

AVR: программирование на языке Си в среде ICCAVR фирмы ImageCraft

Многие российские и зарубежные разработчики применяют в своих проектах AVR-микроконтроллеры, которые фирма ATMEL выпускает с 1997 г. За это время продано уже более 500 миллионов штук микроконтроллеров. Одним из факторов такой популярности является удачная архитектура микросхем, которая оптимизирована для программирования на языке Си. Критики этого утверждения могли заметить, что, младшие представители AVR-микроконтроллеров (ATtiny11, ATtiny12 и ATtiny15), не содержат блок оперативной памяти, и оснащены аппаратным трехуровневым стекком, существенно ограничивающим возможности программиста. Однако, с выходом AVR-микроконтроллера второго поколения ATtiny13 это «узкое место» было устранено. 8-выводной ATtiny13 оснащен модулем оперативной памяти, позволяющий создавать программный стек заданной глубины. Таким образом, полноценное программирование на языке Си стало возможным и для этих микроконтроллеров.

В данной статье рассматривается Си-компилятор фирмы ImageCraft Creations Inc.

Дистрибутив Си-компилятора фирмы ImageCraft Creations Inc. весьма компактен и занимает немногим более 5 Мбайт. Номер версии на момент выхода статьи – 6.31. Этот компилятор весьма дружелюбен к пользователям. После первой инсталляции на компьютере он предоставляет пользователю работать с ним в течение 45 дней и создавать приложения объемом до 64 Кбайт. По истечении указанного срока компилятор переходит в демо-режим, при этом максимальный объем выходного файла ограничивается размером 4 Кбайта. Существует два варианта поставки компилятора – стандартная версия и профессиональная версия. Профессиональная версия имеет следующие отличия:

- поддерживается создание проектов с объемом исполняемого файла до 128 Кбайт (стандартная версия имеет максимальный размер выходного файла 64 Кбайт);
- в профессиональную версию включен оптимизатор кода, уменьшающий размер файла на 8 – 15%;
- поддерживается работа со структурами при отладке проекта в среде AVR Studio.

Типы файлов

В работе компилятора используются следующие типы файлов (по расширениям):

C – исходный текст на языке Си.
S - исходный текст на ассемблере.
H – заголовочный (header) файл.
PRJ – файл проекта.

SRC - список файлов проекта.

S – выходной ассемблерный файл, генерируется для каждого исходного Си-файла.

O – объектный файл, получаемый после компиляции ассемблерного файла.

HEX – выходной файл в формате Intel HEX для загрузки в ПЗУ программ микросхемы.

EEP - выходной файл в формате Intel HEX для загрузки в ПЗУ данных микросхемы.

COF - выходной файл в формате COFF, используется при отладке проекта в AVR Studio.

LST – файл-листинг, содержащий информацию об адресах.

MP – MAP-файл, содержащий символическую информацию.

DBG – файл с отладочной информацией.

A - библиотечный файл.

Дистрибутив содержит более десяти библиотек, в число которых входят библиотека стандартного ввода/вывода, библиотека поддержки вычислений с «плавающей точкой», строковые функции, работа с памятью и т.д. Базовой библиотекой, состоящей из стандартной библиотеки языка Си и дополненной расширениями, специфическими для архитектуры AVR, является Libcavr.a. К специфичным функциям архитектуры AVR, помимо богатого набора аппаратных интерфейсов, также относится возможность удаленного автообновления ПЗУ программ, используя автозагрузчик (bootloader). Компилятор также содержит средства

создания и модификации пользовательских библиотек.

Из перечисления типов файлов можно увидеть, что проект может содержать

несколько исходных файлов, причем часть файлов может быть на ассемблере. Попутно отметим, что можно использовать ассемблерные вставки с очевидным синтаксисом `asm("<string>")`.

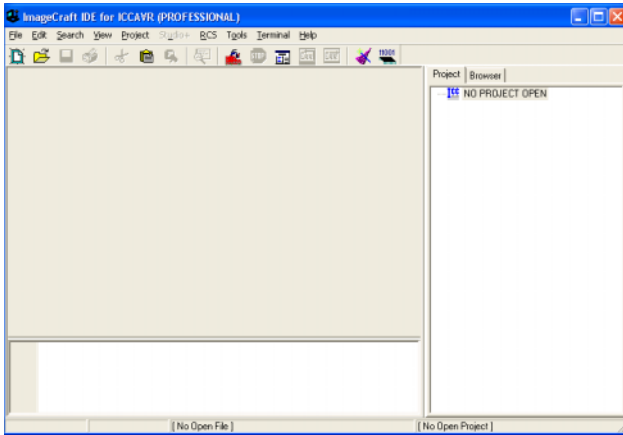


Рис.1

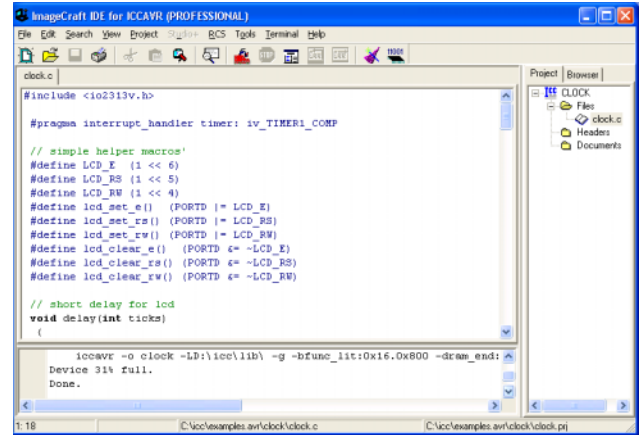


Рис.2

Компиляция проекта

Рассмотрим пример создания простого проекта, состоящего из одного исходного файла. При установке компилятор по умолчанию использует каталог `c:\icc`. В каталоге `c:\icc\examples.avr` находятся несколько примеров программ, которые можно использовать, как учебные. Для определенности выбран файл `clock.c`. Эта программа выполняет подсчет временного интервала и выводит текущее время на ЖК-индикатор. Ниже приведен полный текст программы:

```
#include <io2313v.h>

#pragma interrupt_handler timer:
iv_TIMER1_COMP

#define LCD_E (1 << 6)
#define LCD_RS (1 << 5)
#define LCD_RW (1 << 4)
#define lcd_set_e() (PORTD |= LCD_E)
#define lcd_set_rs() (PORTD |= LCD_RS)
#define lcd_set_rw() (PORTD |= LCD_RW)
#define lcd_clear_e() (PORTD &= ~LCD_E)
#define lcd_clear_rs() (PORTD &= ~LCD_RS)
#define lcd_clear_rw() (PORTD &= ~LCD_RW)

// short delay for lcd
void delay(int ticks)
{
    while(ticks--);
}

// lcd strobe
void lcd_pulse(void)
{
    lcd_set_e();
    delay(4);
    lcd_clear_e();
    delay(4);
}
```

```
// medium delay (long for lcd, but much less
than a second)
void lcd_wait(void)
{
    delay(1000);
}

// send byte to lcd
void lcd_send(unsigned char data)
{
    lcd_wait();
    PORTB = data;
    lcd_pulse();
}

// clear screen
void clrscr(void)
{
    lcd_clear_rs();
    lcd_clear_rw();
    lcd_send(0x01);
    lcd_wait();
}

// init display
void initlcd(void)
{
    DDRB = 0xFF;
    DDRD |= (LCD_E | LCD_RS | LCD_RW);
    lcd_clear_rs();
    lcd_clear_rw();
    lcd_send(0x3C);
    lcd_send(0x3C);
    lcd_send(0x3C);
    lcd_send(0x06);
    lcd_send(0x0C);
}

// goto lcd memory address
void gotoz(unsigned char z)
{
    lcd_clear_rs();
    lcd_clear_rw();
    lcd_send(z | 0x80);
}

#define gotoxy(x,y) gotoz((x)|(y)<<6)

// output single character
void putchar(char c)
{

```

```

lcd_clear_rw();
lcd_set_rs();
lcd_send(c);
}

// output string
void outtext(char* text)
{
    unsigned char i;
    for(i = 0; text[i] && i < 16; i++)
        putchar(text[i]);
}

unsigned char hour = 0, minute = 0, second = 0;

// call one time per second
void timer(void)
{
    // first, output current time
    clrscr();
    gotoxy(0,0);
    putchar('0'+hour/10);
    putchar('0'+hour%10);
    putchar(':');
    putchar('0'+minute/10);
    putchar('0'+minute%10);
    putchar(':');
    putchar('0'+second/10);
    putchar('0'+second%10);
    // then increment counter
    second++;
    if(second == 60)
    {
        second = 0;
        minute++;
        if(minute == 60)
        {
            minute = 0;
            hour++;
            if(hour == 24)
            {
                hour = 0;
            }
        }
    }
}

// 'main' is declared as 'int' to be
compliant with ANSI-C
int main(void)
{
    TIMSK = (1<<6); // set OCIE1A
    TCCR1A = 0;
    TCCR1B = 0x0C; // CTC1, CK/256
    OCR1H = 0x3D; // 400000/256=15625=0x3D09
    OCR1L = 0x09;
    TCNT1H = TCNT1L = 0;
    initlcd();
    timer();
    SREG = 0x80; // SEI
    return 0;
}

```

После запуска пакета на экране можно видеть следующее окно (рис.1). Рабочее пространство разделено на три части:

- слева – область для просмотра и редактирования исходных файлов;
- справа – «список файлов проекта» с перечнем всех файлов проекта,
- внизу – «окно сообщений», где отображается информация о ходе компиляции проекта и выдаются сообщения об ошибках.

При первом запуске все эти области – пустые.

Сначала следует создать проект командой Project->New, где указывается имя проекта и путь к файлу проекта (рис.2). Теперь надо подключить файл с исходным кодом. Можно написать исходный код непосредственно в окне редактора. Редактор имеет богатый выбор опций настройки, включая выбор русского языка (рис.3).

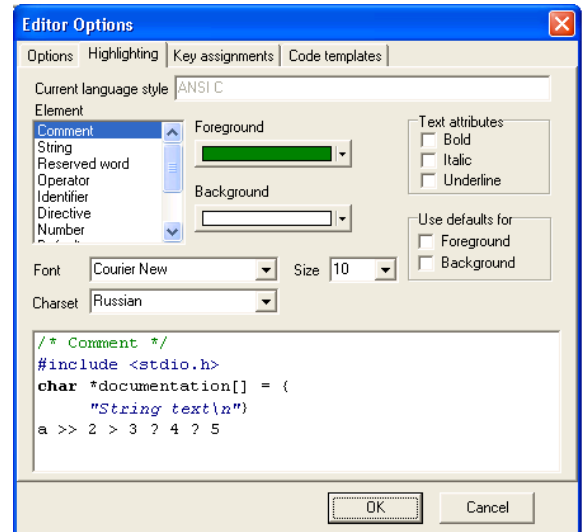


Рис.3

Если файл уже существует, проще всего его добавить в проект, нажав контекстную кнопку мыши на слове Files в поле «список файлов проекта». Содержимое исходного файла будет показано в левом окне, а в нижней части экрана отображаются имена и пути исходного файла и файла проекта. Далее, по команде Project->Options->Target нужно установить тип микросхемы (AT90S2313) (рис.4).

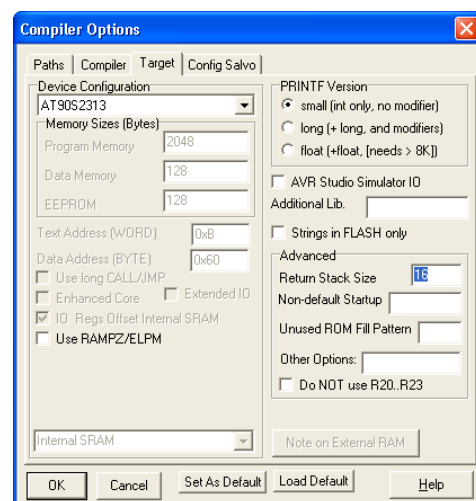


Рис.4

В окне Project->Options->Compiler задается тип выходного файла (рис.5).

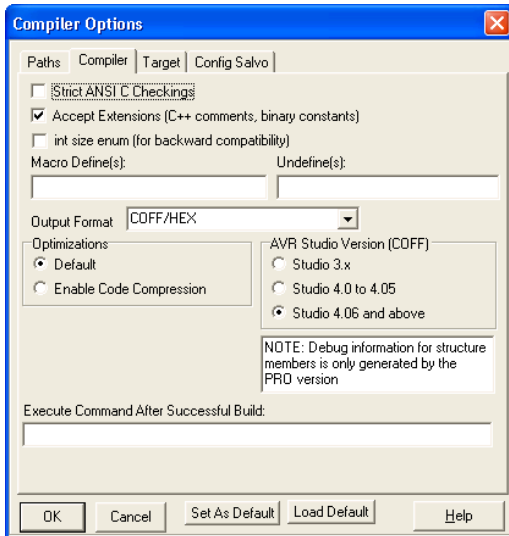


Рис.5

По умолчанию компилятор генерирует два или три файла, один в формате HEX, который используется программатором для загрузки исполняемого кода в микросхему, и второй - в формате COFF, который используется пакетом AVR Studio для отладки программы. Третий файл, с расширением .EEP, содержит образ ПЗУ данных, если это ПЗУ используется в проекте. По команде Project->Make Project (или клавише F9) проект запускается на компиляцию. Результат процесса компиляции показан на рис.6.

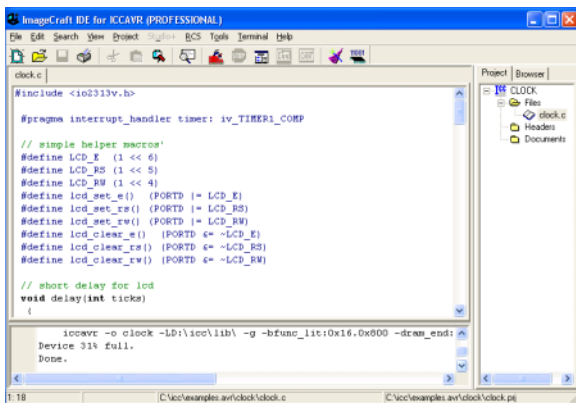


Рис.6

В окне сообщений можно видеть, что компиляция прошла успешно, и полученный файл занимает 31% памяти программ микросхемы AT90S2313.

В состав ICC AVR входит полезный модуль Application Builder, существенно упрощающий рутинную работу по инициализации микроконтроллера. ICC AVR

Application Builder берет на себя инициализацию портов ввода-вывода, аналогово-цифрового преобразователя, компаратора, таймеров/счетчиков, внешних интерфейсов SPI, USART и TWI (аналог I2C), а также базовое распределение памяти и необходимые прерывания – все процедуры, которые обычно занимают много времени программиста, и ошибки в которых трудно найти на стадии разработки. На рис. 7 показано окно Application Builder с основными опциями для микросхемы ATtiny2313, которая приходит на смену AT90S2313.

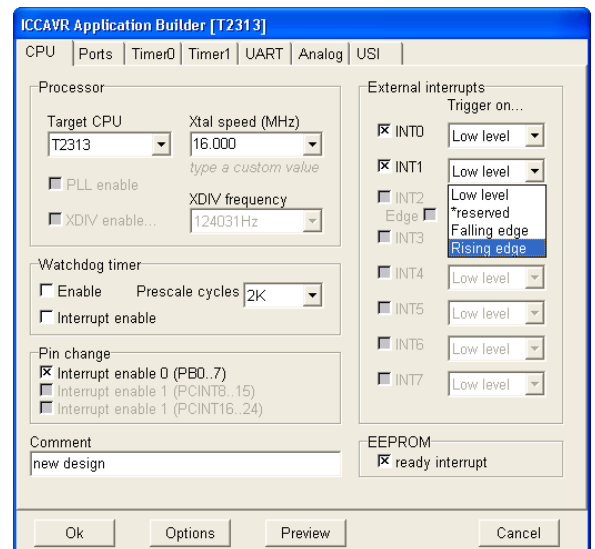


Рис.7

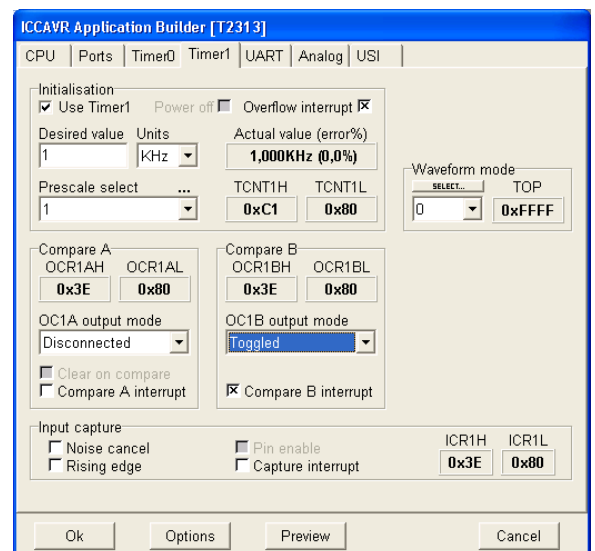


Рис.8

На рис. 8 представлено окно Application Builder с опциями настройки таймера этой микросхемы. Очевидно, что ручная инициализация такого количества параметров займет немало времени.

Компилятор снабжен хорошо структурированной help-системой рис.9, которая существенно облегчает процесс освоения этого пакета.

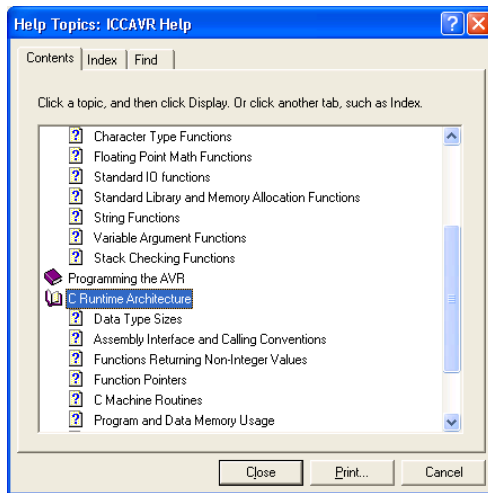


Рис.9

Отладка проекта

Теперь можно полученный HEX-файл при помощи программатора загрузить в микросхему и убедиться в правильном выполнении программы рис.10.

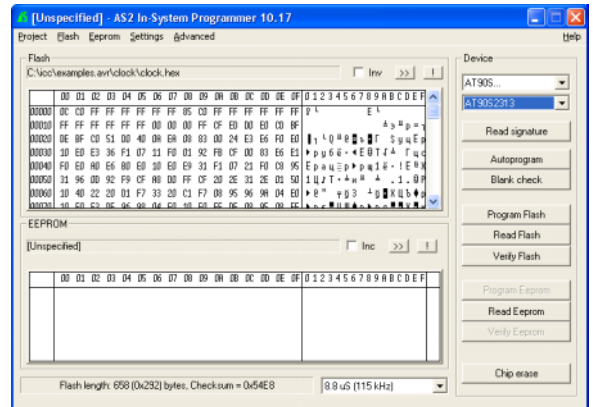


Рис.10

Если программа работает корректно, работу с проектом можно считать завершенной. Однако, эта ситуация характерна для небольших проектов, обычно программа с первого раза «не запускается». В таком случае простейшая отладка может производиться многократной перекомпиляцией и перепрошивкой модифицированного файла в микросхему. Эта процедура занимает от нескольких секунд до нескольких минут. Серьезные проекты, как правило, требуют отладки (симуляции и эмуляции) с применением пакета AVR Studio (рис. 11) и внутрисхемного эмулятора ATJTAGICE2.

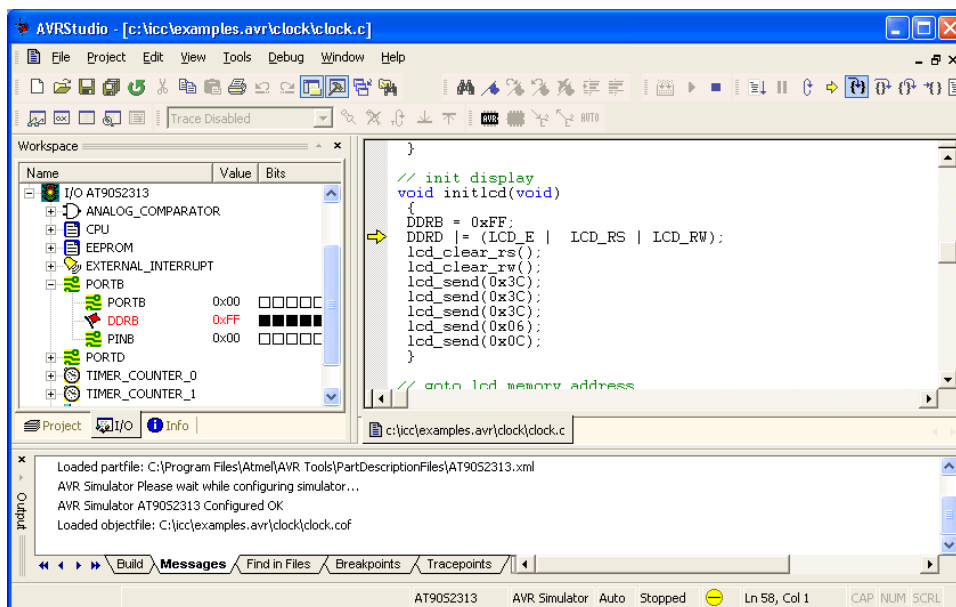


Рис.11

Практическое использование пакета ICCAVR

Современные системы на базе AVR-микроконтроллеров используют разнообразные интерфейсы для обмена информацией с внешним миром. В последнее время

приобретают популярность разработки с использованием интерфейса Ethernet 10/100 Мбит, и протокола обмена TCP/IP. Этот интерфейс сочетает два важных параметра – высокую скорость обмена и гальваническую развязку между контроллером и основной управляющей системой. Существует несколько

технических решений AVR+ Ethernet. Если нужна высокая скорость обмена и минимальная загрузка AVR-контроллера, можно использовать микросхему W3100A фирмы WIZnet. Преимущество такого решения – аппаратная реализация TCP/IP стека, который встроены в микросхему W3100A. Недостаток – относительно высокая стоимость, так как помимо собственно микросхемы W3100A нужна внешняя микросхема для «физического уровня».

Однако, в ряде задач скорости обмена 10 Мбит вполне достаточно, а на одном из первых мест стоит стоимость системы. В этом случае оптимальным является использование микросхемы RTL8019AS фирмы REALTEK, которая стоит менее 5 долларов. Эта микросхема разработана как однокристалльный Ethernet-контроллер для шины ISA, тем не менее она выпускается в настоящее время и является наиболее дешевым решением для создания контроллера и интерфейсом Ethernet. Именно такая конфигурация применяется в стартовом наборе – плате AS-mega2, разработанной специалистами АРГУССОФТ Компани.

Оборотная сторона «дешевого решения» – отсутствие аппаратного TCP/IP стека. И здесь на помощь приходит Си-компилятор ICCAVR. Дело в том, что уже есть такие программы, написанные для микроконтроллера ATmega128 именно в среде ICCAVR. Общий объем кода, который помимо TCP/IP реализует еще протоколы UDP, DHCP и другие, составляет около 60 Кбайт, а если ограничиться только TCP/IP, тогда размер кода не превысит 30 – 35 Кбайт, и контроллер можно реализовать на микросхеме Atmega64 стоимостью менее 5 долларов. Таким образом, можно реализовать, например, дешевую систему сбора данных, имеющую быстрый интерфейс с гальванической развязкой.

Во время написания этой статьи авторы получили информацию от фирмы ImageCraft о

скором выпуске следующей, седьмой версии Си-компилятора ICCAVR. В будущей версии создатели компилятора появятся следующие отличия.

- MIO (Machine Independent Optimizer) – процессорно-независимый оптимизатор. Глобальная оптимизация может существенно сократить размер кода и время выполнения программы. Предполагается, что размер кода может быть уменьшен на 10 – 20% по сравнению с использованием традиционной оптимизации
- C99 Compiler – будет добавлена поддержка стандарта C99, включающего поддержку комплексных чисел, типа данных long long и т.д.
- EC++ Compiler (Embedded C++) – встроенный C++. После добавления в Си-компилятор фирмы ImageCraft стандарта C99 он будет развиваться в сторону полной поддержки встроенного C++, который является наиболее используемым подмножеством языка C++, предназначенного для разработки встраиваемых систем.

Описанные технологии являются настолько новыми, что они даже не объявлены другими фирмами-создателями Си-компиляторов для 8/16 битных процессоров, и это подчеркивает передовые позиции фирмы ImageCraft.

Информацию по приобретению Си-компилятора ICCAVR, техническую консультацию по применению этого компилятора, а также примеры программ можно получить на сайте официального дистрибьютора фирмы ImageCraft Creations Inc. в России, АРГУССОФТ Компани <http://atmel.argussoft.ru>.

Литература

1. Технические материалы фирмы ImageCraft Creations Inc.
2. Николай Королев, Дмитрий Королев AVR: программирование в среде AVR Studio. // Компоненты и технологии, 2004 № 3
3. Николай Королев, Дмитрий Королев AVR-микроконтроллеры второго поколения: средства разработчика. // Компоненты и технологии, 2003 № 7.
4. Николай Королев, Дмитрий Королев AVR-микроконтроллеры второго поколения: новые аппаратные возможности. // Компоненты и технологии, 2003 № 4.