исключение составит обозначение объема памяти программ микроконтроллеров, если он измеряется в двухбайтовых словах: например, 4К слов будет обозначать 4096 ячеек-слов (8 кбайт).

Джордж Перлегос активно включился в этот процесс и сначала при его участии, а затем и под его непосредственным руководством были созданы две технологии, ставшие точкой роста для всей отрасли по производству flash-памяти — одного из главных столпов современной "цифровой революции". Это было сначала изобретение чипа 2716 — 16К (2048×8) EPROM с одним напряжением питания +5 В, а затем 2816 — первой EEPROM, электрически стираемого ПЗУ, ставшего прообразом flash-памяти.

В 1981 г. Перлегос покидает Intel и с несколькими сотрудниками (в числе которых был Гордон Кэмпбелл, будущий создатель другой известной фирмы Chips & Technologies) создает компанию Seeq. Это было время спада в электронной промышленности и через три года компанию пришлось покинуть в связи с претензиями инвесторов. Не доверяя им больше, Джордж с братом Гюстом и еще несколькими сотрудниками Seeq в 1984 г. создает в складчину на личные средства компанию, полное название которой звучит как Advanced Technology MEmory and Logic или сокращенно — Atmel.

Сначала продукцией Atmel были микросхемы энергонезависимой памяти всех разновидностей — как OTP EPROM и EEPROM с последовательным и параллельным доступом, так и Flash. В 1985 г. Atmel выпустила первую в мире EEPROM по доминирующей ныне КМОП-технологии, а в 1989 г. — первую flash-память с питанием от одного напряжения +5 В. В конце 1980-х Intel вознамерилась наказать ряд компаний-производителей EPROM, в том числе и Atmel, якобы за нарушение патентов, но, в конце концов, удалось договориться об обмене лицензиями. Причем в конечном итоге Atmel перепала лицензия на производство классического микроконтроллера 8051, от поддержки которого Intel уже в то время постепенно отходила, сосредоточившись на процессорах для ПК.

Подробности

Напомним, что EEPROM отличается от flash-памяти тем, что первая допускает раздельный доступ к любой произвольной ячейке, а вторая — лишь к целым блокам. Поэтому EEPROM меньше по объему (характерный объем специализированных микросхем EEPROM — от единиц килобит до единиц мегабит) и дороже, в настоящее время ее используют в основном для хранения данных, в том числе в составе микроконтроллеров. Flash-память проще и дешевле, и к тому же дает значительный выигрыш в скорости при больших объемах информации, особенно при потоковом чтении/записи, характерном для медиаустройств (вроде цифровых камер или MP3-плееров). В составе микроконтроллеров flash-память служит для хранения программ. Некоторые подробности о различных типах памяти и их функционировании см. в *главе 11*.

Так Atmel оказалась "втянута" в число производителей микроконтроллеров, в котором очень быстро оказалась на первых позициях: в 1993 г. началось производство первых в отрасли МК AT89C51 со встроенной flash-памятью программ. Это означало начало переворота во всей инженерной практике, потому что существовавшие ранее МК обладали либо однократно программируемой OTP-памятью, либо

УФ-стираемой, которая значительно дороже в производстве и работа с ней приводит к большим потерям времени разработчиков. Число циклов перезаписи для УФ ППЗУ не превышает нескольких десятков, а прямой дневной свет, попавший на такой кристалл, может привести к стиранию информации. Поэтому даже мелкосерийные устройства приходилось изготавливать преимущественно с использованием OTP ROM, что значительно рискованнее: изменить в случае даже малейшей ошибки записанную программу уже было невозможно. Появление flash-памяти изменило весь "ландшафт" в этой области: именно в результате ее внедрения стали возможными такие вещи, как программное обновление BIOS компьютера или "перешивка" управляющих программ для бытовых электронных устройств.

В 1995 г. два студента Норвежского университета науки и технологий в г. Тронхейме, Альф Боген и Вегард Воллен, выдвинули идею 8-разрядного RISC-ядра, которую предложили руководству Atmel. Имена разработчиков вошли в название архитектуры AVR: Alf + Vergard + RISC. Идея настолько понравилась, что в 1996 г. был основан исследовательский центр Atmel в Тронхейме и уже в конце того же года выпущен первый опытный микроконтроллер новой серии AVR под названием AT90S1200. Во второй половине 1997 г. корпорация Atmel приступила к серийному производству семейства AVR.

Почему AVR?

У AVR-контроллеров "с рождения" есть две особенности, которые отличают это семейство от остальных МК. Во-первых, система команд и архитектура ядра AVR разрабатывались совместно с фирмой-разработчиком компиляторов с языков программирования высокого уровня IAR Systems. В результате появилась возможность писать AVR-программы на языке С без большой потери в производительности по сравнению с программами, написанными на языке ассемблера. Подробнее этот вопрос мы обсудим в *главе 5*.

Во-вторых, одним из существенных преимуществ AVR стало применение конвейера. В результате для AVR не существует понятия машинного цикла: большинство команд выполняется за один такт. Для сравнения отметим, что пользующиеся большой популярностью МК семейства PIC выполняют команду за 4 такта, а классические 8051 — вообще за 12 тактов (хотя есть и современные модели *x*51 с машинным циклом в один такт).

Правда, при этом пришлось немного пожертвовать простотой системы команд, особенно заметной в сравнении с *x*51, где, например, любые операции пересылки данных внутри контроллера, независимо от способа адресации, выполняются единственной командой `mov` в различных вариантах, в то время как в AVR почти для каждого способа своя команда, к тому же иногда с ограниченной областью действия. Есть некоторые сложности и в области операций с битами. Тем не менее это не приводит к заметным трудностям при изучении AVR-ассемблера: наоборот, тексты программ получаются короче и больше напоминают программу на языке высокого уровня. Следует также учесть, что из общего числа команд от 90 до 130, в зависимости от модели, только 50–60 уникальных, остальные взаимозаменяемые.

И, наконец, этот недостаток полностью нивелируется при использовании языка С, фактически уравнивающего разные архитектуры с точки зрения особенностей программирования.

Огромное преимущество AVR-архитектуры — наличие 32 оперативных регистров, не совсем равноправных, но позволяющих в ряде случаев вообще не обращаться к оперативной памяти и не использовать стек (что в принципе невозможно в том же семействе *x*51), более того, в младших моделях AVR стек вообще недоступен для программиста. Потому структура ассемблерных программ для AVR стала подозрительно напоминать программы на языке высокого уровня, где операторы работают не с ячейками памяти и регистрами, а с абстрактными переменными и константами.

Еще одна особенность AVR со схемотехнической точки зрения — все выводы в них могут пребывать в трех состояниях (вход — отключено — выход) и электрически представляют собой КМОП-структуры (т. е. имеет место симметрия выходных сигналов и высокое сопротивление для входных). В общем случае это значительно удобнее портов *x*51 (двустабильных и TTL-совместимых) и предполагает лучшую помехозащищенность (по крайней мере, от помех по шине "земли").

Суммировав мнения из различных источников и опираясь на собственный опыт, автор пришел примерно к такому подразделению областей применения трех самых распространенных семейств контроллеров.

- Контроллеры классической архитектуры *x*51 (первые микросхемы семейства 8051 были выпущены еще в начале 1980-х) лучше всего подходят для общего изучения предмета. Отметим, что кроме Atmel, *x*51-совместимые изделия выпускают еще порядка десятка фирм, включая такие гиганты, как Philips и Siemens, есть и отечественные аналоги (серии 1816, 1830 и др.), что делает эту архитектуру наиболее универсальной.
- Семейство AVR рекомендуется для начинающих электронщиков-практиков, в силу простоты и универсальности устройства, преемственности структуры для различных типов контроллеров, простоты схемотехники и программирования (в данном случае под "программированием" понимается процесс записи программ в микросхему).
- PIC фирмы Microchip идеально подходят для проектирования несложных устройств, особенно предназначенных для тиражирования.

Эта классификация во многом субъективна, и автор не будет оспаривать другие точки зрения: различные семейства МК постепенно сближаются по параметрам, становятся полностью взаимозаменяемыми и, как и во всей современной электронике, выбор того или иного семейства часто носит характер "религиозного".

К тому же три упомянутых семейства МК — лишь наиболее распространенные среди универсальных контроллеров, но далеко не самые массовые вообще. Общее количество существующих семейств микроконтроллеров оценивается приблизительно в 100 с лишним, причем ежегодно появляются все новые и новые. Каждое из этих семейств может включать десятки разных моделей. При этом первое место среди производителей 8-разрядных МК традиционно принадлежит фирме Motorola, в основном за счет контроллеров для мобильных устройств. Компания Microchip со

своим семейством PIC занимает третье место, а Atmel — лишь шестое. При этом, кроме 8-разрядных МК AVR, Atmel выпускает еще несколько разновидностей МК, к которым относятся не только упомянутые наследники 8051, но и ARM-процессоры и специализированные МК для различных применений. Тем не менее эта формальная статистика еще ни о чем не говорит — так, среди МК со встроенной flash-памятью Atmel принадлежит уже треть мирового рынка.

Еще в 2002–2003 годах в мире выпускалось ежегодно 3,2 млрд штук микроконтроллеров. Отметим, что объем выпуска процессоров для ПК можно оценить в 200 млн единиц в год, т. е. он составляет всего-навсего около 6% рынка (в финансовом исчислении, правда, соотношение иное, ведь типичная цена рядового МК составляет 2–5 долларов, а процессора для ПК — как минимум на порядок выше, а иногда достигает и сотен долларов). Потому не будет преувеличением утверждать, что специальность электронщика-программиста, специализирующегося на микроконтроллерах, не менее важна и дефицитна, чем компьютерного программиста-системщика или создателя пользовательских приложений.

Что дальше?

Эта книга адресована читателю, который хочет изучить структуру и схемотехнические особенности МК AVR и научиться грамотно использовать их основные возможности. Поэтому автор ограничивается языком ассемблера (подробнее вопрос выбора среды программирования мы рассмотрим в *главе 5*). Упор в книге делается на то, чтобы дать читателю практические советы, описать готовые алгоритмы для типовых задач, возникающих перед разработчиками при реализации тех или иных функций МК. Автор вместе с читателями подробно разбирает ряд вопросов, которые обычно выходят за рамки пособий по программированию МК: работу с последовательными интерфейсами, арифметические операции, сопряжение с ПК, практическую реализацию режимов энергосбережения.

Вместе с тем автор не ставил задачу разобрать подробно абсолютно все возможности МК AVR: для этого не хватило бы и нескольких томов. Мы вынуждены обойти такие вопросы, как отладка и программирование по интерфейсу JTAG, перспективы, которые открывает самопрограммирование контроллеров, лишь вскользь коснемся интереснейшей задачи синтеза звука и других применений PWM-режимов таймеров.

В этой книге большинство примеров ориентировано на применение младших (с объемом памяти 8 Мбайт) моделей подсемейства Mega, т. к. именно они наиболее универсальны и пригодны для широкого круга задач без излишнего усложнения схемы. В более простых случаях автор ориентировался на наиболее универсальную из младших моделей ATtiny2313 (о ее совместимости с "классической" версией AT90S2313 см. *разд. "Программа счетчика с использованием прерываний" главы 5*). Большинство приведенных примеров могут быть практически без переделок адаптированы к другим моделям AVR, обладающим соответствующей конфигурацией.

Книга не заменяет фирменные справочники по структуре и системе команд конкретных моделей AVR — их число постоянно растет, некоторые снимаются с производства, другие приходят на их место. Поэтому автор старался давать максимально обобщенные примеры, которые, по возможности, пригодны для большинства существующих моделей. Необходимое дополнение к этой книге — справочник Евстифеева [1, 2], где на русском языке собраны фирменные технические описания для большинства моделей AVR и, в частности, приведена полная таблица команд AVR-ассемблера с подробным формальным описанием каждой из них. Читатели, хорошо владеющие техническим английским, смогут обойтись англоязычными описаниями (т. н. datasheets) конкретных моделей, которые можно скачать с сайта atmel.com. Знакомство с этими материалами так или иначе потребуется, т. к. в построении отдельных моделей AVR слишком много нюансов, делающих их в некоторых частных случаях невзаимозаменяемыми, и все алгоритмы применительно к конкретной ситуации следует проверять. Рекомендации по применению (application notes) Atmel также полезны для изучения, но они, к сожалению, довольно отрывочны, не охватывают всего круга задач и иногда содержат ошибки. Примеры законченных устройств читатель может найти в книге [8]. Там же приведены элементарные сведения по логическим элементам, системам счисления и другие азы микроэлектроники для тех, кто в этом не ориентируется.

Тем, кто уже знаком с семейством AVR, разумеется, можно читать настоящую книгу выборочно, пользуясь ею, как справочником. Всем остальным автор советует хотя бы один раз изучить ее подряд, глава за главой, иначе осознанно применить разрозненные сведения из отдельных глав может и не получиться.

Схемы, рисунки и фотографии выполнены автором.

Пожелания, вопросы и указания на неточности можно направлять по адресу: **revich@lib.ru**.

ЧАСТЬ I

Общие принципы устройства и функционирования Atmel AVR

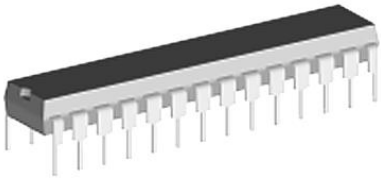
Глава 1. Обзор микроконтроллеров Atmel AVR

Глава 2. Общее устройство, организация памяти, тактирование, сброс

Глава 3. Знакомство с периферийными устройствами

Глава 4. Прерывания и режимы энергосбережения

ГЛАВА 1



Обзор микроконтроллеров Atmel AVR

Atmel AVR представляет собой семейство универсальных 8-разрядных микроконтроллеров на основе общего ядра с различными встроенными периферийными устройствами. Возможности МК AVR позволяют решить множество типовых задач, возникающих перед разработчиками радиоэлектронной аппаратуры.

Особенности микроконтроллеров Atmel AVR.

- **Производительность порядка 1 MIPS/МГц.** MIPS (Millions of Instructions Per Second, миллион команд в секунду) — одна из самых старых и во многом формальная характеристика производительности процессоров, т. к. наборы команд для различных процессоров различаются, и, соответственно, одно и то же число инструкций на различных системах даст разную полезную работу. Тем не менее для простых 8-разрядных вычислительных систем, не содержащих команд, оперирующих с большими числами, числами с плавающей точкой и массивами данных, это неплохой показатель для сравнения их производительности. Вычислительное ядро AVR на ряде задач по производительности превосходит 16-разрядный процессор 80286.
- **Усовершенствованная RISC-архитектура.** Концепция RISC (Reduced Instruction Set Computing, вычисления с сокращенным набором команд) предполагает наличие набора команд, состоящего из минимума компактных и быстро выполняющихся инструкций; при этом такие более громоздкие операции, как вычисления с плавающей точкой или арифметические действия с многоразрядными числами, предполагается реализовать на уровне подпрограмм. Концепция RISC упрощает устройство ядра (в типовом ядре AVR содержится лишь 32 тыс. транзисторов, в отличие от десятков миллионов в процессорах для ПК) и ускоряет его работу: типовая инструкция выполняется за один такт (кроме команд ветвления программы, обращения к памяти и некоторых других, оперирующих с данными большой длины). В AVR имеется простейший двухступенчатый конвейер, когда команда выполняется в одном такте с выборкой следующей. В отличие от Intel-архитектур, в "классическом" AVR нет аппаратного умножения/деления, однако в подсемействе Mega присутствуют операции умножения.
- **Раздельные шины памяти команд и данных.** AVR (как и большинство других микроконтроллеров) имеет т. н. *гарвардскую архитектуру*, где области памяти

программ и данных разделены (в отличие от классической архитектуры фон Неймана в обычных компьютерах, где память общая). Раздельные шины для этих областей памяти значительно ускоряют выполнение программы: данные и команды могут выбираться одновременно.

- **32 регистра общего назначения (РОН).** Atmel была первой компанией, далеко отошедшей от классической модели вычислительного ядра, в которой выполнение команд предусматривает обмен данными между АЛУ и запоминающими ячейками в общей памяти. Введение РОН в таком количестве (напомним, что в архитектуре *x*86 всего четыре таких регистра, а в *x*51 понятие РОН, как таковое, отсутствует) в ряде случаев позволяет вообще отказаться от расположения глобальных и локальных переменных в ОЗУ и от использования стека, операции с которым усложняют и загромождают программу. В результате структура ассемблерной программы приближается к программам на языках высокого уровня. Правда, это привело к некоторому усложнению системы команд, номенклатура которых для AVR больше, чем в других RISC-семействах (хотя значительная часть инструкций — псевдонимы).
- **Flash-память программ** (10 000 циклов стирание/запись) с возможностью внутрисистемного перепрограммирования и загрузки через последовательный канал прямо в готовой схеме. О преимуществах такого подхода, ныне ставшего общепринятым, подробно рассказано во *введении*.
- **Отдельная область энергонезависимой памяти** (EEPROM, 100 000 циклов стирание/запись) для хранения данных, с возможностью записи программным путем, или внешней загрузки через SPI-интерфейс.
- **Встроенные устройства для обработки аналоговых сигналов**: аналоговый компаратор и многоканальный 10-разрядный АЦП.
- **Сторожевой таймер**, позволяющий осуществлять автоматическую перезагрузку контроллера через определенные промежутки времени (например, для выхода из "спящего" режима).
- **Последовательные интерфейсы SPI**, **TWI** (**I^2C**) и **UART** (**USART**), позволяющие осуществлять обмен данными с большинством стандартных датчиков и других внешних устройств (в том числе таких, как персональные компьютеры) аппаратными средствами.
- **Таймеры-счетчики** с предустановкой и возможностью выбора источника счетных импульсов: как правило, один-два 8-разрядных и как минимум один 16-разрядный, в том числе могущие работать в режиме многоканальной 8-, 9-, 10-, 16-битовой широтно-импульсной модуляции (PWM).
- **Возможность работы при тактовой частоте** от 0 Гц до 16–20 МГц.
- **Диапазон напряжений питания от 2,7 до 5,5 В** (в некоторых случаях от 1,8 или до 6,0 В).
- **Многочисленные режимы энергосбережения**, отличающиеся числом узлов, остающихся подключенными. Выход из "спящих" режимов по сторожевому таймеру или по внешним прерываниям.

- **Встроенный монитор питания** — детектор падения напряжения (Brown-out Detection).

Здесь перечислены далеко не все особенности, характерные для различных моделей AVR. С некоторыми другими мы познакомимся в дальнейшем, а также на практике рассмотрим перечисленные подробнее. Но сначала дадим общую характеристику различных семейств AVR с точки зрения их преимущественного назначения.

Семейства AVR

В 2002 г. фирма Atmel начала выпуск новых подсемейств 8-разрядных МК на базе AVR-ядра. С тех пор все МК этого семейства делятся на три группы (подсемейства): Classic, Tiny и Mega. МК семейства Classic (AT90S*xxxx*) уже не выпускаются; дольше всего в производстве "задержалась" очень удачная (простая, компактная и быстродействующая модель) AT90S2313, но и она была в 2005 г. заменена на ATtiny2313. Все "классические" AVR с первыми цифрами 2 и 8 в наименовании модели (что означает объем памяти программ в килобайтах) имеют аналоги в семействах Tiny и Mega. Для Mega при программировании возможна установка специального бита совместимости, который позволяет без каких-либо изменений использовать программы, созданные для семейства Classic. Поэтому ряд примеров в данной книге в целях упрощения изложения приводится в версии для семейства Classic.

Примеры различных типов корпусов, в которых выпускаются микросхемы AVR, приведены на рис. 1.1. Более подробную информацию на эту тему можно найти в *приложении 1* (табл. П1.2), а также в технической документации на устройства. Отметим, что для радиолюбительских нужд и макетирования удобнее всего микросхемы в PDIP-корпусах, но не все модели МК в таких корпусах производятся.

Все семейства могут иметь две модификации: буква "L" в обозначении говорит о расширенном диапазоне питания 2,7–5,5 В, отсутствие такой буквы означает диапазон питания 4,5–5,5 В. При выборе конкретного типа микросхемы нужно быть внимательным, т. к. L-версии одновременно также и менее быстродействующие, у большинства из них максимальная тактовая частота ограничена значением 8 МГц. Для "обычных" версий максимальная частота составляет 16 или 20 МГц. Хотя, как правило, при запуске L-микросхем с напряжением питания 5 В на частотах до 10–12 МГц неприятностей ожидать не следует (аналогично версии без буквы L вполне могут работать при напряжении питания около 3 В, разумеется, не на экстремальных значениях частот), тем не менее при проектировании высоконадежных устройств следует учитывать это требование.

Микросхемы Tiny имеют Flash-ПЗУ программ объемом 1–8 кбайт и размещаются в основном в корпусах с 8–20 выводами (кроме ATtiny28), т. е. они в целом предназначены для более простых и дешевых устройств. Это не значит, что их возможности во всех случаях более ограниченны, чем у семейства Mega. Так, например, ATtiny26 при цене менее 2 долларов содержит таймер с высокоскоростным ШИМ-режимом (в других моделях такого нет), а также 11-канальный АЦП с возможностью работы в дифференциальном режиме, с регулируемым входным усилителем

и встроенным источником опорного напряжения, что характерно для старших моделей. Микросхема ATtiny2313, как уже говорилось, представляет собой улучшенную версию одного из наиболее универсальных и удобных "классических" AVR AT90S2313.

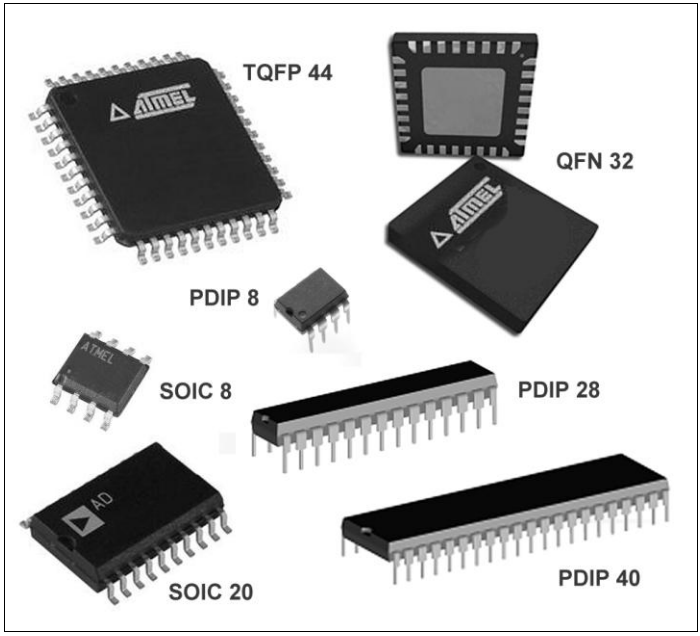


Рис. 1.1. Примеры различных типов корпусов для МК AVR

Подсемейство Mega оснащено Flash-ПЗУ программ объемом 8–256 кбайт и корпусами с 28–100 выводами. В целом МК этой группы более "навороченные", чем Tiny, имеют более разветвленную систему встроенных устройств с более развитой функциональностью.

Таблицы с основными характеристиками некоторых моделей Tiny и Mega из числа самых ходовых приведены в *приложении 1*. Там же даны некоторые общие технические характеристики семейства AVR. Более подробные сведения можно почерпнуть из [1, 2] и фирменной технической документации, которая доступна на сайте Atmel для каждой модели.

Кроме этих трех семейств, на базе AVR-ядра выпускаются специализированные микросхемы для работы с USB-интерфейсом (AT90USB*xxxx*), промышленным интерфейсом CAN (AT90CAN*xxx*), для управления ЖК-дисплеями (ATmega329 и др.), с беспроводным интерфейсом IEEE 802.15.4 (ZigBee) для предприятий торговли и некоторые другие. В последнее время некоторые микроконтроллеры серий Tiny и Mega стали выпускаться в версиях со сверхмалым потреблением (технология picoPower с напряжением питания от 1,8 В, в конце наименования МК этой серии добавлена буква "P") и высокотемпературных для использования в автомобильной промышленности (версии Automotive). Появилось семейство XMega с напряжением питания 1,8–3,6 В, повышенным быстродействием (тактовая частота до 32 МГц),

12-разрядным 16-канальным АЦП и 2–4 каналами ЦАП (до сих пор в структуре AVR они отсутствовали), несколькими каналами UART и других последовательных портов (причем с возможностью работы в автономном режиме, при остановленном ядре), встроенной поддержкой криптографии, усовершенствованным режимом picoPower и другими "наворотами". Существует также отдельное семейство 32-разрядных МК AVR32, предназначенное для высокоскоростных приложений, таких как обработка видеопотока или распознавание образов в реальном времени.

Особенности практического использования МК AVR

При использовании AVR возникает ряд вопросов практического характера, игнорирование которых может иногда привести к неработоспособности или сбоям устройства (а в некоторых случаях — даже к невозможности его запрограммировать). Например, одна из таких проблем — возможность потери содержимого EEPROM при выключении питания. Эту и подобные проблемы мы подробно рассмотрим в соответствующих главах. Здесь же остановимся на некоторых общих вопросах включения МК AVR.

О потреблении

МК AVR потребляют в среднем 5–15 мА (без учета потребления внешних устройств через выводы МК). Потребляемый ток зависит не только от степени "навороченности" модели, но и от тактовой частоты и напряжения питания. На рис. 1.2 приведена типовая диаграмма зависимости тока потребления от напряжения питания и тактовой частоты для младших моделей семейства Mega.

Из рис. 1.2, в частности, следует, что значительно уменьшить потребление можно, снижая тактовую частоту в тех случаях, когда время выполнения программы некритично. Это позволяет упростить программу, отказавшись от режимов энергосбережения: например, при установке "часового" кварца 32 768 Гц в качестве тактирующего потребление МК может составить порядка 200–300 мкА.

Заметки на полях

Величину тока потребления 1–2 мА и менее можно условно считать приемлемой для батарейных устройств, которые рассчитаны на долговременный режим непрерывной работы. Элементы типоразмера АА (типа alcaline, т. е. щелочные) имеют емкость порядка 2000 мА·ч, т. е. устройство с указанным потреблением от этих элементов проработает не менее 1000 ч (реально даже несколько больше) или более 40 суток. Время работы от батарей типоразмера D с энергоемкостью порядка 15–18 000 мА·ч составит около года, чего для большинства практических применений достаточно. Выбирать для питания подобных устройств (особенно, включающихся периодически на короткое время) следует именно щелочные элементы, т. к. они обладают большой емкостью, не текут при переразряде и, главное, имеют значительно больший срок хранения (порядка 7 лет) по сравнению с другими типами элементов.

Но внимательное рассмотрение вопроса показывает, что именно этим — упрощением программы — в подавляющем большинстве случаев преимущества более

низкой тактовой частоты и ограничиваются. Графики на рис. 1.2 линейны, отсюда следует, что пропорционально снижению тактовой частоты растет время выполнения команд. Таким образом, процедура, выполнение которой при тактовой частоте 4 МГц займет 100 мкс, при тактовой частоте 32 768 Гц будет длиться более 12 мс. Легко подсчитать, что в том и другом случае количество энергии, потребленной на выполнение этой процедуры, будет одинаковым.

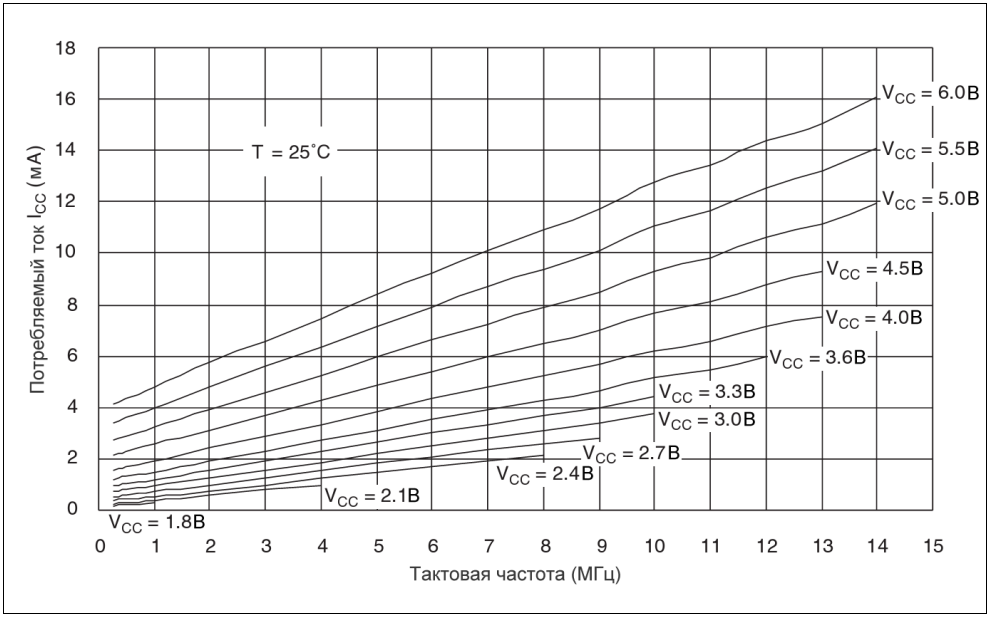


Рис. 1.2. Диаграмма зависимости тока потребления от напряжения питания и тактовой частоты для младших моделей семейства Mega

Поэтому можно сделать следующий общий вывод: если вы не желаете вникать в тонкости режимов энергосбережения и не реализуете их в программе, то для общего снижения потребления нужно выбирать тактовую частоту как можно ниже (на практике обычно достаточно ограничиться величиной 1 МГц, т. к. дальнейшее снижение, скорее всего, не даст эффекта из-за дополнительного потребления внешними цепями, неизбежно присутствующими во всех схемах). Если же у вас предусмотрен один из режимов "глубокого" энергосбережения (см. *главу 4*), то тактовая частота с точки зрения суммарного потребления практически не имеет значения.

Другое дело — выбор напряжения питания, которое желательно сделать как можно меньше, если это позволяют внешние устройства. Зависимость тока потребления от напряжения питания, как легко уяснить из графиков на рис. 1.2, нелинейная: с увеличением напряжения ток потребления быстро возрастает. Поэтому снижать напряжение питания даже с учетом ограничения на тактовую частоту для большинства моделей AVR (не более 8 МГц при питании 2,7 В) все равно выгодно. Например, устройство с питанием 3 В при тактовой частоте 8 МГц, согласно рис. 1.2, будет потреблять около 3 мА или, в пересчете на единицы мощности, 9 мВт; на процеду-

ру длительностью 100 мкс уйдет энергия 0,9 мкДж. При частоте 16 МГц та же процедура займет 50 мкс, но потребление при необходимом напряжении питания 5 В составит около 14 мА, т. е. 70 мВт; итого на выполнение процедуры уйдет энергия 3,5 мкДж, почти в 4 раза больше.

Для всех внешних цифровых устройств, за редчайшим исключением, можно подобрать современный аналог, предназначенный для работы при напряжениях 2,7–3,0 В (и даже ниже, если модель контроллера это позволяет), так что с этой стороны ограничений нет; то, что большинство примеров в этой книге ориентировано на напряжение питания 5 В, есть лишь дань традиции. К тому же примеры эти, как правило, подразумевают питание от сети, где потребление не имеет большого значения. Лимитировать снижение напряжения питания могут светодиодные индикаторы (из-за того, что прямое падение напряжения на светодиодах само по себе составляет порядка 2 В, а для больших индикаторов даже 5 В для управления недостаточно), но в таких устройствах потребление контроллера уже не играет большой роли: четыре семисегментные цифры сами по себе будут потреблять ток порядка 100 мА и более. Другой случай представляют аналоговые схемы, где повышение напряжения питания выгодно с точки зрения увеличения отношения "сигнал-шум".

Заметим, что выводы AVR могут в долговременном режиме отдавать значительный ток (до 20-40 мА), однако не следует забывать об общем суммарном ограничении на потребление по выводу питания (см. табл. П1.3). Следует также отметить, что при подаче аналоговых напряжений на входы АЦП входной цифровой КМОП-элемент (вход соответствующего порта) не отключается, и при значении данного напряжения вблизи порога срабатывания элемента это может приводить к возрастанию потребления за счет протекания сквозного тока через выходные каскады КМОП (в том числе иногда и при нахождении микросхемы в "спящем" режиме, см. *главу 14*). Этого недостатка лишены микросхемы с технологией picoPower.

Некоторые особенности применения AVR в схемах

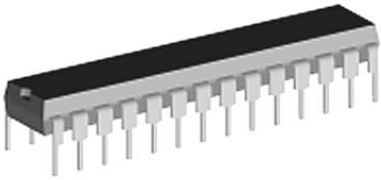
У большинства выводов МК имеется встроенный подключаемый "подтягивающий" (т. е. подсоединенный к шине питания) резистор, что, казалось бы, решает одну из обычных схемотехнических проблем, когда наличие такого резистора требуется для подключения двухвыводных кнопок или выходов с "открытым коллектором". Однако в критичных случаях необходим внешний резистор сопротивлением 2–5 кОм (в критичных для потребления случаях до 10–30 кОм).

"Подтягивающий" резистор следует устанавливать не только на выводе /RESET (о чем пойдет речь в *главе 2*), но и в том случае, когда выводы SCK, MOSI и MISO соответствующих портов используются для программирования и подключены к программирующему разъему ISP (см. *главу 5*), а также по выводам внешних прерываний, если они задействованы. Если эти выводы не "подтягивать" к напряжению питания дополнительными резисторами (хотя это и не оговорено в технической документации), то не исключены ложные срабатывания внешних прерываний, перезапуск системы, а при очень мощных помехах — даже порча программы в памяти программ. С другой стороны, когда выводы программирования служат и в каче-

стве обычных портов, сконфигурированных на выход, а в устройстве применяются режимы энергосбережения, наличие "подтягивающих" резисторов может привести к лишнему потреблению тока (при установке вывода в логический ноль через резистор потечет ток от источника питания на вход МК). Если реализован один из режимов энергосбережения, то нужно тщательно проанализировать схему, чтобы исключить ситуации, при которых через эти резисторы протекает ток.

Также всегда следует устанавливать внешние резисторы при работе выводов МК на общую шину, как в интерфейсе I^2C (или просто при подсоединении входа МК к выходу другого устройства с открытым коллектором, например, мониторов питания, описанных в *главе 3*), при подключении к двухвыводным кнопкам (особенно при наличии внешнего прерывания, см. *главы 4* и *5*). Сопротивление встроенного резистора (на самом деле представляющего собой, разумеется, полевой транзистор) в таких случаях слишком велико для того, чтобы электромагнитные помехи ("наводки") на нем эффективно "садились".

Микросхемы AVR, как и всякая КМОП-логика, благодаря высокому порогу срабатывания эффективно защищены от помех по шине "земли". Однако они ведут себя гораздо хуже при помехах по шине питания. Поэтому не забывайте о развязывающих конденсаторах, которые нужно устанавливать непосредственно у выводов питания (керамические 0,1–0,5 мкФ), а также про качество сетевых выпрямителей и стабилизаторов.



ГЛАВА 2

Общее устройство, организация памяти, тактирование, сброс

Общая структура внутреннего устройства МК AVR приведена на рис. 2.1. На этой схеме показаны все основные компоненты AVR (за исключением модуля JTAG); в отдельных моделях некоторые составляющие могут отсутствовать или различаться по характеристикам, неизменным остается только общее 8-разрядное процессорное ядро (GPU, General Processing Unit). Кратко опишем наиболее важные компоненты, большинство из которых мы подробно будем рассматривать в дальнейшем.

Начнем с памяти. В структуре AVR имеются три разновидности памяти: flash-память программ, ОЗУ (SRAM) для временных данных и энергонезависимая память (EEPROM) для долговременного хранения констант и данных. Рассмотрим их по отдельности.

Память программ

Объем встроенной flash-памяти программ в AVR-контроллерах составляет от 1 кбайта у ATtiny11 до 256 кбайт у ATmega2560. Первое число в наименовании модели соответствует величине этой памяти из ряда: 1, 2, 4, 8, 16, 32, 64, 128 и 256 кбайт. Память программ, как и любая другая flash-память, имеет страничную организацию (размер страницы, в зависимости от модели, составляет от 64 до 256 байт). Страница может программироваться только целиком. Число циклов перепрограммирования достигает 10 тыс.

С точки зрения программиста память программ можно считать построенной из отдельных ячеек — слов по два байта каждое. Устройство памяти программ (и только этой памяти) по двухбайтовым словам — очень важный момент, который нужно твердо усвоить. Такая организация обусловлена тем, что любая команда в AVR имеет длину ровно два байта. Исключение составляют команды `JMP`, `CALL` и некоторые другие (например, `LDS`), которые оперируют с 16-разрядными и более длинными адресами, длина этих команд равна четырем байтам и они применяются лишь в моделях с памятью программ объемом свыше 8 кбайт (подробнее см. *главу 5*). Во всех остальных случаях счетчик команд сдвигается при выполнении очередной

команды на два байта (одно слово), поэтому необходимую емкость памяти легко подсчитать, зная число используемых команд. Абсолютные адреса в памяти программ (указываемые, например, в таблицах векторов прерываний в техническом описании МК) также отсчитываются в словах.

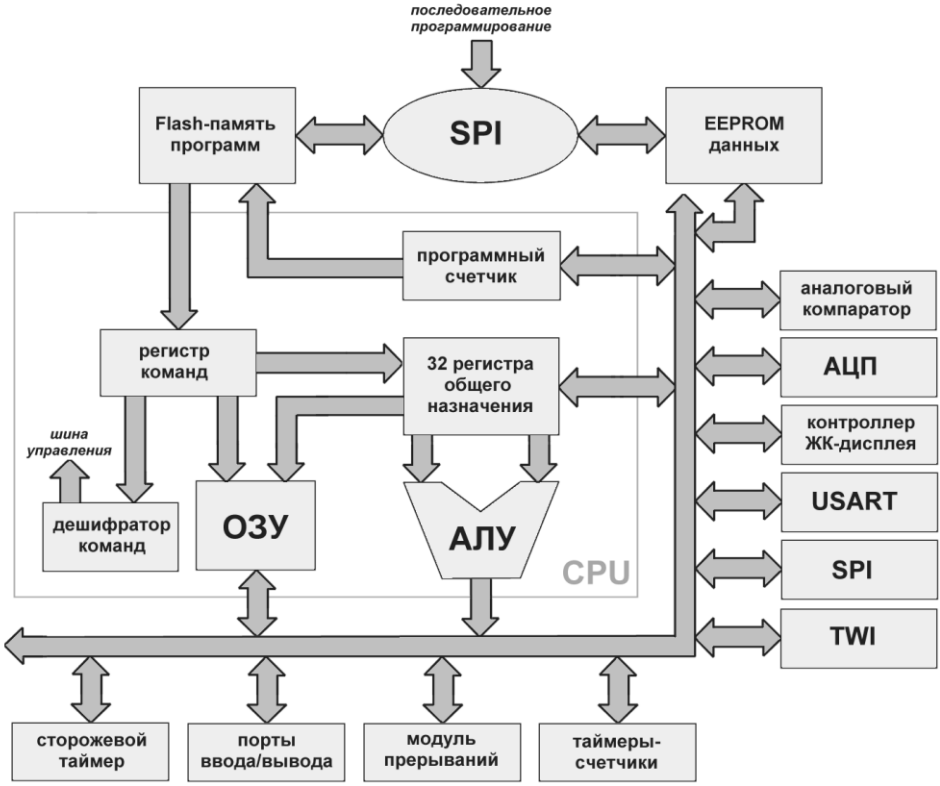


Рис. 2.1. Общая структурная схема микроконтроллеров AVR

Заметки на полях

Приведем пример интересного случая адресации, который представляет команда для чтения констант из памяти `LPM` (а также ELPM в МК с памятью программ 128 кбайт и более). Эта команда подразумевает чтение по *байтовому* адресу, указанному в двух старших РОН (образующих т. н. регистр `Z`, см. далее). Однако чтобы не нарушать "чистоту" концепции организации памяти программ по словам, разработчики запутали этот простой вопрос, указав в описании, что при вызове команды `LPM` старшие 15 разрядов регистра `Z` адресуют *слово* в памяти, а младший разряд выбирает младший или старший байт (при равенстве разряда 0 или 1 соответственно) этого слова. Легко, однако, заметить, что байтовая и пословная организации памяти при таком подходе эквивалентны.

Последний адрес существующего объема памяти программ для конкретной модели обозначается константой `FLASHEND`. По умолчанию все контроллеры AVR всегда начинают выполнение программы с адреса $0000. Если в программе нет прерываний, то с этого адреса может начинаться прикладная программа. В противном слу-

чае по данному адресу располагается т. н. таблица *векторов прерываний*, подробнее о которой мы будем говорить в *главах 4* и *5*. Здесь укажем лишь, что первым в этой таблице (по тому же адресу $0000) всегда размещается вектор сброса `RESET`, который указывает на процедуру, выполняющуюся при сбросе МК (в том числе и при включении питания).

Примечание

В ассемблере AVR можно обозначать шестнадцатеричные числа в "паскалевском" стиле, предваряя их знаком $, при этом стиль языка C (0x00) тоже действителен, а вот "интеловский" способ (00h) не работает. Подробнее об обозначениях чисел различных систем счисления в AVR-ассемблере см. *главу 5*.

В последних адресах памяти программ контроллеров семейства Mega может располагаться т. н. *загрузчик* — специальная программа, которая управляет загрузкой и выгрузкой прикладных программ из основного объема памяти. В этом случае положение вектора сброса и всей таблицы векторов прерываний (т. е. фактически начального адреса, с которого начинается выполнение программы) может быть изменено установкой специальных конфигурационных ячеек (см. *главу 5*).

Память данных (ОЗУ, SRAM)

В отличие от памяти программ, адресное пространство памяти данных адресуется побайтно (а не пословно). Адресация полностью линейная, без какого-то деления на страницы, сегменты или банки, как это принято в некоторых других системах. Младшие МК семейства Tiny (включая Tiny1*x* и Tiny28) памяти данных, как таковой, не имеют, ограничиваясь лишь регистровым файлом (РОН) и регистрами ввода-вывода (РВВ). В других моделях объем встроенной SRAM колеблется от 128 байт в представителях семейства Tiny (например, у ATtiny2313) до 4–8 кбайт у старших моделей Mega.

Адресное пространство статической памяти данных (SRAM) условно делится на несколько областей, показанных на рис. 2.2. Темной заливкой выделена часть, относящаяся к собственно встроенной SRAM, до нее по порядку адресов расположено адресное пространство регистров (первые 32 байта занимает РОН, еще 64 — РВВ). Для старших моделей Mega со сложной структурой (например, ATmega128) 64-х регистров ввода-вывода может оказаться недостаточно, поэтому в них для дополнительных РВВ выделяется отдельное адресное пространство (от $60 до максимально возможного в байтовой адресации значения $FF, итого таких регистров может быть всего 160).

Заметки на полях

В архитектуре МК AVR понятие "ввода-вывода" употребляется в двух смыслах: во-первых, имеются "порты ввода-вывода" (I/O ports), которые мы рассмотрим в *главе 3*. Во-вторых, "регистрами ввода-вывода" (РВВ) в структуре AVR называются регистры, которые обеспечивают доступ к дополнительным компонентам, внешним по отношению к GPU, за исключением ОЗУ (в том числе и к портам ввода-вывода). Такое подразделение приближает структуру МК AVR к привычной конфигурации персонального компьютера, где доступ к любым внешним по отношению к центральному процессору компонентам, кроме памяти, осуществляется через порты ввода-вывода.

Для некоторых моделей Mega (ATmega8515, ATmega162, ATmega128, ATmega2560 и др.) предусмотрена возможность подключения внешней памяти объемом до 64 кбайт, которая может быть любой статической разновидностью (SRAM, Flash или EEPROM) с параллельным интерфейсом.

Отметим, что адреса РОН и РВВ не отнимают пространство у ОЗУ данных (за исключением подключаемой внешней памяти у старших моделей Mega, максимальный адрес которой ограничен значением $FFFF): так, если в конкретной модели МК имеется 512 байт SRAM, а пространство регистров занимает первые 96 байт (до адреса $60), то адреса SRAM займут адресное пространство от $0060 до $025F (т. е. от 96-й до 607-й ячейки включительно). Конец встроенной памяти данных обозначается константой `RAMEND`.

Рис. 2.2. Адресное пространство статической памяти данных (SRAM) микроконтроллеров AVR

Операции чтения/записи в память одинаково работают с любыми адресами из доступного пространства, и при работе с SRAM нужно быть внимательным: вместо записи в память вы легко можете "попасть" в какой-нибудь регистр. Например, команда загрузки значения регистра `r16` в регистр `r0` (`mov r0,r16`) равносильна записи в SRAM по нулевому адресу (`sts $0000,r16`). Адрес в памяти для РОН совпадает с его номером. В то же время для непосредственной записи в РВВ по его адресу в памяти к номеру регистра следует прибавить $20: так, регистр флагов `SREG`, который для большинства моделей располагается в конце таблицы РВВ по адресу $3F, в памяти имеет адрес $5F. Устанавливать РОН и РВВ прямой адресацией памяти неудобно: такая запись всегда отнимает два такта вместо одного, характерного для большинства других команд, хотя иногда это позволяет обойти ограничения на манипуляции с некоторыми РВВ. Но если имеется готовая программа, работающая с SRAM, то при замене моделей процессоров на более старшие нужно быть внимательным из-за того, что в них младшие адреса SRAM могут перекрываться дополнительными РВВ.

Энергонезависимая память данных (EEPROM)

Все модели МК AVR (кроме снятого с производства ATtiny11) имеют встроенную EEPROM для хранения констант и данных при отключении питания. В разных моделях объем ее варьируется от 64 байт (ATtiny1x) до 4 кбайт (старшие модели Mega). Конец EEPROM обозначается константой `EEPROMEND` (это обозначение введено только для более поздних моделей AVR, потому при использовании этой константы иногда ее придется определять самому). Число циклов перепрограммирования EEPROM может достигать 100 тыс.

Напомним, что EEPROM отличается от Flash возможностью выборочного программирования побайтно (в принципе даже побитно, но этот способ недоступен пользователю). Однако в старших моделях семейства EEPROM, как и flash-память программ, имеет страничную организацию, правда, страницы эти невелики — до 4 байт каждая. На практике, как при программировании EEPROM по последовательному каналу (т. е. через SPI-интерфейс программирования), так и при записи и чтении EEPROM из программы, эта особенность не имеет значения, и доступ осуществляется побайтно.

Чтение из EEPROM осуществляется в течение одного машинного цикла (правда, на практике оно растягивается на четыре цикла, но программисту следить за этим специально не требуется). А вот запись в EEPROM протекает значительно медленнее, и к тому же с точно не определенной скоростью: цикл записи одного байта может занимать от 2 до ~ 4 мс и более. Процесс записи регулируется встроенным *RC*-генератором, частота которого нестабильна (при более низком напряжении питания можно ожидать, что время записи будет больше). За такое время при обычных тактовых частотах МК успевает выполнить несколько тысяч команд, потому программирование процедуры записи требует аккуратности: например, нужно следить, чтобы в момент записи не "вклинилось" прерывание (подробнее об этом см. *главы 4* и *9*).

Главная же сложность при работе с EEPROM — возможность повреждения ее содержимого при недостаточно быстром снижении напряжения питания в момент выключения. Обусловлено это тем, что при уменьшении напряжения питания до некоторого порога (ниже порога стабильной работы, но недостаточного для полного выключения) из-за колебаний напряжения МК начинает выполнять произвольные команды, в том числе может осуществить процедуру записи в EEPROM. Если учесть, что типовая команда МК AVR выполняется за десятые доли микросекунды, то ясно, что никакой реальный источник питания не может обеспечить снижение напряжения до нуля за нужное время. По опыту автора при питании от обычного стабилизатора типа LM7805 с рекомендованными значениями емкости конденсаторов на входе и на выходе содержимое EEPROM будет неизбежно испорчено примерно в половине случаев.

Этой проблемы не должно существовать, если константы записывают в EEPROM при программировании МК, а процедура записи в программе отсутствует (о том, как сформировать файл с данными для EEPROM, см. *раздел "Директивы и функции" главы 5*). Большая сохранность данных в таких случаях подтверждается и эм-

пирическими наблюдениями, и тем, что разрешение записи в EEPROM — процедура двухступенчатая (см. *главу 9*). Во всех же остальных случаях (а их, очевидно, абсолютное большинство — в EEPROM чаще всего хранят пользовательские установки и текущую конфигурацию при выключении питания) приходится принимать специальные меры. Наиболее кардинальной и универсальной из них является установка внешнего монитора питания, удерживающего МК в состоянии сброса при уменьшении напряжения питания ниже пороговой величины. Той же цели служит встроенный детектор падения напряжения (Brown-out Detection, BOD), имеющийся практически во всех моделях Tiny и Mega, но техническая документация не исключает при этом для надежности дублирования его и внешним монитором питания. Подробнее о схеме BOD и режимах сброса МК см. *далее в этой главе*, а о программировании EEPROM и мерах предосторожности при ее использовании см. *главу 9*.

Способы тактирования

Канонический способ тактирования МК — подключение кварцевого резонатора к соответствующим выводам (рис. 2.3, *а*). Емкость конденсаторов C1 и C2 в типовом случае должна составлять 15–22 пФ (может быть увеличена до 33–47 пФ с одновременным повышением потребления). В большинстве моделей Tiny и Mega имеется специальный конфигурационный бит `CKPOT`, который позволяет регулировать потребление. При установке этого бита в 1 (незапрограммированное состояние) размах колебаний генератора уменьшается, однако при этом сужается возможный диапазон частот и общая помехоустойчивость, поэтому задействовать этот режим не рекомендуется. Может быть также выбран низкочастотный кварцевый резонатор (например, "часовой" 32 768 Гц), при этом конденсаторы C1 и C2 могут отсутствовать, т. к. при установке `CKPOT` в значение 0 подключаются имеющиеся в составе МК внутренние конденсаторы емкостью 36 пФ.

Кварцевый резонатор можно заменить керамическим. Автору этих строк удавалось запускать МК на нестандартных частотах, используя вместо кварца в том же подключении миниатюрную индуктивность (при ее значении 4,7 мкГн и емкостях конденсаторов 91 пФ частота получается около 10 МГц), что заодно позволяет немного уменьшить габариты схемы.

Естественно, тактировать МК можно и от внешнего генератора (рис. 2.3, *б*). Особенно это удобно, когда требуется либо синхронизировать МК с внешними компонентами, либо получить очень точную частоту тактирования, выбрав соответствующий генератор (например, серии SG-8002 фирмы Epson).

Наоборот, когда точность не требуется, можно подключить внешнюю *RC*-цепочку (рис. 2.3, *в*). В этой схеме емкость C1 должна быть не менее 22 пФ, а резистор R1 выбирается из диапазона 3,3–100 кОм. Частота при этом определяется по формуле $F = 2/3\ RC$. C1 можно не устанавливать вообще, если записать лог. 0 в конфигурационную ячейку `CKPOT`, подключив тем самым внутренний конденсатор 36 пФ.

Наконец, можно вообще отказаться от внешних компонентов и обойтись встроенным *RC*-генератором, который способен работать на четырех приблизительных

значениях частот (1, 2, 4 и 8 МГц). В ряде моделей предусмотрена возможность подстройки частоты этого генератора (подробнее см. [2] или техническое описание конкретных моделей). Эту возможность наиболее целесообразно использовать в младших моделях Tiny, выпускающихся в 8-контактном корпусе — тогда выводы, предназначенные для подключения резонатора или внешнего генератора, можно задействовать для других целей, как обычные порты ввода-вывода.

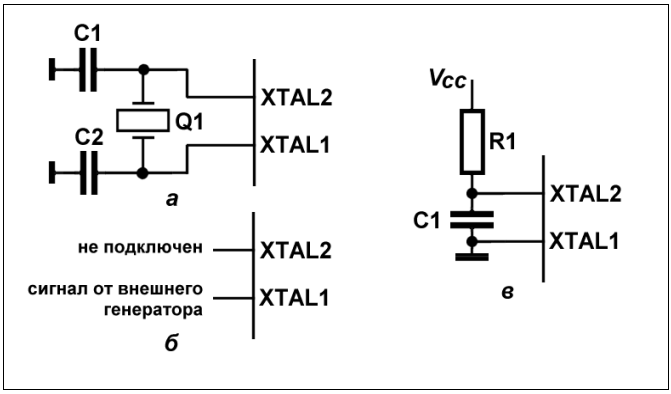


Рис. 2.3. Способы тактирования МК AVR с использованием: *а* — кварцевого резонатора; *б* — внешнего генератора; *в* — *RC*-цепочки

Семейство Classic встроенного *RC*-генератора не имеет, а специальных конфигурационных ячеек у этих МК значительно меньше, и в общем случае на них можно не обращать внимания. Для других семейств это не так. По умолчанию МК семейств Tiny и Mega установлены в состояние для работы со встроенным генератором на частоте 1 МГц (`CKSEL` = 0001), поэтому для других режимов нужно соответствующим образом установить конфигурационные ячейки `CKSEL` (см. табл. 2.1). При этом следует учитывать, что состояние ячеек `CKSEL` = 0000 (зеркальное по отношению к наиболее часто употребляемому значению для кварцевого резонатора 1111) переводит МК в режим тактирования от внешнего генератора, и при этом его нельзя даже запрограммировать без подачи внешней частоты. О рекомендуемых установках конфигурационных ячеек и об особенностях их программирования см. также *главу 5*.

Таблица 2.1. *Установка конфигурационных ячеек* `CKSEL` *в зависимости от режимов тактирования*

`CKSEL3...0`	**Источник тактирования**	**Частота**
0000	Внешняя частота	0... 16 МГц
0001	Встроенный *RC*-генератор	1 МГц
0010	Встроенный *RC*-генератор	2 МГц
0011	Встроенный *RC*-генератор	4 МГц
0100	Встроенный *RC*-генератор	8 МГц
0101	Внешняя *RC*-цепочка	< 0,9 МГц