AVR031: Getting Started with ImageCraft C for AVR

Features

- **How to Open a New Project**
- **Description of Option Settings**
- **Writing and Compiling the C Code**
- **How to Load the Executable File into the STK200 Starter Kit**

Introduction

The purpose of this application note is to guide new users through the initial settings of the ImageCraft IDE and compile a simple C program. The application note shows how to set up the compiler to generate an executable hex file and how to download this file into the device. The example described in this application note is written for the AT90S2313 using the STK200 starter kit, or alternatively, an emulator.

Preparations

You may download and install the full-featured 30-day demo program from the ImageCraft web site (http://www.imagecraft.com), or install the product directly from the ImageCraft product CD. To turn the demo program into a licensed program, all you need is an unlock disk or unlock code from ImageCraft. In this document, we will assume that you install the toolset into the default directory “c:\icc”. Once installed, a program shortcut “ICC AVR” is placed into the start menu program group “ImageCraft Development Tools”.

Creating a New Project

Invoke the program “ICC AVR”. The IDE is divided into three areas: “editors”, “project file list”, and “status window”. When you first start the program, all three areas are empty, but a typical use may result in something that looks like Figure 1. The “editors” section uses a separate “tab” for each syntax aware editor window. The “project file list” stores all the files in a project. The “status window” displays any messages resulting from building the project. You may choose not to view the project file list or the status window in order to maximize the screen space devoted to the editors.

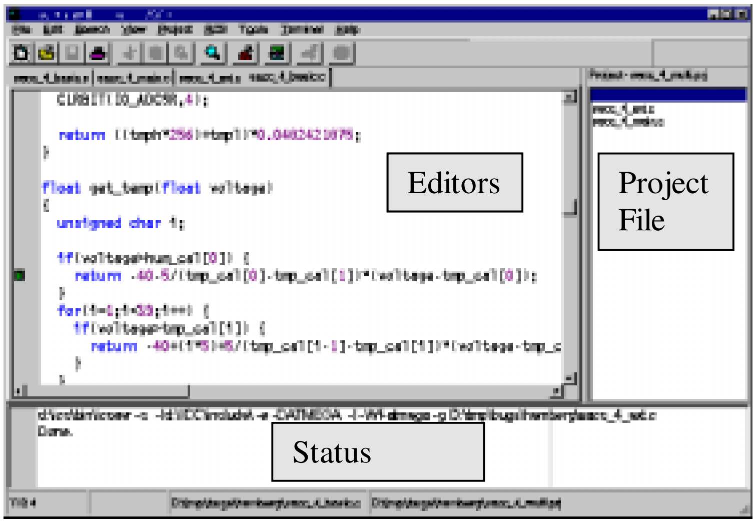
AVR®

8-bit RISC Microcontroller

Application Note

Rev. 1630A–01/00

Figure 1. The IDE

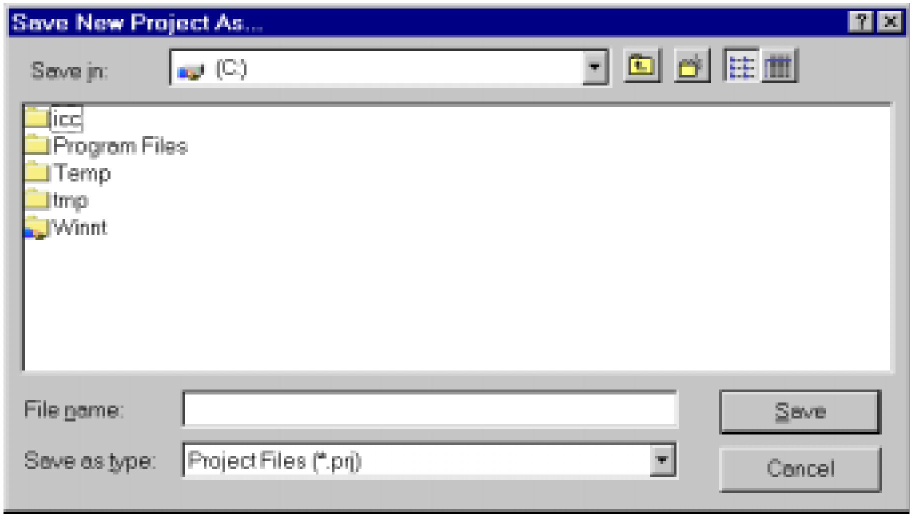


```
  CLRBIT(IO_ADCSR,4);

  return ((tmph*256)+tmpl)*0.0482421875;
}

float get_temp(float voltage)
{
  unsigned char i;

  if(voltage>tmp_cal[0]) {
    return -40-5/(tmp_cal[0]-tmp_cal[1])*(voltage-tmp_cal[0]);
  }
  for(i=1;i<33;i++) {
    if(voltage>tmp_cal[i]) {
      return -40+(i*5)+5/(tmp_cal[i-1]-tmp_cal[i])*(voltage-tmp_c
    }
  }
```

```
Done.
```

To create a new test project, select “Project>New”. The dialog box shown in Figure 2 appears. In this dialog box, first create a folder called “c:\icctest” and then type “example” in the “File name” window. The name of your project is now “example” and “c:\icctest\example” is your project directory.

Figure 2. New Project Dialog Box



Save New Project As...

Save in: (C:)

icc
Program Files
Temp
tmp
Winnt

File name:

Save as type: Project Files (*.prj)

Settings in “Project>Options”

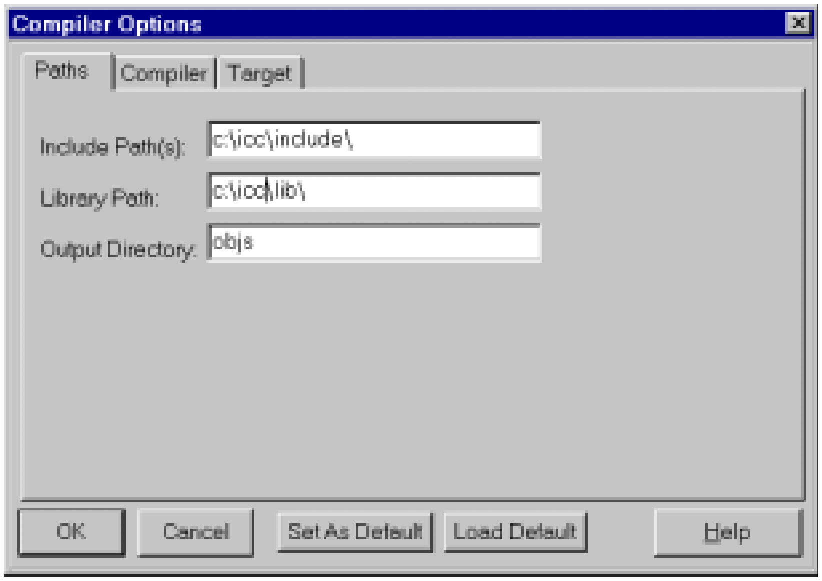
Before any code can be compiled and linked, you must set up the options for the compiler correctly. You select “Project>Options” to display a dialog box to change the settings. Each project can have its own settings and you can store a particular group of settings as your default settings. Default settings can be loaded into a project. There are three pages in the “Project Options” dialog box. For this document, we will only look at options that are relevant to our example.

Paths

In the “Paths” page of the dialog box, you specify the directory paths to the header files, the libraries, and the output directory. The include paths and library path should be automatically set up to “c:\icc\include” and “c:\icc\lib” respectively.

The output directory is where the compiler should put the output and project related files. This path will be relative to your project directory unless you specify an absolute directory. Enter “objs” in the edit box. The IDE will use “c:\icctest\example\objs” as the output directory.

Figure 3. “Paths” Options



Include Path(s): c:\icc\include\

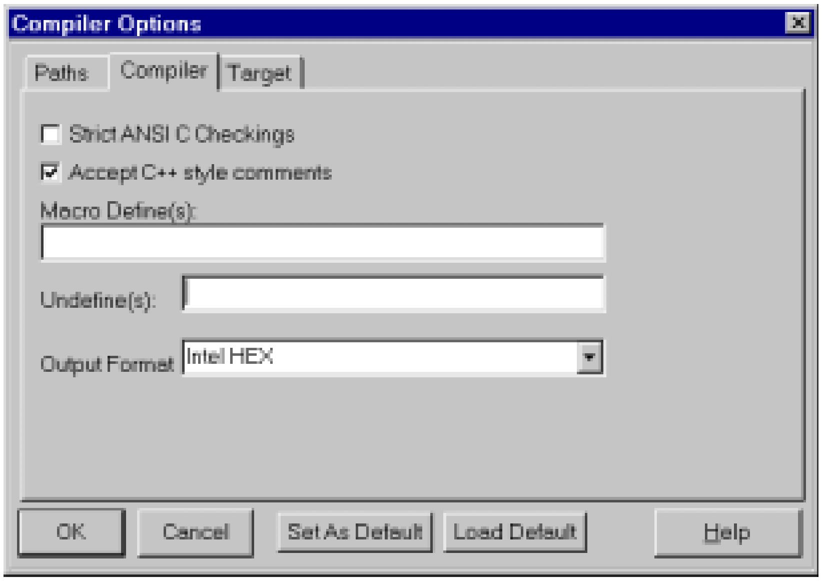
Library Path: c:\icc\lib\

Output Directory: objs

Compiler

In the “Compiler” page of the dialog box, you specify options that affect the compiler operations. For this example, go to the “Output Format” list box and select “Intel HEX”. Also check the option “Accept C++ Comments”.

Figure 4. "Compiler" Options



☐ Strict ANSI C Checkings

☑ Accept C++ style comments

Macro Define(s):

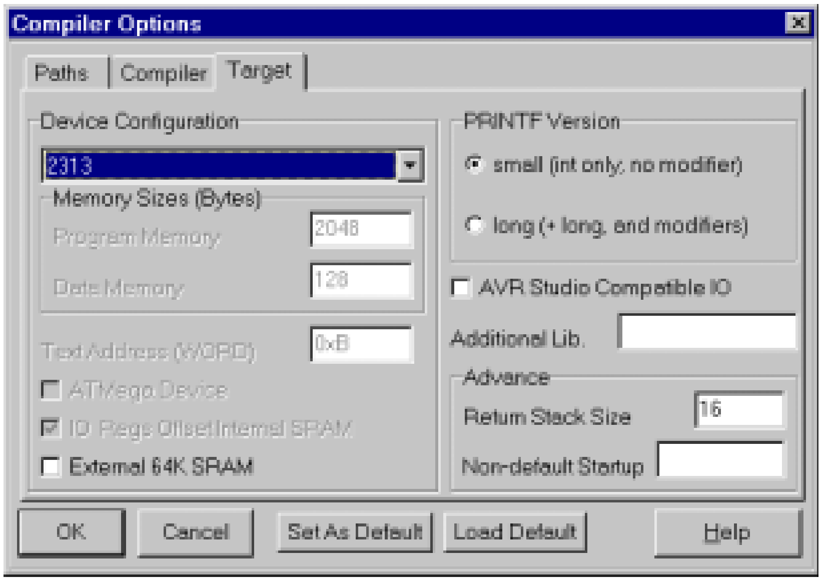
Undefine(s):

Output Format: Intel HEX

Target

In the “Target” page of the dialog box, you specify options that pertain to your target. For this example, go to the “Device Configuration” list box and select “2313”. Unlike some other compilers, the ImageCraft IDE specifies the exact options needed by the entire compiler toolchain (including the compiler, assembler and linker), making the IDE very simple to use.

Figure 5. “Target” Options



Device Configuration: 2313

Memory Sizes (Bytes)

Program Memory: 2048

Data Memory: 128

Text Address (WORD): 0xB

☐ ATMega Device

☑ IO Regs Offset Internal SRAM

☐ External 64K SRAM

PRINTF Version

● small (int only, no modifier)

○ long (+ long, and modifiers)

☐ AVR Studio Compatible IO

Additional Lib.:

Advance

Return Stack Size: 16

Non-default Startup:

Writing the Source File

When the project options are properly configured, the next step is to write the source code. This application note uses a simple program that increments PORTB on which the eight LEDs are attached. An 8-bit timer is used to generate a delay between increments, making it possible to see the LEDs flashing.

To create a new source file, select “File>New” or click on the “New” icon in the toolbar. In the new window that appears, type in the text below, and save it as “avr031.c” by selecting “File>SaveAs”. Although it is not required, you should save the file in the project folder, e.g. “c:\icctest\example”.

Program Listing for AT90S2313

```
#include <io2313.h>
void initialization(void);
void delay(void);

void initialization(void)
      {
      DDRB = 0xff;// set PORTB as output
      TCCR0 = 0x05;// Count clock/1024
      }

void delay(void) //Produce a delay of 65 ms at 4 MHz
      {
      while (!(TIFR&0x02));
                                               // Wait for timer0 overflow flag to be set
```

```
        TIFR = 0x02;                                // Clear overflow flag
        }

   void main (void)
        {
        initialization();                           //Initialize Peripherals
        while (1)                                   //Forever
            {
            PORTB++;                                //Increment PORTB
            delay();                                //Short delay
            }
        }
```

The program is divided into three parts: initialization, delay and main-loop. In the initialization section, PORTB is set as output and TIMER0 starts to count the main clock divided by 1024.

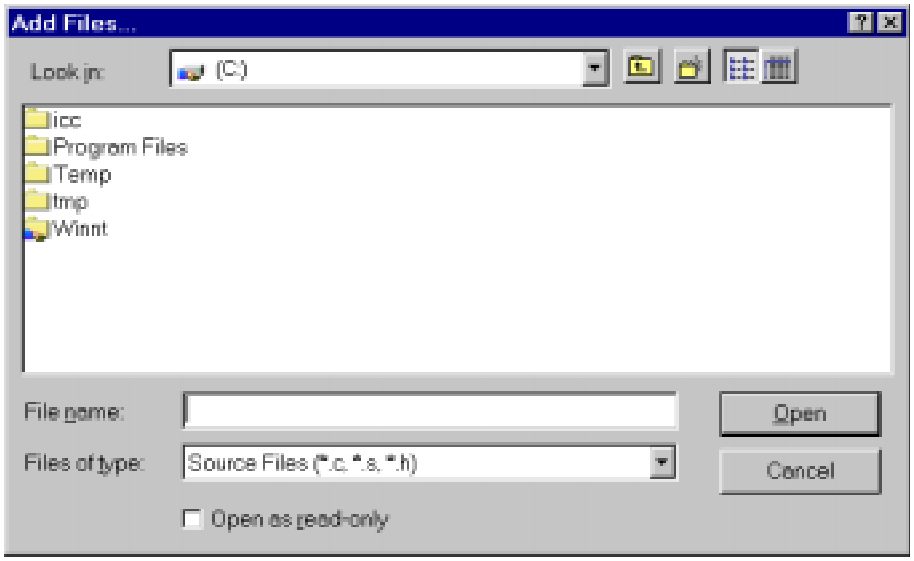
In the delay subroutine, the controller waits for the TIMER0 overflow flag to be set then clears the flag and exits.

In the main-loop, the content in PORTB is incremented and a delay is called to make the change on PORTB visible.

Including the Source File in the Project

When the source code is written, it has to be included in the project. Select the “Project>Add File(s)” menu. The dialog box shown in Figure 6 appears. Open the “c:\icctest\example” folder, select the file “avr031.c” by clicking on it, and click “OK”. You can also invoke this dialog box by right-clicking on the “Project List” window (the window pane on the right side of the IDE display) and selecting “Add Files” on the popup menu.

Figure 6. "Adding File(s)" to Project Dialog



Look in: (C:)

Files of type: Source Files (*.c *.s *.h)

Compiling the Code

To compile the code, Select “Project>Make Project” or click on the “Build Project” icon on the toolbar. If everything is done correctly, the code compiles and links with no errors and an executable hex code is placed in the file “c:\icctest\example\objs\example.hex”.

Loading the File into the STK200 Starter Kit

To run the code, the file has to be programmed into an AT90S2313. This application note describes how to load it to an AT90S2313 in the STK200 starter kit.

The software used by the STK200 is called AVR ISP. The STK200 dongle must be mounted on the parallel port. When this is done, a new project can be opened.

A new project is opened by selecting “Project>New Project” in AVR ISP. Highlight the AT90S2313 from the device selection menu and click “OK”.

In the “Project Manager” window information about the project can be typed in, and “fuse” and “lock-bit” options can be set. This is not necessary for this project.

The next step is to load the hex-file into the “Program Memory” window. To do this, activate this window by clicking on the title frame of the window. Now go to the "File" menu and select “Load”. In the dialog box that appears, open the “c:\icctest\example\objs” folder, and select the “example.hex” file.

To load the program into the AT90S2313 on the starter kit, select the “Program>Auto-Program” option. In the “Auto-Program” dialog box, tag “Reload Files”, “Erase device” and “Program device”. Now click “OK”. The LEDs on the starter kit should be counting.

Short Reference

Preparations

- Create your destination folder

Getting Started

1. Project>New
2. Project name and path
3. Project>Options
4. In the Paths options, enter “objs” in the output directory
5. In the Compiler options, check “Accept C++ Comments” and “Intel HEX” as the “output format”
6. In the Target options, select “2313” under “Device Configuration”
7. Write the source code
8. Add the source file to the project by selecting “Project>Add File(s)” and select the file just written
9. Compile by selecting “Project>Make Project” or by clicking on the “build” icon
10. Open AVR ISP and download the hex file located in the “objs” folder into the device

Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe
Atmel U.K., Ltd.
Coliseum Business Centre
Riverside Way
Camberley, Surrey GU15 3YL
England
TEL (44) 1276-686-677
FAX (44) 1276-686-697

Asia
Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan
Atmel Japan K.K.
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Atmel Colorado Springs
1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset
Zone Industrielle
13106 Rousset Cedex
France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

Fax-on-Demand
North America:
1-(800) 292-8635

International:
1-(408) 441-0732

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

BBS
1-(408) 436-4309

Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

Printed on recycled paper.