TASKING 251/USB Quick Start

Welcome

Welcome to the TASKING software evaluation tools for the 251 and 8x930 USB. The evaluation software consists of EDE, our MS-Windows based Embedded Development Environment, editor, the Power C compiler, assembler, linker/locator, and the CrossView Pro simulator and ROM monitor debugger. You can get up and running with the tools very quickly by following the Quick Start instructions below. The software can be used for any member of the 251 family, including the 8x930 USB peripheral controller. You will be able to compile, assemble and link programs, produce ROMable code and run it on CrossView Pro. In order to use CrossView Pro ROM, you will need a target board running the RISM ROM monitor. All Intel boards come with RISM inside an EPROM. TASKING has ported RISM to the Temic and Keil board. Preconfigured examples are delivered for:

CrossView Pro Execution Environment	Example	Project File for EDE
Simulator	Demo C program	c:\d251\examples\demo\sim.pjt
Intel 8x930 USB Evaluation Board	Demo C program	c:\d251\examples\demo\usb.pjt
Intel 151/251 Evaluation Board	Demo C program	c:\d251\examples\demo\rism.pjt
Intel 251 ProjectBuilder Board (old)	Demo C program	c:\d251\examples\demo\rismpb.pjt
Temic 251A1/G1 Evaluation Board	Demo C program	c:\d251\examples\demo\ri_temic.pjt
Keil MCB251 Evaluation Board	Demo C program	c:\d251\examples\demo\ri_temic.pjt
Simulator	Sieve C benchmark	c:\d251\examples\sieve\sieve.pjt
Simulator	Assembly program	c:\d251\examples\asm\asm.pjt

Table-1 Example Projects

Restrictions

The editor is a demo version and the compiler and assembler support a limited number of symbols and operands. The linker allows up to 3K of code size. CrossView also has some restrictions including the About box popping up every 5 minutes. Please refer to the Demo Limits help file for more details. Although these restrictions do apply, we believe that the demo package is adequate for you to be able to make a purchasing decision.

Installation

Start MS-Windows. Insert the 251/USB demo CD-ROM. For Windows 3.1x, in the Program Manager select the `File|Run...` menu item. For Windows 95, press the `Start` button and select the `Run...` menu item. In the dialog box type `D:\SETUP` (where D: is the drive letter of your CD-ROM player) and follow the instructions on the screen. At the end of the installation procedure a Program Group window will show up on the screen of your computer. It will look like the one shown in figure-1. You will use the EDE icon the most, since it presents you with a complete environment from where you can invoke the manuals and the other tools.

NOTE: When using Windows 95 you can create a shortcut on your desktop by dragging the EDE icon to the desktop using the right mouse button!

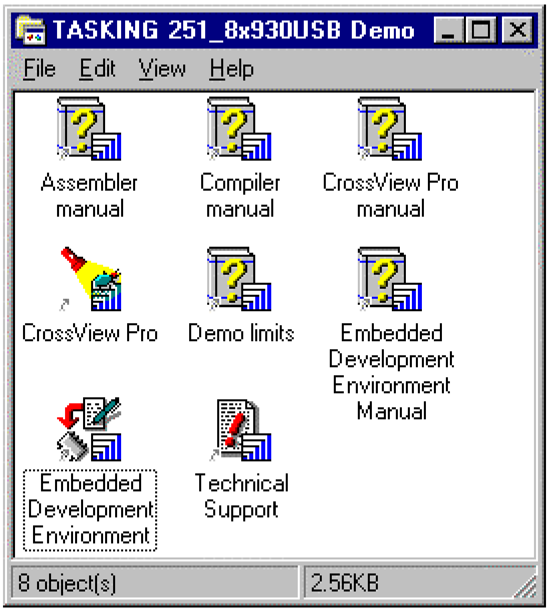


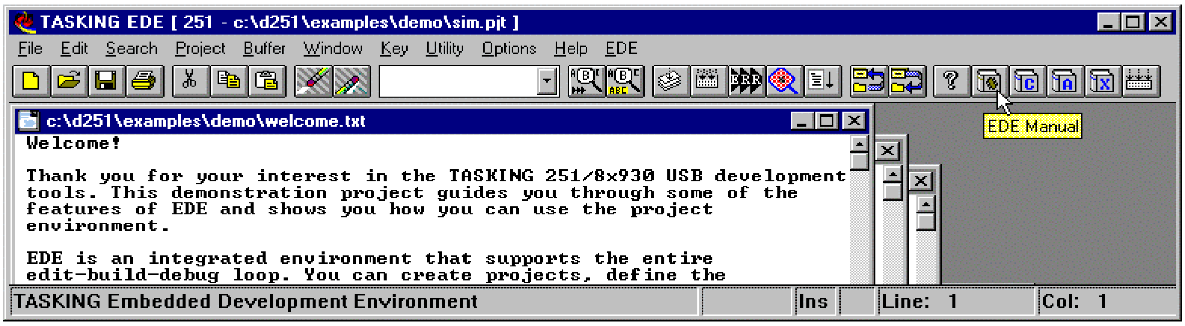
Figure 1 Program Group

Starting EDE

EDE is a standard Windows application which you can launch either by double-clicking the EDE icon in the Program Group, or via the Start Menu when using Windows 95. Please note that the EDE on-line manual also contains an extensive ***Getting Started with EDE*** chapter.

EDE Overview

EDE is an integrated embedded software development environment that combines a powerful editor with project management and an automated make facility. After you invoke the EDE, the window shown in figure-2 will show up on your screen.


TASKING EDE [251 - c:\d251\examples\demo\sim.pjt]
File Edit Search Project Buffer Window Key Utility Options Help EDE
EDE Manual
c:\d251\examples\demo\welcome.txt
TASKING Embedded Development Environment
Ins
Line: 1
Col: 1

```
Welcome!

Thank you for your interest in the TASKING 251/8x930 USB development
tools. This demonstration project guides you through some of the
features of EDE and shows you how you can use the project
environment.

EDE is an integrated environment that supports the entire
edit-build-debug loop. You can create projects, define the
```

Figure 2 Embedded Development Environment

From this environment you can create and maintain projects, edit files, specify tool options, compile, assemble and link your application, access on-line manuals and invoke the CrossView Pro debugger environment.

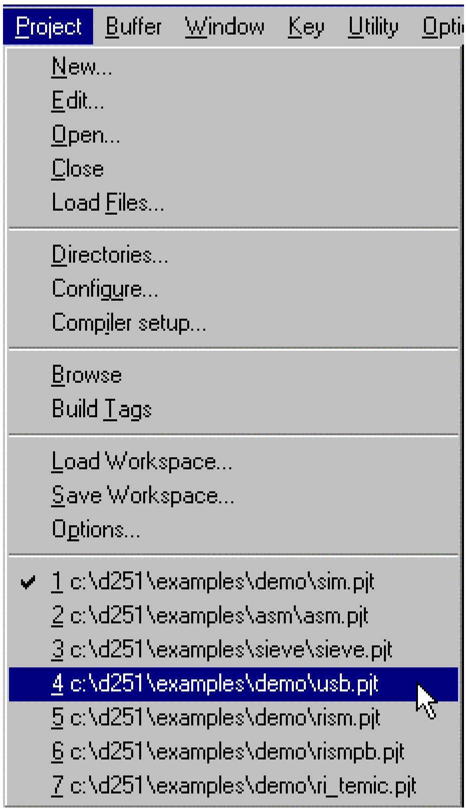

Project Buffer Window Key Utility Opti
New...
Edit...
Open...
Close
Load Files...
Directories...
Configure...
Compiler setup...
Browse
Build Tags
Load Workspace...
Save Workspace...
Options...
✓ 1 c:\d251\examples\demo\sim.pjt
2 c:\d251\examples\asm\asm.pjt
3 c:\d251\examples\sieve\sieve.pjt
4 c:\d251\examples\demo\usb.pjt
5 c:\d251\examples\demo\rism.pjt
6 c:\d251\examples\demo\rismpb.pjt
7 c:\d251\examples\demo\ri_temic.pjt

Figure 3 Project Menu

Project Management

EDE is more than a language sensitive editor. It is a complete project environment which gives you direct access to the tools and features you need to be your most productive. EDE lets you define the files of your project and, by navigating through the tabs, select the tool options that apply. With the EDE project manager you create and maintain a project so your application is always up-to-date. All aspects of a project are saved in the project file: the source files that make up the application, the tool options (compiler, assembler, linker and CrossView Pro debugger), the tool directories and the options describing the building process. The project manager handles file dependencies as well as the exact sequence of operations required to build your application.

There are a number of example demo project files included in the package, as listed in table-1 on page 1. By rebuilding a CrossView Pro demo you can verify the proper installation of the software and confirm that you can compile, assemble, link and debug an application. This tutorial will build the demo C application for the 8x930 USB evaluation board, but the steps are quite similar for the other execution environments. To switch from the simulator execution environment, which is the default project of the demo, to the 8x930 USB board, select `USB.PJT` from the `Project` menu.

Setting Processor Options

You can now edit the files belonging to the project. The files `DEMO.C` and `ADD251.SRC` are automatically opened. To see which files belong to the project, select the `Project|Load Files...` menu item. The next step is to specify the options for the Processor and the different parts of the toolchain.

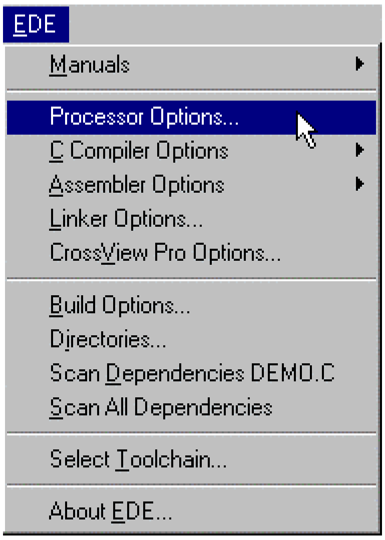
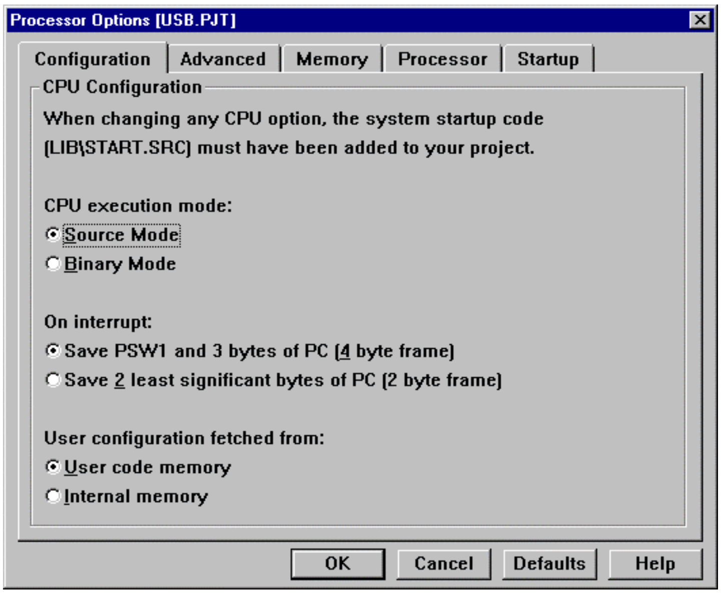


Figure 4 EDE Options


Processor Options [USB.PJT]

CPU Configuration

When changing any CPU option, the system startup code (LIB\START.SRC) must have been added to your project.

CPU execution mode:
● Source Mode
○ Binary Mode

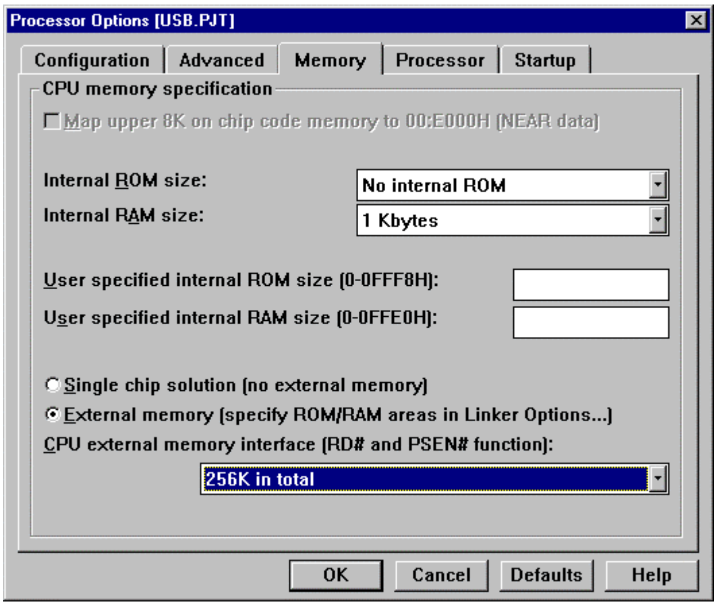
On interrupt:
● Save PSW1 and 3 bytes of PC (4 byte frame)
○ Save 2 least significant bytes of PC (2 byte frame)

User configuration fetched from:
● User code memory
○ Internal memory

Figure 5 CPU Execution Mode

The Processor Options allow you to choose the execution mode of the CPU: binary mode or source mode. Please note that the factory setting of most boards assumes binary mode. So you either have to rebuild the application for binary mode (select Binary Mode and click OK) or set the board to run in source mode (DIP switch MOD0 must be set to ‘on’ for Source Mode, ‘off’ for Binary Mode).

In the Memory tab of the Processor Options you specify how the CPU is configured with regards to internal ROM/RAM as well as the external memory interface (RD# and PSEN# functions). For the 8x930 USB board, the USB controller is a ROM-less part and has 1K of internal RAM. The external memory interface is configured for using 18 address lines (A0-A17), so the CPU addresses 256KB as one linear space from 0-3FFFFH. In the Linker Options you can now specify the external ROM and RAM areas within the 256KB range.


Processor Options [USB.PJT]

CPU memory specification

☐ Map upper 8K on chip code memory to 00:E000H (NEAR data)

Internal ROM size: No internal ROM
Internal RAM size: 1 Kbytes

User specified internal ROM size (0-0FFF8H):
User specified internal RAM size (0-0FFE0H):

○ Single chip solution (no external memory)
● External memory (specify ROM/RAM areas in Linker Options...)

CPU external memory interface (RD# and PSEN# function): 256K in total

Figure 6 CPU Memory Settings

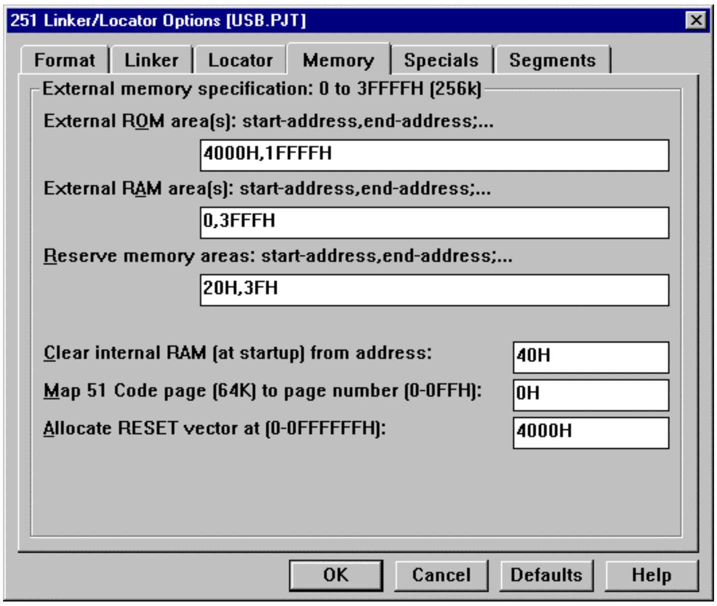
NOTE: *When changing any processor option the assembly system startup code (LIB\START.SRC) must be part of your project!*

Setting Development Tool Options

You can now specify the compiler and assembler options for your project. For example, the memory model to be used and which optimizations must apply. The compiler and assembler option selection responds to the active window. If the active window in EDE is e.g. a C source file, the compiler selection in the EDE menu allows you to set options for that specific file or for all the C files in the project. The order in which you do so is not important. Individual file options always prevail over project wide options. Every time you close the option selection from the EDE menu a new build file (makefile) is generated.

There is also a Memory tab inside the Linker Options dialog. This tab is very important, since here you specify where your external ROM and RAM areas are, which memory areas must be reserved and more. Figure 7 shows the settings for the 8x930 USB evaluation board.

The 8x930 USB board has 128KB of RAM for downloading application code and data. The upper 128 KB of memory is not used, except for the 32K EPROM where RISM resides. RISM expects the application and it’s interrupt vectors to be located at 4000H. In this example we have specified RAM in the lower 16KB and the rest of the memory (112K) to be used for downloading user code and constant data. When using RISM you must reserve memory from 20H to 3FH, since the ROM monitor uses this area. For the same reason, the system startup code must start to clear internal RAM at 40H (instead of 8H). For the 8x930 USB board the linker/locator must also map all the segments of type CODE (8051 compatible 64KB code area) to page 0 (0-0FFFFH) instead of page 0FFH (0FF0000H-0FFFFFFH), since RISM expects the user code to start in page 0. The last item to take care of is the reset vector. RISM assumes the application’s reset vector is at 4000H instead of 0FF0000H, so you must specify this to the linker/locator.



251 Linker/Locator Options [USB.PJT]

External memory specification: 0 to 3FFFFH (256k)

External ROM area(s): start-address,end-address;...
4000H,1FFFFH

External RAM area(s): start-address,end-address;...
0,3FFFH

Reserve memory areas: start-address,end-address;...
20H,3FH

Clear internal RAM (at startup) from address: 40H

Map 51 Code page (64K) to page number (0-0FFH): 0H

Allocate RESET vector at (0-0FFFFFFH): 4000H

Figure 7 Memory Settings for the 8x930 USB board

Building the Application

The next step is to compile and link the files in your project and build the program so you can debug the application. First save any edited files and then click on the ‘Rebuild’ button in the toolbar. EDE compiles and links your project and creates an absolute object module called `USB.ABS`, ready for debugging.

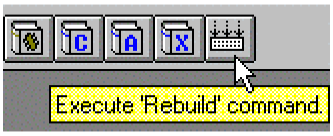


Figure 8 Rebuild

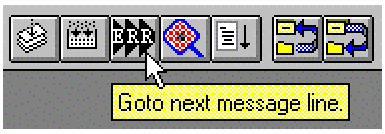
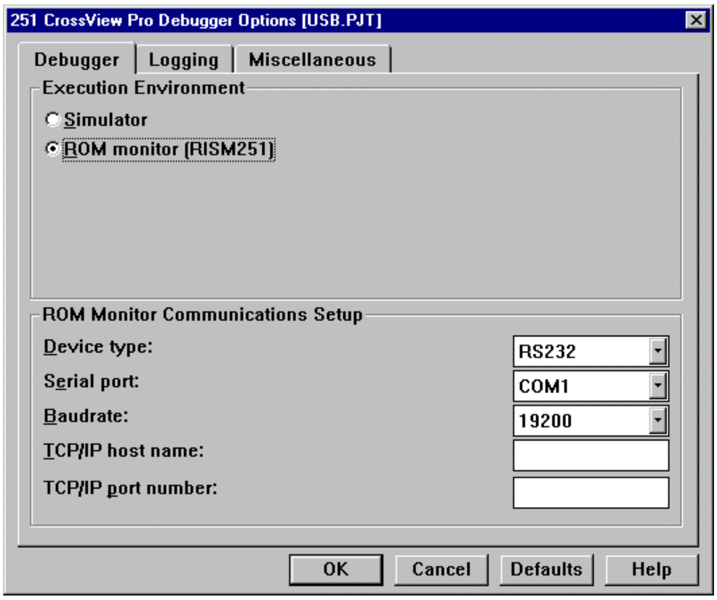


Figure 9 Error Messages

The building process is logged in the Output Window. If error messages occur, there is a button to browse through a list of error messages. When you use the 'ERR' button the editor prompts you automatically to the right position in the source file where the error has been detected. You can inspect the resulting map file by opening `USB.MAP` in the editor (`File|Open`…)

Debugging the Application with CrossView Pro

In order to simulate code execution or download code to the evaluation board, you need to invoke the CrossView Pro debugging environment. Before launching CrossView Pro you need to select the execution environment: either the simulator or the ROM monitor. For the ROM monitor version you have to connect the evaluation board to your PC and specify additional communications parameters. Select the `EDE|CrossView Pro Options...` menu item in order to invoke the CrossView Pro Options dialog.



251 CrossView Pro Debugger Options [USB.PJT]

Execution Environment

○ Simulator
● ROM monitor (RISM251)

ROM Monitor Communications Setup

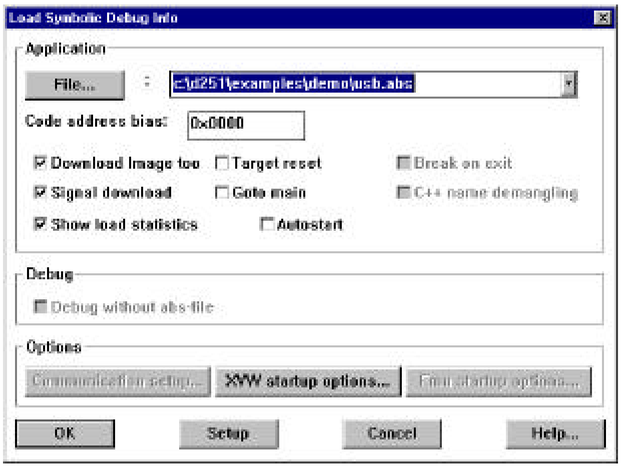
Device type: RS232
Serial port: COM1
Baudrate: 19200
TCP/IP host name:
TCP/IP port number:

Figure 10 CrossView Pro Options

Connect the 8x930 USB evaluation board to the correct COM port of your PC and power on the board. Specify a baudrate of 19200 when using the external serial port (UART), 9600 when using the internal serial port. After having pressed the OK button, you can launch the CrossView debugger by clicking on the ‘Debug’ (fly-swatter) button in the toolbar.

Loading the Application

CrossView Pro has now already loaded the application and downloaded the image to the board. Select the `File|Load Application` menu item in order to invoke the Load Application dialog. In this dialog you can specify an ‘Automatic Start’, ‘Target reset’ and ‘Goto main’ for subsequent debugging sessions.



Load Symbolic Debug Info

Application

File... : c:\d251\examples\demo\usb.abs

Code address bias: 0x0000

☑ Download Image too ☐ Target reset ☐ Break on exit
☑ Signal download ☐ Goto main ☐ C++ name demangling
☑ Show load statistics ☐ Autostart

Debug

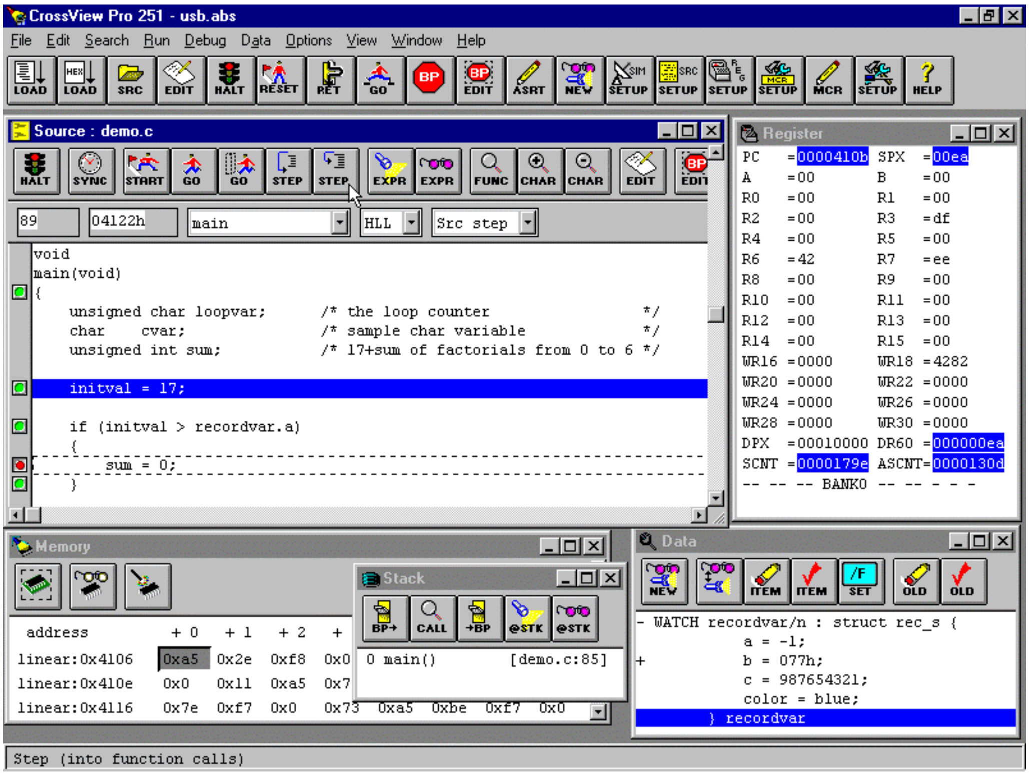
☐ Debug without abs-file

Options

Figure 11 Load Application

Stepping Through

Now click on `Run` in the menu bar and select `Program Reset`. This resets the software and the board and may take a few seconds. The reset empties the Source Window when it is in source display mode. At the reset address there is only startup code that does not relate to C-source. One program step (Step button in the toolbar) causes the program to execute the C-startup code, enter the `main()` function and fills the Source Window.


CrossView Pro 251 - usb.abs
File Edit Search Run Debug Data Options View Window Help
LOAD LOAD SRC EDIT HALT RESET RET GO BP EDIT ASRT NEW SETUP SETUP SETUP SETUP MCR SETUP HELP
Source : demo.c
HALT SYNC START GO GO STEP STEP EXPR EXPR FUNC CHAR CHAR EDIT EDIT
89
04122h
main
HLL
Src step
Register
Memory
Stack
BP→ CALL →BP @STK @STK
Data
NEW ITEM ITEM SET OLD OLD

```
void
main(void)
{
    unsigned char loopvar;      /* the loop counter            */
    char    cvar;               /* sample char variable        */
    unsigned int sum;           /* 17+sum of factorials from 0 to 6 */

    initval = 17;

    if (initval > recordvar.a)
    {
        sum = 0;
    }
```

```
PC    =0000410b SPX   =00ea
A     =00       B     =00
R0    =00       R1    =00
R2    =00       R3    =df
R4    =00       R5    =00
R6    =42       R7    =ee
R8    =00       R9    =00
R10   =00       R11   =00
R12   =00       R13   =00
R14   =00       R15   =00
WR16  =0000     WR18  =4282
WR20  =0000     WR22  =0000
WR24  =0000     WR26  =0000
WR28  =0000     WR30  =0000
DPX   =00010000 DR60  =000000ea
SCNT  =0000179e ASCNT=0000130d
-- -- -- BANK0 -- -- - - -
```

```
  address           + 0   + 1   + 2   +
linear:0x4106     0xa5  0x2e  0xf8  0x0
linear:0x410e     0x0   0x11  0xa5  0x7
linear:0x4116     0x7e  0xf7  0x0   0x73  0xa5  0xbe  0xf7  0x0
```

```
0 main()            [demo.c:85]
```

```
- WATCH recordvar/n : struct rec_s {
            a = -1;
+           b = 077h;
            c = 987654321;
            color = blue;
        } recordvar
```

```
Step (into function calls)
```

Figure 12 CrossView Pro Desktop

Using the toolbar or the menu bar you can edit breakpoints, watch data, display the stack, simulate I/O and much more. Please note that all the User Manuals are available as help files with hypertext links for easy navigation and the EDE manual has an extensive ***Getting Started with EDE*** chapter guiding you through the process of creating a new project.

Questions or Problems?

Depending on where you are located call one of the following numbers and tell customer support that you are using the 251/8x930 USB demo version 2.0.

USA:	**1-800-458-8276, fax 781 320 9212, e-mail: support_us@tasking.com**
Europe:	**+31 33 455 85 84, fax +31 33 455 10 05, e-mail: support_nl@tasking.com**
Japan:	**+81 3 3457 6831, fax +81 3 3457 6834, e-mail: support@tasking.co.jp**

See us also at: **http://www.tasking.com**