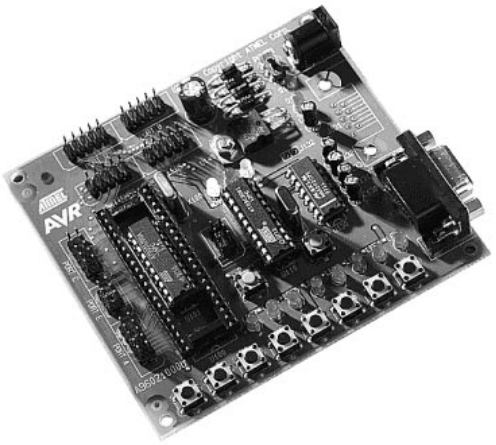


AVR: НОВЫЕ 8-РАЗРЯДНЫЕ RISC-МИКРОКОНТРОЛЛЕРЫ ФИРМЫ ATMEL

Фирма ATMEL хорошо известна российским разработчикам. Ее микросхемы перепрограммируемой памяти и микроконтроллеры семейства AT89 пользуются заслуженным спросом на российском рынке. В декабре 1997 г. фирма ATMEL впервые в мире выпустила новое семейство 8-разрядных микроконтроллеров, построенных на базе расширенной RISC-архитектуры. В статье дан краткий обзор архитектуры AVR и описан цифровой термометр – конкретная разработка на основе микроконтроллера этого семейства.

ОБЩИЕ СВЕДЕНИЯ

AVR представляет собой пример воплощения современных тенденций в построении микропроцессоров вообще и микроконтроллеров в частности. AVR-технология фирмы ATMEL является абсолютно новой разработкой, не базирующейся на какой-либо известной архитектуре. Это смелое решение позволило создать микроконтроллер с исключительным быстродействием и чрезвычайно эффективным программным кодом.

Серия AVR на самом деле содержит три семейства. Первое, получившее название **tiny AVR** (префикс **ATtiny**), выпускается в 8-выводных корпусах и предназначено для недорогих устройств, таких как пульты дистанционного управления и приборы противопожарной безопасности. Объем памяти программ от 1 до 2 килобайт. Семейство **classic AVR** (префикс **AT90S**) выпускается в корпусах с 20, 40 и 44 выводами, содержит память программ от 1 до 8 килобайт и предназначено для типовых применений. Семейство **mega AVR** (префикс **ATmega**) выпускается в корпусах с 64 выводами, имеет объем памяти программ до 128 килобайт, 8-канальный 10-битный АЦП, часы реального времени и различные периферийные узлы. Оно предназначено для применений, использующих большой объем памяти программ и критичных к габаритам системы. В табл. 1 приведены основные сведения о микросхемах AVR.

ОБЗОР АРХИТЕКТУРЫ

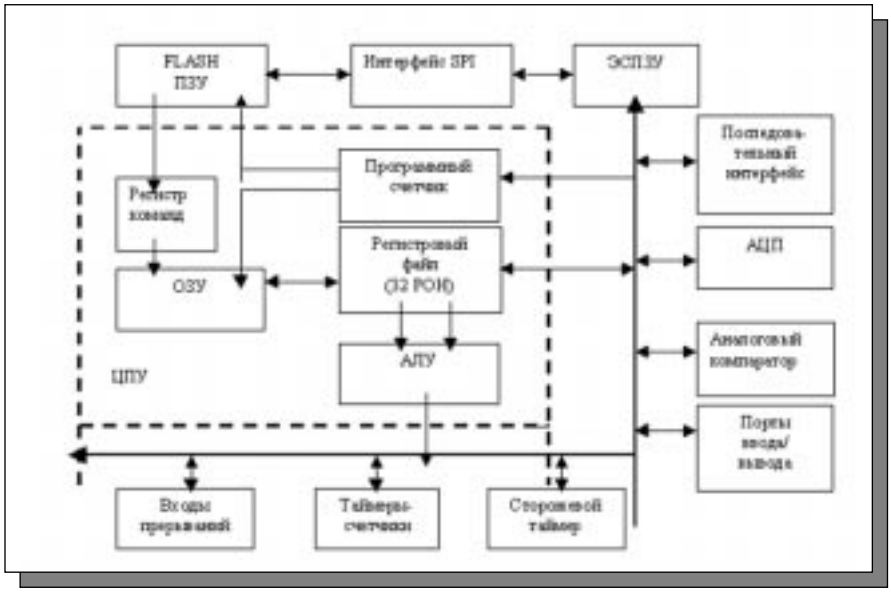
Микроконтроллеры семейства AVR построены на основе расширенной Гарвардской архитектуры с раздельными шинами адресов и данных и раздельными адресными пространствами памяти программ и памяти данных. Максимальное адресное пространство составляет 8 МБайт для памяти программ и 8 МБайт для памяти данных. Обобщенная блок-схема микроконтроллеров серии AVR представлена на рис. 1. В сравнении с «классическими» микроконтроллерами, имеющими один аккумулятор, контроллеры семейства AVR оснащены необычно большим банком регистров состоящим из 32 регистров (R0–R31), называемых «рабочими регистрами общего назначения» (General Purpose Working Registers), а сам блок из 32 регистров называется «регистровый файл»

Таблица 1. Характеристики контроллеров

Микросхема	ПЗУ, кБайт	Стат. ОЗУ, Байт	ЭСПЗУ, Байт	Таймеры	Посл. порт	АЦП	Диап. частот, МГц	Корпус (к-во выводов)
AT90S1200	1	0	64	1	–	–	0…12	20
AT90S2313	2	128	128	2	+	-	0…10	20
AT90S2323	2	128	128	2	-	-	0…10	8
AT90S2343	2	128	128	2	-	-	0…10	8
AT90S4414	4	256	256	2	+	-	0…8	40,44
AT90S4434	4	256	256	3	+	+	0…8	40,44
AT90S8515	8	512	512	2	+	-	0…8	40,44
AT90S8535	8	512	512	3	+	+	0…8	40,44
AT Mega103	128	4096	4096	3	+	+	0…6	64
AT Mega603	64	4096	2048	3	+	+	0…6	64

Примечание: AT90S1200 и AT90S2343 имеют встроенный задающий RC генератор (1 MHz).

Рис. 1. Структурная схема AVR



(Register File). Данные, находящиеся в любом из этих регистров, доступны для немедленного использования в любой момент времени, поэтому можно говорить, что контроллер содержит 32 аккумулятора. Благодаря такой организации существенно снижается количество пересылок между памятью и аккумулятором, характерное для одноаккумуляторных структур. Вследствие этого существенно уменьшается размер программного кода. Также сокращает длину программы и возможность обращения к регистру по индексу, находящемся в другом регистре (Z-регистр, см. ниже).

Контроллер имеет внутреннюю 16-разрядную шину команд и 8-разрядную шину данных. К внутренней шине данных подключены периферийные модули. Все микросхемы AVR имеют порты ввода-вывода, входы прерываний, сторожевой таймер (Wathcdog Timer), один или более таймер-счетчик, аналоговый компаратор. Наиболее функционально насыщенные контроллеры содержат также 8-канальный АЦП и часы реального времени.

Регистровый файл размещен в нижней области адресного пространства. Регистрам R0-R31 соответствуют адреса в памяти данных с $0000 до $001F. Это позволяет также обращаться к регистрам, как к ячейкам памяти. Каждый регистр может хранить как адреса, так и данные. Шесть старших регистров (R26-R31) могут объединяться в пары, образуя три шестнадцатибитных указателя адреса. Они называются X-регистр, Y-регистр и Z-регистр соответственно.

Следующие 64 ячейки образуют пространство ввода/вывода. По этим адресам размещаются регистры управления, таймеры-счетчики, АЦП и другие периферийные модули. К регистрам ввода/вывода можно обращаться как непосредственно, либо как к ячейкам памяти с адресами 0020H–$005F, что очень удобно при работе с портами.

Программная память имеет одноуровневый конвейер. Во время выполнения очередной инструкции происходит выборка следующей. Непосредственное подключение регистрового файла к АЛУ обеспечивает выполнение команды за один такт кварцевого генератора. Два операнда извлекаются из регистрового файла, затем происходит выполнение операции и результат сохраняется в регистровом файле. Для сравнения: в микроконтроллерах семейства x51 команда выполняется за 12 тактов, а в pic-контроллерах фирмы Microchip – за 4 такта кварцевого генератора. Таким образом, можно говорить о 4...12 кратном увеличении быстродействия микроконтроллеров AVR только за счет однотактного выполнения команды. Если разрабатываемое устройство критично к потребляемой мощности, то можно, оставив без изменения быстродействие, в те же 4...12 раз уменьшить тактовую частоту генератора, соответственно снизив ток, потребляемый микроконтроллером.

Микроконтроллеры имеют два режима энергосбережения – idle mode и power down mode. В устройствах, имеющих на кристалле часы реального времени реализован третий режим – power save mode. В «спящий» режим микроконтроллер входит после выполнения команды SLEEP при условии, что бит SE в регистре управления MCUCR установлен в 1. Биты SM1 и SM0 (в MCUCR) выбирают один из трех режимов «сна». В режиме idle mode (SM1/SM0 = 00) контроллер просто останавливается до следующего прерывания. Все его внутренние устройства продолжают функционировать. При появлении прерывания контроллер «просыпается», выполняет одну команду после SLEEP и затем отрабатывает прерывание. Это относится ко всем режимам энергосбережения. В режиме power down mode (SM1/SM0 = 10) внешний осциллятор останавливается. Систему можно «разбудить» либо по сторожевому таймеру (если он разрешен), либо внешним прерыванием, либо внешним сигналом RESET. Режим power save mode (SM1/SM0 = 11) идентичен power down mode, но в этом режиме часы реального времени (Timer/Counter0) продолжают работать, если подключен часовой кварц.

ПЕРИФЕРИЙНЫЕ МОДУЛИ

Микроконтроллеры AVR в зависимости от варианта исполнения имеют различный набор периферии, таким образом, конкретные блоки могут отсутствовать в какой-либо модификации контроллера.

Таймеры/счетчики. Контроллеры содержат один 8-битный таймер/счетчик и один или более 16-битный. Каждый из них имеет независимый 10-битный предварительный делитель со следующими выходными частотами: CLK/8, CLK/64, CLK/256 и CLK/1024, где CLK – частота задающего генератора. На входы таймеров можно подавать также сигналы от внешних источников. 16-битный таймер/счетчик имеет несколько дополнительных режимов. Возможно сравнение текущего состояния с двумя предустановленными значениями и функционирование в режиме ШИМ-модулятора с 8-, 9- или 10-битным разрешением. Имеется также функция запоминания текущего состояния таймера/счетчика по внешнему событию.

Сторожевой таймер – работает от независимого встроенного RC-генератора с частотой 1 МГц (при питании 5 В). Допускается восемь различных временных интервалов срабатывания сторожевого таймера – от 16 мс до 2,048 с. Срабатывание таймера эквивалентно подаче сигнала RESET на соответствующий вход контроллера.

Таблица 2. Процедуры умножения и деления

Оптимизация	по времени выполнения				по занимаемому пространству			
Процедуры беззнакового умножения и деления								
Процедура	MUL8x8	DIV 8x8	MUL 16x16	DIV 16x16	MUL 8x8	DIV 8x8	MUL 16x16	DIV 16x16
Время (max, clocks)	36	68	107	198	60	97	156	251
Размер (words)	35	67	106	197	10	14	15	19
Процедуры знакового умножения и деления								
Процедура	MUL8x8	DIV 8x8	MUL 16x16	DIV 16x16	MUL 8x8	DIV 8x8	MUL 16x16	DIV 16x16
Время (max, clocks)	–	–	–	–	75	103	228	263
Размер (words)	–	–	–	–	11	22	17	39

Последовательный периферийный интерфейс (Serial Peripherial Interface – SPI) выполняет две функции. Во-первых, его можно использовать для загрузки ФЛЭШ-ПЗУ программ и данных. Запись может происходить после установки контроллера на печатной плате, при этом требуется только одно питающее напряжение (+5 В), что чрезвычайно упрощает процесс обновления версий программ. Во время штатной работы интерфейс SPI может быть использован для высокоскоростного (до 5 Мбит/с) двунаправленного синхронного обмена данными между контроллером и внешним устройством. В качестве внешнего устройства может выступать, например, последовательная память фирмы ATMEL с интерфейсом SPI – серий AT25XXX или AT45XXX емкостью от 1 кбит до 8 Мбит.

Последовательный асинхронный интерфейс (UART). Стандартный последовательный интерфейс с 8/9-битными данными и скоростью работы от 2400 до 115200 бод, аналогичный имеющемуся в контроллерах x51.

Аналоговый компаратор. Сравнивает два аналоговых сигнала на прямом (AIN1) и инверсном (AIN2) входах. Выход компаратора может устанавливать триггер запоминания (capture) текущего состояния таймера/счетчика. Кроме того, компаратор может вырабатывать отдельный запрос на прерывание.

Аналого-цифровой преобразователь. Это 8-входовый 10-разрядный АЦП со скоростью преобразования 12-13 тысяч выборок в секунду, в зависимости от способа перебора каналов – автоинкрементный или с предварительной установкой номера канала.

Часы реального времени. Используется отдельный осциллятор с частотой 32768 Гц.

СИСТЕМА КОМАНД

Микроконтроллеры AVR используют развитую систему команд. Набор команд для микросхемы AT90S1200 содержит 89 инструкций, для остальных микросхем – 120 инструкций. Команды имеют формат 16 и 32 бита. Большинство из них выполняется за один такт кварцевого генератора. АЛУ выполняет арифметические и логические операции между регистрами или между константой и регистром.

Микроконтроллеры AVR не имеют команды деления. Команда умножения (время выполнения – 2 такта кварцевого генератора) предусмотрена только в старших моделях. Однако, есть подпрограммы умножения и деления 8- и 16- битных чисел, причем существуют варианты, минимизированные как по времени выполнения, так по занимаемому объему памяти программ. Характеристики этих процедур представлены в табл. 2.

СИСТЕМА ПРЕРЫВАНИЙ AVR

Микроконтроллеры AVR имеют развитую систему прерываний.

Прерывания микроконтроллеров tinyAVR и classicAVR:

- внешние прерывания (1 тип у tinyAVR и AT90S1200, 2 типа у остальных classicAVR);
- 1 тип прерывания от Timer 0;
- 4 типа прерываний от Timer1;
- 1 тип прерывания от SPI;
- 3 типа прерывания от UART;
- прерывание от аналогового компаратора

Прерывания микроконтроллеров megaAVR:

- 8 типов внешних прерываний;
- 2 типа прерывания от Timer 0;
- 4 типа прерываний от Timer1;
- 2 типа прерывания от Timer 2;
- 1 тип прерывания от SPI;
- 3 типа прерывания от UART;
- прерывание от аналогового компаратора;
- прерывание по окончании преобразования АЦП;
- прерывание по окончании записи в EEPROM.

СПОСОБЫ АДРЕСАЦИИ ПАМЯТИ

В контроллерах AVR используется пять типов адресации для памяти данных – непосредственная, косвенная, относительная косвенная, относительная предекрементная и относительная постинкрементная.

С переменной в SRAM можно работать с помощью непосредственной адресации, например, следующим образом:

```
.CSEG                ; Кодовый сегмент
                     ; (Program memory)
LDI R16,10           ; R16 = 10
A:                   ; do {
STS VAR,R16          ; VAR = R16
DEC R16              ; R16--
BRNE A               ; } while(R16)

. . .
LDS R24,VAR          ; R24 = VAR
. . .
```

Таблица 3. Команды работы с X-,Y- и Z-регистрами

АДРЕСАЦИЯ / КОМАНДА	R косвенная	R+ постинкрементная	–R предекрементная	R+q относительная
LD & LDD	X,Y,Z	X,Y,Z	X,Y,Z	Y,Z
ST & STD	X,Y,Z	X,Y,Z	X,Y,Z	Y,Z
LPM (ELPM)	Z	–	–	–

```
.DSEG                ; Сегмент данных
                     ; (SRAM)
VAR: .BYTE 1         ; VAR - не «1»,
                     ; а 1 байт!
```

В программной памяти могут храниться константы. Для работы с ними предусмотрена специальная команда LPM (Load Program Memory) – чтение в R0 значения Program Memory из ячейки с адресом, записанным в Z-регистре (R30:R31). Процедура чтения (с помощью косвенной адресации) может выглядеть следующим образом:

```
.DEF OFFL = R29      ; определение
.DEF OFFH = R28      ; регистров
LDI ZH,HIGH(BUF*2)   ; запись в
                     ; Z-регистр
LDI ZL,LOW (BUF*2)   ; адреса BUF в
                     ; байтах
ADD ZL,OFFL          ; добавить
ADC ZL,OFFH          ; смещение
. . .                ; . . .
LPM                  ; R0 = Program
                     ; memory[Z]
. . .                ; . . .
ADIW ZL,1            ; Z = Z+1
LPM                  ; чтение
                     ; следующего байта
. . .                . . .
BUF:                 ; константная
.DB «SAMPLE STRING»  ; строка
```

Этот способ использован для вывода бегущей строки в приводимом ниже примере.

Но главная особенность AVR (в области работы с SRAM) – адресация последней через X, Y и Z-регистры. Это двухбайтовые регистры (Z – R31:R30; Y – R29:R28; X – R27:R26), в которых хранятся адреса для работы с памятью. AVR имеет 23 команды (!) для работы только с этими регистрами (табл. 3).

В качестве примера, иллюстрирующего все преимущества этого подхода, можно рассмотреть т.н. «слияние массивов». Задача заключается в следующем. Имеется два байтовых массива беззнаковых целых чисел (Y и Z) по 10 элементов каждый, отсортированных по возрастанию. Требуется составить из их элементов третий массив из 20 элементов (X) так, чтобы его элементы тоже были отсортированы по возрастанию. Программа, осуществляющая эту процедуру, приведена ниже:

```
unsigned char X[20],Y[10],Z[10];     // Входные
                                     // массивы.
      Sliyanie()                     //
      {                              //
      int XC=0,YC=0,ZC=0;            //
      while(YC < 10 && ZC < 10)      // Пока ни
                                     // один
      if(Y[YC] <= Z[ZC])             // из массивов
                                     // X и Y
      X[XC++] = Y[YC++];             // не кончился
                                     // пишем
      else                           // в X большее
                                     // значение
      X[XC++] = Z[ZC++];             // из их
                                     // «хвостов».
      if(YC == 10)                   // Кончился Y?
      while(ZC != 10)                // дополняем X
      X[XC++] = Z[ZC++];             // элементами
                                     // Z.
      else                           // Кончился Z?
      while(YC != 10)                // дополняем X
      X[XC++] = Y[YC++];             // элементами
                                     // Y.
      }
```

Если писать эту программу на классическом ассемблере для устройства, не имеющего вышеупомянутых специальных средств адресации, придется вручную увеличивать все счетчики и индексы. С использованием же возможностей AVR она будет выглядеть так:

```
.DEF TEMPY = R16     ; Определение
.DEF TEMPZ = R17     ; регистров
.DEF YC  = R18       ;
.DEF ZC  = R19       ;
.CSEG        ;
LDI XL,LOW (XA)      ; Инициализация
LDI XH,HIGH(XA)      ; регистров
LDI YL,LOW (YA)      ; X - адрес XA
LDI YH,HIGH(YA)      ; Y - адрес YA
LDI ZL,LOW (ZA)      ; Z - адрес ZA
LDI ZH,HIGH(ZA)      ;
CLR YC               ;
CLR ZC               ;
ARRNOTEND:           ; Первый цикл
CPI YC,10            ; Кончился Y?
BREQ ARRENDY         ; Да, выход из цикла
CPI ZC,10            ; Кончился Z?
BREQ ARRENDZ         ; Да, выход из цикла
LD  TEMPY,Y          ; Сравниваем
LD TEMPZ,Z           ; «хвосты» Y и Z
CP TEMPZ,TEMPY       ; Z меньше?
BRLO DEPLZ           ; Да, пишем его в X
DEPLY:               ; Иначе
ST X+,TEMPY          ; пишем
ADIW YL,1            ; Y в X
INC YC               ; В начало
RJMP ARRNOTEND       ; цикла.
DEPLZ:               ; Пишем
ST  X+,TEMPZ         ; Z в X
ADIW ZL,1            ;
INC ZC               ; В начало
RJMP ARRNOTEND       ; цикла.
ARRENDZ:             ; Кончился Z:
MOV ZC,YC            ; готовимся к
MOV ZL,YL            ; перезаписи
MOV ZH,YH            ; конца Y в X
ARRENDY:             ; Запись
LD  TEMPZ,Z+         ; конца
ST  X+,TEMPZ         ; одного
INC ZC               ; из
CPI ZC,10            ; массивов
BRNE ARRENDY         ; (Y или Z) в X
FOREVER:             ;
RJMP FOREVER         ; Работа закончена!
```

```
.DSEG                   ; Сегмент данных
XA: .BYTE 20            ; Объявление
YA: .BYTE 10            ; рабочих
ZA: .BYTE 10            ; массивов
```

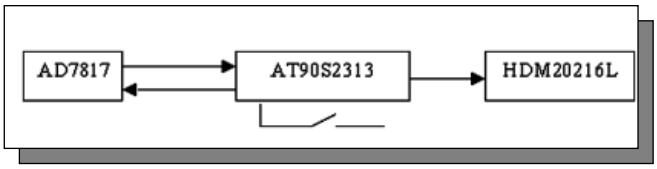
Эта программа не претендует на идеальное построение, но она показывает, как работать с X, Y и Z-регистрами.

ПРИМЕР ИСПОЛЬЗОВАНИЯ МИКРОКОНТРОЛЛЕРА СЕМЕЙСТВА AVR

Приводимый ниже пример – разработка цифрового термометра на базе микроконтроллера AVR – можно использовать в учебных целях для ознакомления с процессом создания систем на основе этих микроконтроллеров.

В состав цифрового термометра входят три устройства – собственно контроллер, в качестве которого была использована микросхема AT90S2313 (хотя здесь можно использовать любую из микросхем серии AT90S), микросхема АЦП с встроенным термодатчиком AD7817 фирмы Analog Devices и двухстрочный цифробуквенный жидкокристаллический индикатор HDM20216L фирмы HANTRONIX. Блок-схема прибора очевидна и приведена на рис. 2.

Рисунок 2. Цифровой термометр на основе AVR



Несколько слов об используемых компонентах.

Микросхема AD7817 содержит на кристалле следующие узлы:

- 10-битный АЦП с входным диапазоном измеряемых напряжений от 0 В до + 2,5 В;
- 4-канальный входной мультиплексор;
- внутренний источник опорного напряжения (+1,23V);
- термодатчик, работающий в диапазоне температур от -55°С до +125°С;
- последовательный интерфейс SPI для работы с контроллером.

Время преобразования при оцифровке внешнего сигнала составляет 9 мкс, при измерении температуры – 27 мкс. Питающее напряжение – 2,7 … 5,5 В, потребляемая мощность 6 мВт в активном режиме, 6 мкВт в режиме ожидания.

Индикатор HDM20216L имеет две строки по 20 символов размером 5x7 точек со встроенным знакогенератором. Интерфейс реализован на микросхеме HD44780. Использован однонаправленный обмен с 8-битной шиной данных.

Ниже приведены исходные тексты программы цифрового термометра.

Размер кодового файла, полученного в результате трансляции программы – чуть меньше 1000 байт, из которых почти 300 – символьные строки.

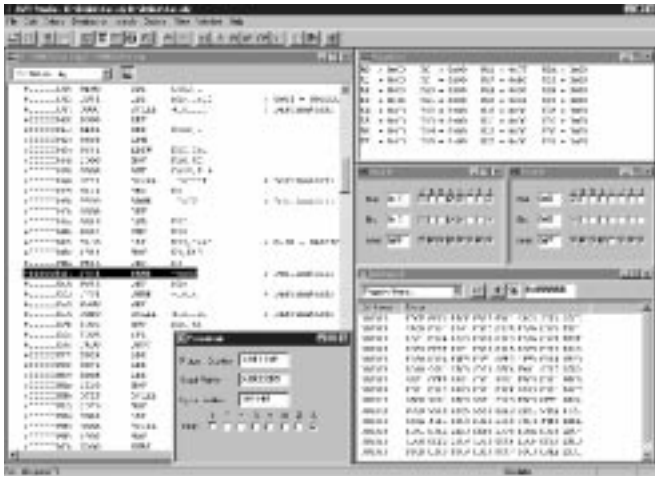
Для компиляции исходного файла использовался ассемблер WAVRASM , бесплатно распространяемый фирмой ATMEL. Его сновные характеристики:

– исключительно быстрое ассемблирование;
– поддержка всех микроконтроллеров семейства AVR;
– мощные возможности по использованию макрокоманд;
– поддержка всех стандартных выходных форматов;
– просмотр предыдущей/следующей ошибки компиляции;
– легкий в использовании интерфейс MS Windows;
– встроенный текстовый редактор;
– существование версии ассемблера, функционирующей в MS-DOS.

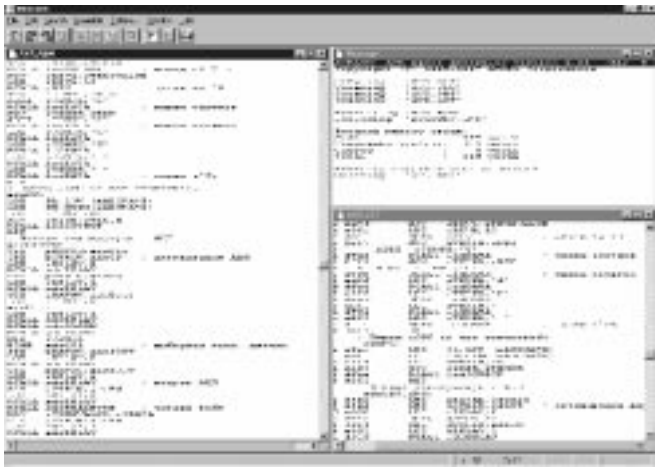
Для отладки программы использовался пакет AVR-Studio, также поставляемый бесплатно и являющийся полной и более удобной заменой другого отладочного средства – AVR Simulator.

Исходные тексты программы цифрового термометра приведены на стр. 36–37.

Пример интерфейса программы AVR-Studio.



Пример интерфейса программы WAVRASM.



Данный пример показывает, что разработка конкретных устройств на базе микроконтроллеров AVR не является сложным делом, а удобные и легкие в изучении средства программирования делают процесс разработки увлекательным занятием. Дополнительным фактором, способствующим переходу на новую элементную базу, является наличие десятка разнообразных микросхем с единой системой команд, что позволяет оптимально строить разрабатываемую систему, а в дальнейшем достаточно просто ее модифицировать, приспосабливая к изменившимся требованиям заказчика.

Н.В. Королев, Д.Н. Королев
Фирма **«АРГУССОФТ Компани»,**
официальный дистрибьютор фирмы ATMEL в России.
Тел. (095)-215-9110, 215-9556, 216-5729, 216-5929
e-mail: components@argussoft.ru
Internet: www.argussoft.ru

```
; *** ЦИФРОВОЙ ТЕРМОМЕТР на AT90S2313***

        .INCLUDE "2313DEF.INC"
        ; Основные определения
        .EQU    pDATAW    =PORTB   ; определение
        .EQU    pDATAR    =PINB    ; портов
        .EQU    pDATAD    =DDRB    ; и
        .EQU    pCTRLW    =PORTD   ; битов
        .EQU    pCTRLR    =PIND
        .EQU    pCTRLD    =DDRD

        .EQU    lcdE      =0
        .EQU    lcdRS     =1

        .EQU    adcSTRT   =1       ; порт B
        .EQU    adcRW     =4       ; порт D
        .EQU    adcCLK    =3       ; порт D
        .EQU    adcDout   =2       ; порт B
        .EQU    adcDin    =3       ; порт B
        .EQU    adcCS     =2       ; порт D

        .DEF    rTEMPL    =R1      ; временные
        .DEF    rTEMPH    =R16     ; регистры
        .DEF    rTEMP2    =R17

        .DEF    rZDATA    =R0      ; для LPM

        .DEF    rSIZE     =R2      ; длина строки
        .DEF    rLOWD     =R3      ; для задержки

        .DEF    rDELAY    =R19     ; для задержки

        .DEF    rTEMPVALUE=R6      ; температура
        .DEF    rDIV1     =R23     ; для
        .DEF    rDIV2     =R24     ; процедуры
        .DEF    rRES      =R23     ; деления
        .DEF    rREM      =R25
        .DEF    rDIVCNT   =R26

        .DEF    rOFFL     =R28     ; положение
        .DEF    rOFFH     =R29     ; бег. строки

        .EQU    vSTREND   =250     ; признак конца
                                   ; строки
        ; Макрокоманды
        .MACRO ADDI                ; добавить
        SUBI @0,-@1                ; к регистру
        .ENDMACRO                  ; константу

        .MACRO INCW                ; инкрементировать
        ADIW @0,1                  ; слово
        .ENDMACRO                  ; (Z-регистр)

        ; Кодовый сегмент
        LDI     rTEMPH,LOW(RAMEND) ; иниц. стека
        OUT     SPL,rTEMPH

        SER     rTEMPH             ; объявление
        OUT     pDATAD,rTEMPH      ; портов B и D
        OUT     pCTRLD,rTEMPH      ; выходными

        SBI     pCTRLW,adcCS       ; дезактив. АЦП

        CLR     rOFFH              ; сброс
        CLR     rOFFL              ; регистров

        SBI     pCTRLW,lcdE        ; иниц. ЖКИ
        LDI     rDELAY,1
        RCALL   lcdDELAY
        LDI     rTEMPH,$0C         ; Display ON
        RCALL   lcdCOMMAND         ; Cursor OFF
        LDI     rDELAY,10
        RCALL   lcdDELAY
        LDI     rTEMPH,$01         ; Clear display
        RCALL   lcdCOMMAND
        LDI     rDELAY,10
        RCALL   lcdDELAY
        LDI     rTEMPH,$38         ; Function set
        RCALL   lcdCOMMAND         ; 5x7 matrix
        LDI     rDELAY,10
        RCALL   lcdDELAY

lFOREVER:
        SERrDELAY                  ; большая
        RCALL   lcdDELAY           ; задержка
        SER     rDELAY
        RCALL   lcdDELAY
        LDI     rTEMPH,$80         ; 1я строка:
        RCALL   lcdCOMMAND
        RCALL   lcdOUTTEMP         ; вывод темп.
        LDI     rTEMPH,$C0         ; 2я строка:
        RCALL   lcdCOMMAND
        LDI     rDELAY,32
        RCALL   lcdDELAY           ; загрузка адреса
        LDI     ZL,LOW (sSECOND*2)
        LDI     ZH,HIGH(sSECOND*2)
        ADD     ZL,rOFFL           ; сдвиг
        ADC     ZH,rOFFH
        LDI     rTEMPH,20          ; 20 символов
        MOV     rSIZE,rTEMPH
        RCALL   lcdOUTBUF          ; вывод
        INCW    ZL
        LPM
        MOV     rTEMPH,rZDATA      ; если дальше
        CPI     rTEMPH,vSTREND     ; STREND
        BRNE    lRUNMORE
        CLR     rOFFL              ; «бежим»
        CLR     rOFFH              ; сначала
lRUNMORE:
        INCW    rOFFL,1            ; увел. смещения
        RJMP    lFOREVER
lcdCOMMAND:                        ; Послать командный
                                   ; байт
        CBI     pCTRLW,lcdRS
        RJMP    lcdOUT
lcdDATA:                           ; Послать байт данных
        SBI     pCTRLW,lcdRS
        ; Послать байт на ЖКИ
lcdOUT:
        OUT     pDATAW,rTEMPH
        RCALL   lcdPULSE
        RET
        ; Импульс записи
lcdPULSE:
        LDI     rDELAY,1
        RCALL   lcdDELAY
        CBI     pCTRLW,lcdE        ; E = 0
        LDI     rDELAY,1
        RCALL   lcdDELAY
        SBI     pCTRLW,lcdE        ; E = 1
        LDI     rDELAY,1
        RCALL   lcdDELAY
        RET
        ; Вывод rSIZE символов на ЖКИ
lcdOUTBUF:
        SBI     pCTRLW,lcdRS
lOUT:
        LPM
        INCW    ZL
        MOV     rTEMPH,rZDATA
        OUT     pDATAW,rTEMPH
        RCALL   lcdPULSE
        DEC     rSIZE
        BRNE    lOUT
        RET
        ; Вывод температуры
lcdOUTTEMP:
        RCALL   adcGETTEMP         ; чтение температуры
```

```
MOV     rTEMPH,rTEMPVALUE
CPI     rTEMPH,100          ; если больше 100
BRSH    adcNC               ; АЦП не подключен
LDI     ZL,LOW (sFIRST*2)
LDI     ZH,HIGH(sFIRST*2)
LDI     rTEMPH,14
MOV     rSIZE,rTEMPH
RCALL   lcdOUTBUF           ; вывод «T = «
MOV     rDIV1,rTEMPVALUE
LDI     rDIV2,10
RCALL   pDIV                ; делим на 10
MOV     rTEMPH,rRES
ADDI    rTEMPH,'0'
RCALL   lcdDATA             ; пишем частное
MOV     rTEMPH,rREM
ADDI    rTEMPH,'0'
RCALL   lcdDATA             ; пишем остаток
LDI     rTEMPH,'Я'
RCALL   lcdDATA
LDI     rTEMPH,'C'
RCALL   lcdDATA
LDI     rTEMPH,' '
RCALL   lcdDATA
LDI     rTEMPH,' '
RCALL   lcdDATA             ; пишем «'C»
RET
; Пишем «ADC is not connected»
adcNC:
LDI     ZL,LOW (sADCNA*2)
LDI     ZH,HIGH(sADCNA*2)
LDI     rTEMPH,20
MOV     rSIZE,rTEMPH
RCALL   lcdOUTBUF
RET
; Чтение температуры с АЦП
adcGETTEMP:
CBI     pDATAD,adcDin
CBI     pCTRLW,adcCS        ; инициализация АЦП
LDI     rDELAY,2
RCALL   adcDELAY
CBI     pCTRLW,adcRW
LDI     rDELAY,1
RCALL   adcDELAY
CBI     pDATAW,adcDout
LDI     rTEMPH,8
adcGT1:
LDI     rDELAY,1
RCALL   adcDELAY
RCALL   adcCLOCK
RCALL   adcDELAY
DEC     rTEMPH              ; канал 0 -
BRNE    adcGT1              ; темп. датчик
SBI     pDATAW,adcSTRT
LDI     rDELAY,2
RCALL   adcDELAY
CBI     pDATAW,adcSTRT
LDI     rDELAY,2
RCALL   adcDELAY            ; запуск АЦП
SBI     pCTRLW,adcRW
LDI     rDELAY,8
RCALL   adcDELAY
RCALL   adcREADBYTE         ; читаем байт
MOV     rTEMPVALUE,rZDATA
CBI     pCTRLW,adcRW
LDI     rDELAY,1
RCALL   adcDELAY
SBI     pCTRLW,adcCS
SBI     pDATAD,adcDin
MOV     rTEMPH,rTEMPVALUE   ; температура
SUBI    rTEMPH,103          ; = 103 -
MOV     rTEMPVALUE,rTEMPH   ; считанный байт
RET
; Импульс чтения/записи для АЦП
adcCLOCK:
LDI     rDELAY,1
RCALL   adcDELAY
SBI     pCTRLW,adcCLK
LDI     rDELAY,1
RCALL   adcDELAY
CBI     pCTRLW,adcCLK
LDI     rDELAY,1
RCALL   adcDELAY
SBI     pCTRLW,adcCLK
LDI     rDELAY,1
RCALL   adcDELAY
RET
; Чтение байта с АЦП
adcREADBYTE:
CLR     rTEMPH
MOV     rZDATA,rTEMPH
LDI     rTEMPH,8
adcRB1:                     ; 8 раз
LSL     rZDATA
RCALL   adcRtoT
BLD     rZDATA,0
DEC     rTEMPH              ; читаем бит
BRNE    adcRB1              ; и сдвигаем влево
RET
; Чтение бита из АЦП в T
adcRtoT:
LDI     rDELAY,1
RCALL   adcDELAY
SBI     pCTRLW,adcCLK
LDI     rDELAY,1
RCALL   adcDELAY
CBI     pCTRLW,adcCLK
LDI     rDELAY,1
RCALL   adcDELAY
IN      rTEMPH,pDATAR
BST     rTEMPH,adcDin
RET
; Беззнаковое деление 8/8
pDIV:
SUB     rREM,rREM
LDI     rDIVCNT,9
D1:
ROL     rDIV1
DEC     rDIVCNT
BRNE D2
RET
D2:
ROL     rREM
SUB     rREM,rDIV2
BRCC    D3
ADD     rREM,rDIV2
CLC
RJMP    D1
D3:
SEC
RJMP    D1
; Текстовые строки
sFIRST:
.DB "Temperature = "
sADCNA:
.DB "ADC is not connected"
sSECOND:
.DB "                    "
.DB ">>> ARGUSSOFT COMPANY <<<"     ; бегущая
                                    ; строка
.DB "Working ATMEL AVR AT90S2313"
.DB "with HANTRONIX LCD indicator"
.DB "and ANALOG DEVICES ADC AD7817"
.DB "with on-chip temperature sensor."
.DB "Ask for more information in"
.DB "ARGUSSOFT COMPANY."
sCLEAR:
.DB "                    "
sEND:
.DB vSTREND
```