珠海欧比特控制工程股份有限公司

ZHUHAI ORBITA CONTROL ENGINEERING CO.，LTD.

集成开发环境 Orion 4.0
用户手册

集成开发环境 Orion4.0
用户手册

版本号：Rev.2.1
2008.01

珠海欧比特控制工程股份有限公司
www.myorbita.net

前　　言

本手册是 Orion4.0 集成开发环境的用户手册。该手册就使用 Orion4.0 进行嵌入式应用开发的流程进行了详尽描述，每个步骤都给出图例和解释，以方便用户尽快地熟悉和掌握 Orion4.0 集成开发环境。

Orion4.0 集成开发环境界面友好，利用此开发环境，用户不仅可以编辑、编译、调试程序，还可以查看开发板内所有存储器资源及处理器的内部寄存器，更加方便嵌入式应用软件的开发。

选用 SPARC V7/V8 系列处理器开发嵌入式系统时，选择合适的开发工具可以加快开发进度，节省开发成本。集成编辑软件、编译软件、汇编软件、链接软件、调试软件、工程管理及函数库的 Orion4.0 将使您的开发得心应手，事半功倍。

Orion4.0 运行的主机环境为 Windows98/NT/2000/XP，支持的开发语言包括标准 C，C++和汇编语言。

Orion4.0 集成开发环境包含 Cygwin、Leccs 开发包以及调试烧写工具软件如 V8mon，mkflash 等。Orion4.0 对这些工具进行了集成，使用户在统一的图形界面里进行开发。Orion4.0 是一个高度集成的图形界面操作环境，其界面同 Microsoft Visual Studio 类似。Orion4.0 同时支持国际流行的 GRMON 调试器。

该手册的读者应当具备 C/C++ 编程基础，并且了解嵌入式软件开发过程中的编译、链接、调试等概念。

对Orion4.0 集成开发环境详细了解，可登陆http://www.myorbita.net

目　录

1 引言

1.1 编写目的

本手册向读者介绍 Orion4.0 集成开发环境的基本概念和操作，引导读者使用 Orion4.0 开发基于 SPARC 系列处理器的嵌入式软件。

1.2 背景

a. 软件名称：Orion4.0

b. Orion4.0 集成开发环境提供针对 SPARC 架构处理器(TSC695F、S698 系列等)的嵌入式软件开发过程中的工程管理、源程序编辑、编译、链接、运行、调试等功能，以及提供各种工具协助软件的开发。

c. Orion4.0 由珠海欧比特控制工程股份有限公司开发和维护。

1.3 定义

TSC695F：基于 SPARC V7 指令集的一款 CPU。

S698：由珠海欧比特控制工程股份有限公司设计，基于 SPARC v8 指令集的处理器芯片系列。

RTEMS：Real-Time Executive for Multiprocessor Systems，现由 OAR 公司开发维护。

DSU（ Debug Support Unit）： Sailing S698 处理器内部附带的硬件调试支持单元，在激活时能提供读/写 Sailing S698 的寄存器和内存的功能，提供内置的反汇编器和 Trace buffer 管理；具有下载和运行程序，断点和监视点管理，远程连接 GDB 等功能。

Cygwin：是一个在 windows 平台上运行的 linux 模拟环境，运行 Cygwin 后，会得到一个类似 Linux 的 Shell 环境，在其中可以使用绝大部分 Linux 软件和功能，如 Gcc,Make 等；Cygwin 是 cygnus solutions 公司开发的自由软件。

2 运行环境

2.1 硬件设备

运行本软件所要求的硬件设备的配置：

a. 建议采用 Pentium 或更高级的处理器

b. 256 MB 或以上的内存

c. 至少 350 MB 的可用硬盘空间

d. 至少一个串口

e. 网卡（可选）

2.2 支持软件

运行本软件所需要的支持软件：

a. 操作系统：Windows95/98/NT/2000/XP

b. 超级终端（可选）

3 安装与卸载

3.1 Orion4.0 安装

进入安装光盘，直接双击安装文件 setup.exe，系统将引导您安装 Orion4.0。

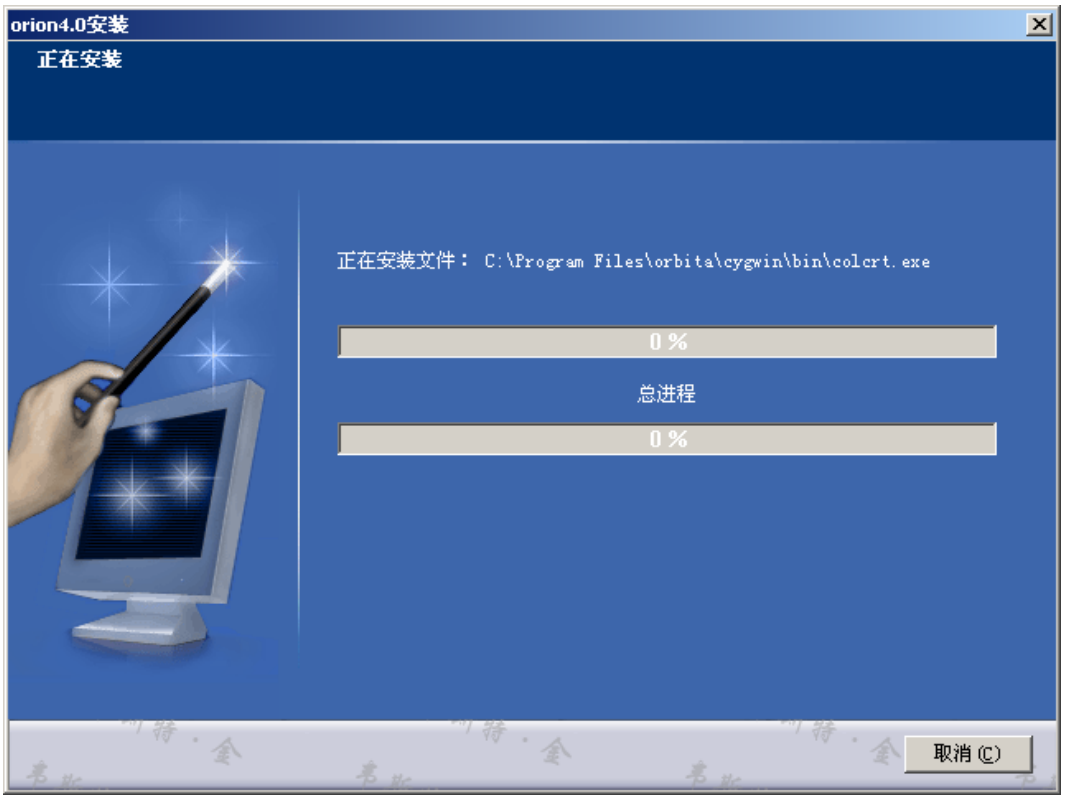


图 3-1 Orion4.0 的安装

3.2 Orion4.0 卸载

点击[开始]->[程序]->[Orion]->[卸载 Orion]项卸载 Orion4.0；或者通过[控制面板]->[添加或删除程序]项卸载 Orion4.0。

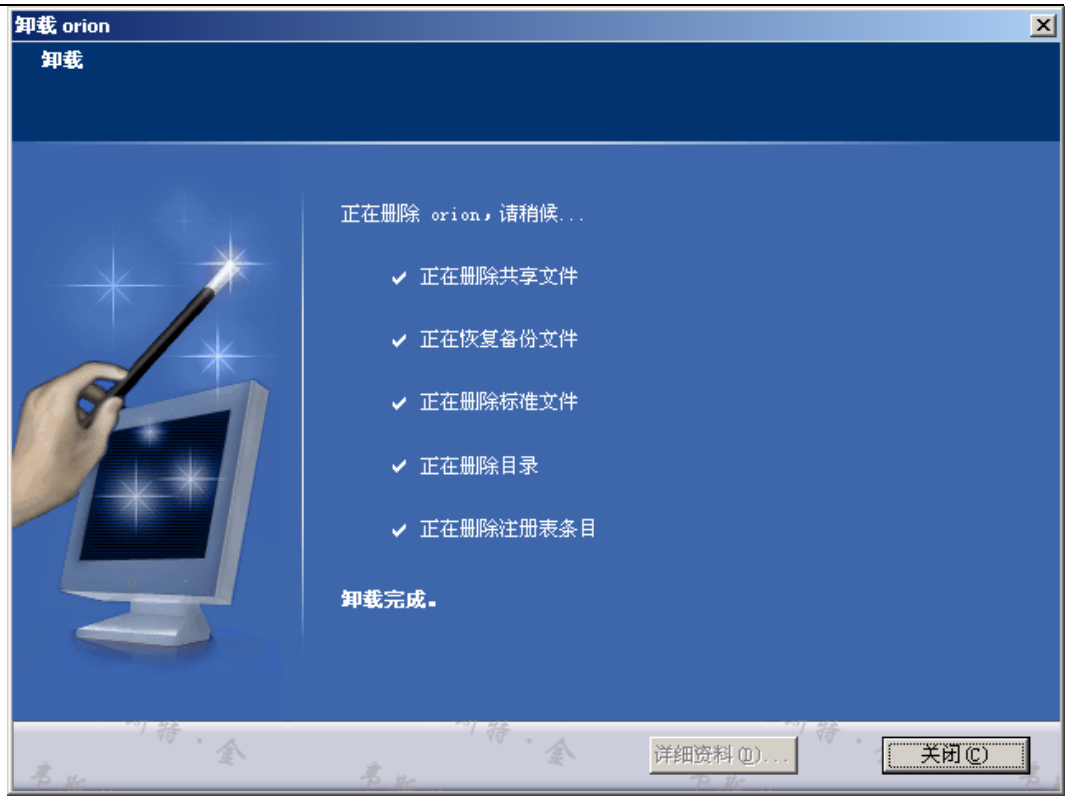


图 3-2 Orion4.0 的卸载

如果不能删除指定的目录，可能是因为程序占用，请手动删除，并将环境变量 Path 中与 Orion4.0 相关的部分删去。

3.3 启动 Orion4.0

Windows95/98/NT/2000/XP 的开始菜单中点击 Orion4.0 相应的菜单项或双击桌面上的 Orion4.0 的快捷方式图标后，启动 Orion4.0 集成开发环境。

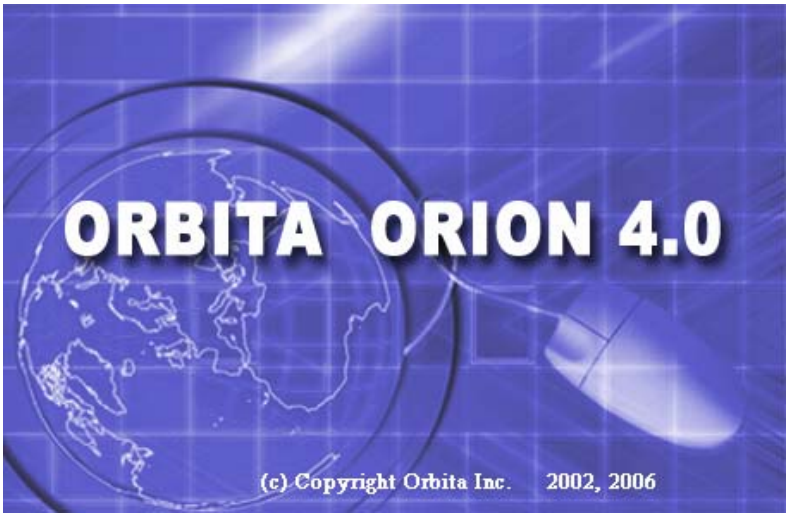


图 3-3 启动画面

4 界面组件介绍

4.1 工作台

图 4-1 为 Orion4.0 集成开发环境主界面示意图。

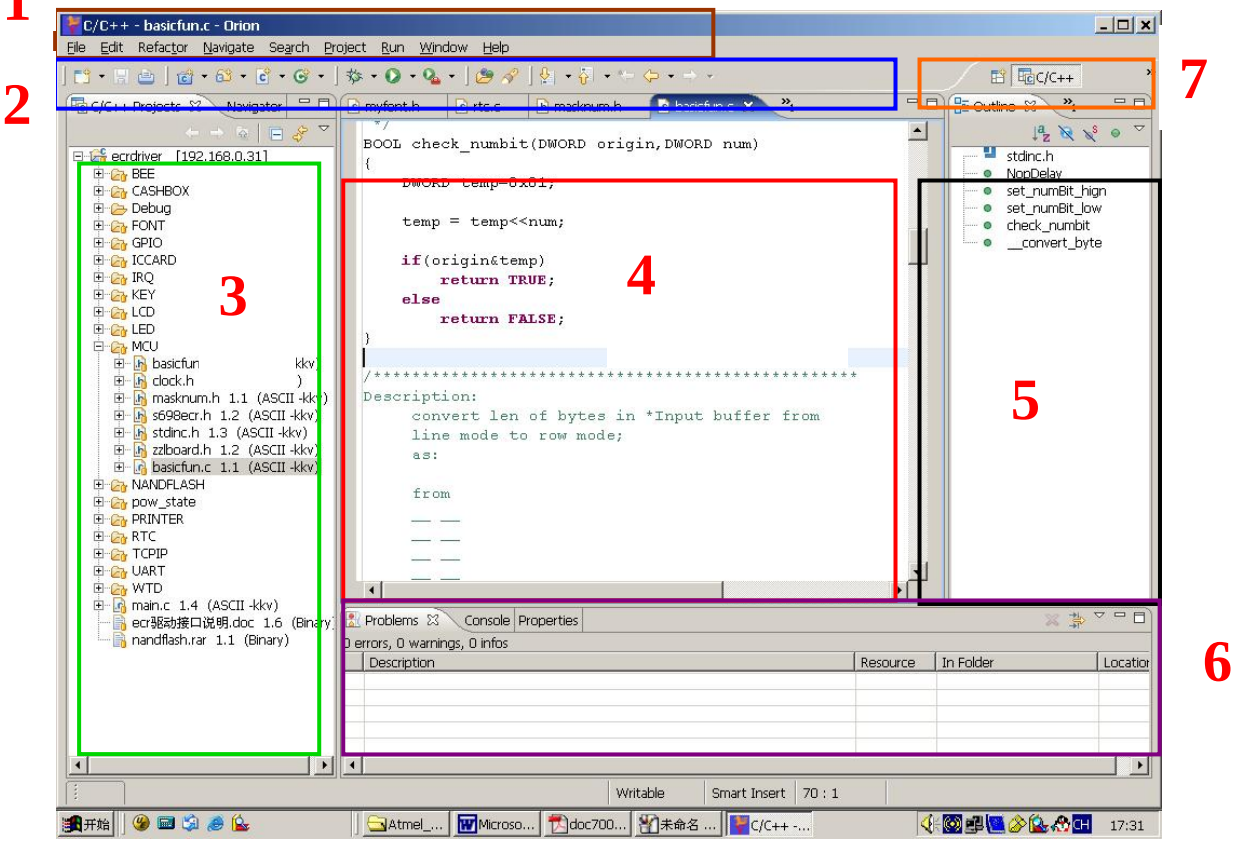


图 4-1 主界面

以下是图 4-1 主界面的图解：

1. 菜单栏
2. 工具栏
3. 工程管理视图
4. 编辑视图
5. 大纲视图
6. 控制台视图
7. 捷径工具列

Orion4.0 集成开发环境主界面的最上端为工具栏。工程管理视图用于显示当前打开工程的有关信息，包括工程的文件组成等内容。控制台视图通常在工程管理视图和编辑视图的下面，用于输出编译信息、调试信息和输出一些查询结果信息等。主界面的最底端为状态栏，用于显示行列数的提示信息。

4.2 按钮与工具栏

下列按钮将会出现在工作台工具栏、视图工具栏和快捷方式栏中：

按钮	描述	按钮	描述
	打开新的透视图		保存活动编辑器内容
	保存所有编辑器的内容		用新的名称或位置保存编辑器内容
	打开“搜索”对话框		打印编辑器内容
	打开资源创建向导		打开文件创建向导
	打开文件夹创建向导		打开项目创建向导
	打开导入向导		打开导出向导
	编译程序		运行程序
	调试程序		打开视图的下拉菜单
	将选择剪切到剪贴板中		将选择复制到剪贴板中
	从剪贴板粘贴选择		撤销最近的编辑
	重做最近撤销的编辑		浏览至列表中的下一项
	关闭视图或编辑器		锁定编辑器以防止内部自动调用
	刷新视图内容		按字母顺序对列表进行排序

按钮	描述	按钮	描述
	取消长期运行的操作		删除选择的项或内容

4.3 视图

工作台会有许多不同种类的内部窗口，称之为视图(view)，以及一个特别的窗口--编辑器(editor)。不同的视图都以不同的角度来显示各项目，例如在主界面中 Outline 的视图可以看项目中的概略状况，而 Navigator 的视图可以浏览各项目。

视图支持编辑器，且可提供工作台中信息的代替呈现或浏览方式。如： Navigator 视图会显示项目和其它资源。图 4-2 为任务视图示意图，4-3 为 navigator 视图示意图：

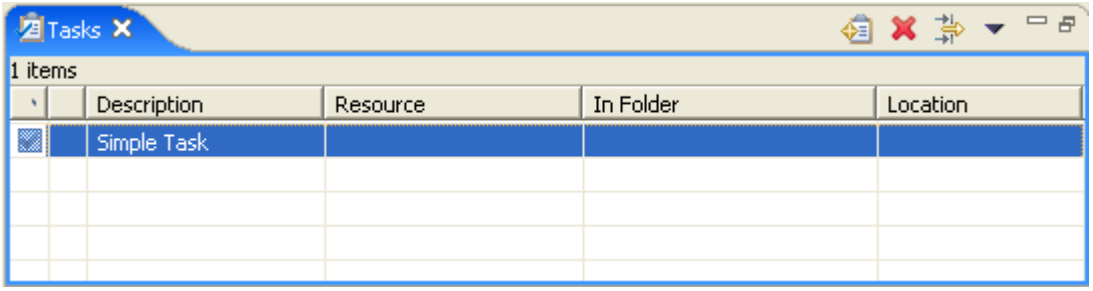


图 4-2 任务视图

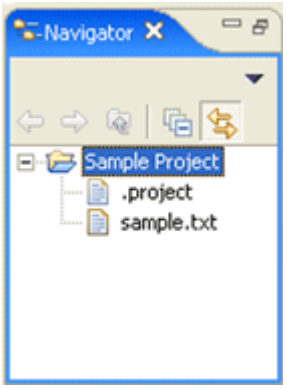


图 4-3 navigator 视图

视图有两个菜单，第一个是用鼠标右键按一下视图图标来存取的菜单，它可以利用类似工作台窗口相关菜单的相同方式来操作视图。

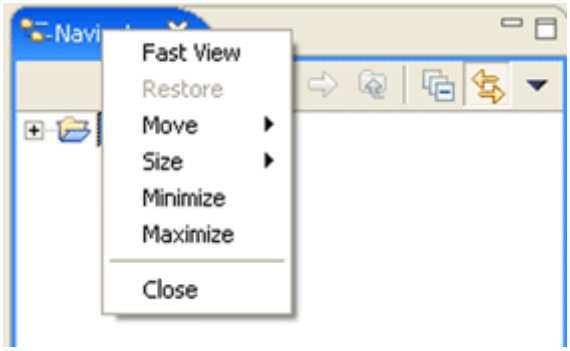


图 4-4 视图图标存取菜单

第二个菜单为视图下拉菜单，存取方式是按一下向下箭头。视图下拉菜单所包含的选项通常会应用到视图的全部内容，而不是应用到视图中所显示的特定项目。

图 4-5 视图下拉菜单

可以从主菜单 Window->Show View 中选取一个视图。选择 Show View 子菜单底端的 Other...时，就可以使用其它的视图。

4. 4 编辑器

编辑器是很特殊的窗口，会出现在工作台的中央。当打开文件、程序代码或其它资源时，Orion4.0 会选择最适当的编辑器打开文件。若是纯文字文件，Orion4.0 就用内置的文字编辑器打开；若是 C 程序代码，就用 C 编辑器打开。

Orion4.0 编辑器是一个标准的文本编辑器，它具有如下特点：

a）支持 C/C++语言的语法高亮显示，并有其他视图协助观察程序；

b）和调试器集成，编辑窗口同时作为调试时的源代码可执行跟踪窗口；

c）和编译器集成，而且编译信息可定位到源代码编辑窗口中。

Orion4.0 编辑器可打开多个窗口，同时编辑多个文件，而且编辑文件的大小理论上是无限制的（当然，会受计算机内存、硬盘空间等硬件资源的限制）。编辑器的编辑命令和编辑操作同标准 Windows 的编辑器功能一致，因此操作起来十分的方便。

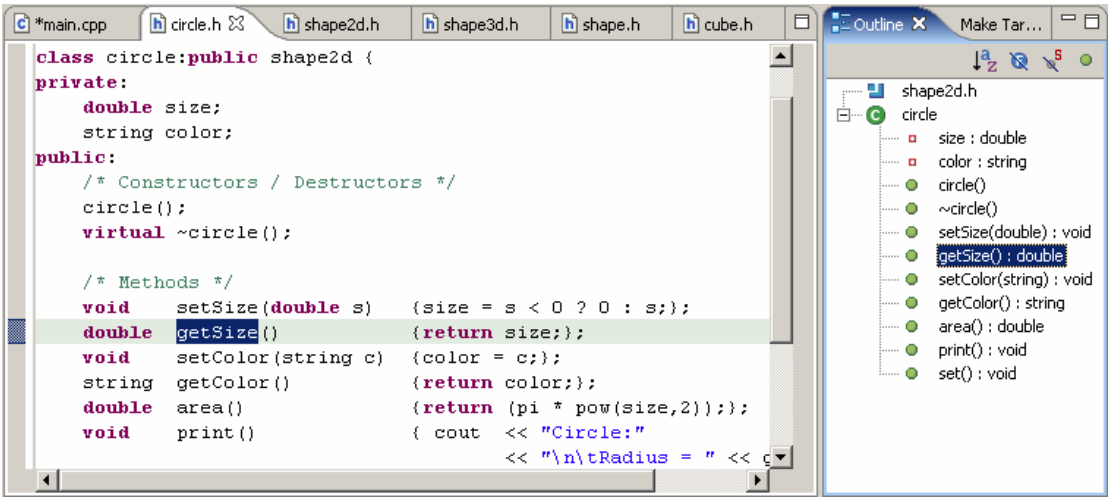


图 4-6 编辑器

4.5 透视图

在不同的状况下，Orion4.0 会将各种视图按照不同的方式排列，不同的排列组合模式定义为不同的透视图(perspective)。在我们的嵌入式开发过程中将主要用到 C/C++和 Debug 两种透视图。

比如在 Debug 的透视图中，其中一个视图会显示程序代码，另一个视图会显示变量目前的值，还有一个视图会显示程序的执行结果。图 4-7 为 Debug 透视图示意图：

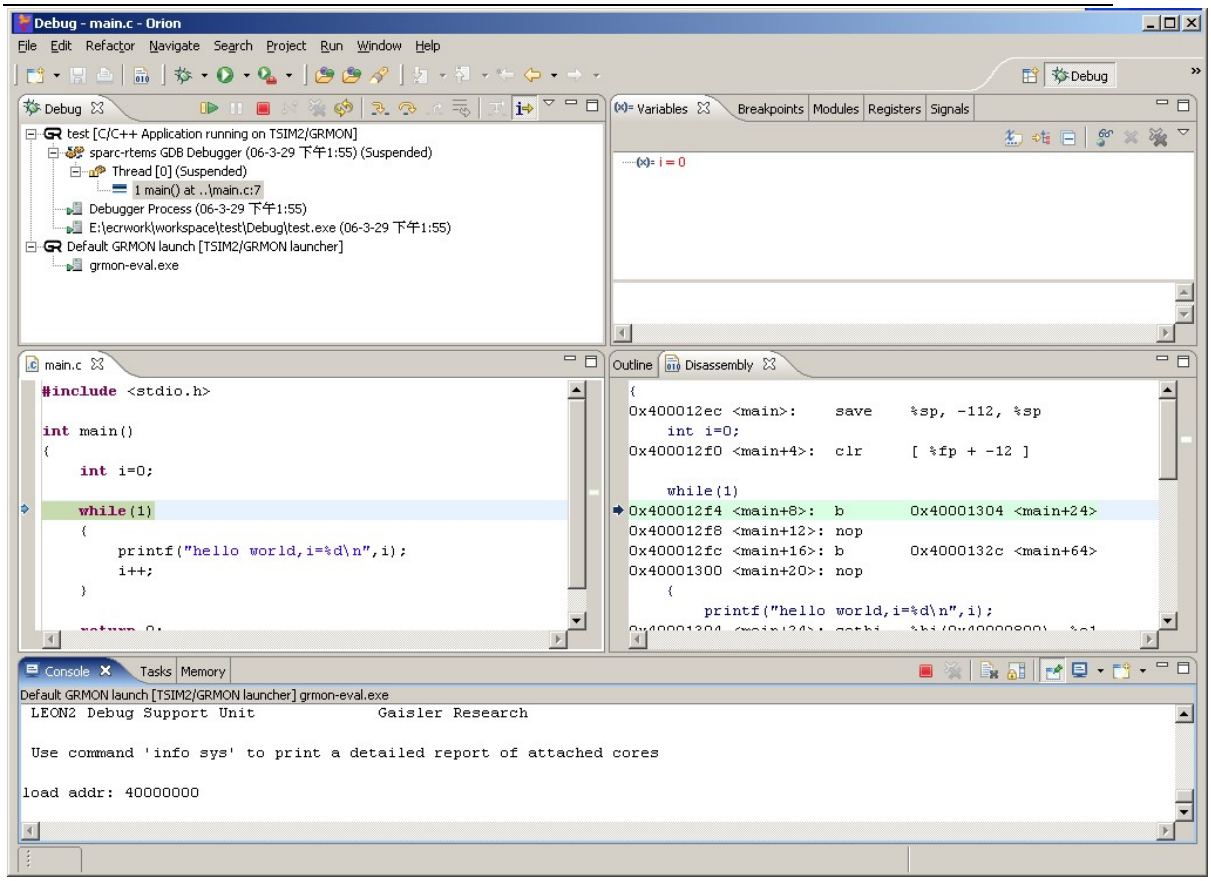

Debug - main.c - Orion
File Edit Refactor Navigate Search Project Run Window Help
Debug
Debug
test [C/C++ Application running on TSIM2/GRMON]
sparc-rtems GDB Debugger (06-3-29 下午1:55) (Suspended)
Thread [0] (Suspended)
1 main() at ..\main.c:7
Debugger Process (06-3-29 下午1:55)
E:\ecrwork\workspace\test\Debug\test.exe (06-3-29 下午1:55)
Default GRMON launch [TSIM2/GRMON launcher]
grmon-eval.exe
Variables
Breakpoints
Modules
Registers
Signals
(x)= i = 0
main.c
#include <stdio.h>
int main()
{
int i=0;
while(1)
{
printf("hello world,i=%d\n",i);
i++;
}
Outline
Disassembly
{
0x400012ec <main>: save %sp, -112, %sp
int i=0;
0x400012f0 <main+4>: clr [%fp + -12]
while(1)
0x400012f4 <main+8>: b 0x40001304 <main+24>
0x400012f8 <main+12>: nop
0x400012fc <main+16>: b 0x4000132c <main+64>
0x40001300 <main+20>: nop
{
printf("hello world,i=%d\n",i);
Console
Tasks
Memory
Default GRMON launch [TSIM2/GRMON launcher] grmon-eval.exe
LEON2 Debug Support Unit Gaisler Research
Use command 'info sys' to print a detailed report of attached cores
load addr: 40000000

图 4-7 Debug视图

5 工程管理

5.1 选择工作区

图 5-1 选择工作区

工作区是工程的集合；Orion4.0 启动时首先要求选择工作区，输入工作区路径后点 OK 进入主界面；此时可以新建工程，打开、关闭已有的工程。

工作区与工作区之间是相互独立的，进入主界面后可以通过菜单 File->Switch Workspace 来切换不同的工作区。

注意：工作区的路径名中不允许包含汉字以及空格等特殊字符，否则调试下载时将产生错误。

5.2 新建工程

1. 进入主界面后点击主菜单 File->New->Project，弹出工程创建对话框，在 Wizards 列表中选中 Orion C 中的 Bare C Project 或者 Rtems C Project，单击 next 按钮。

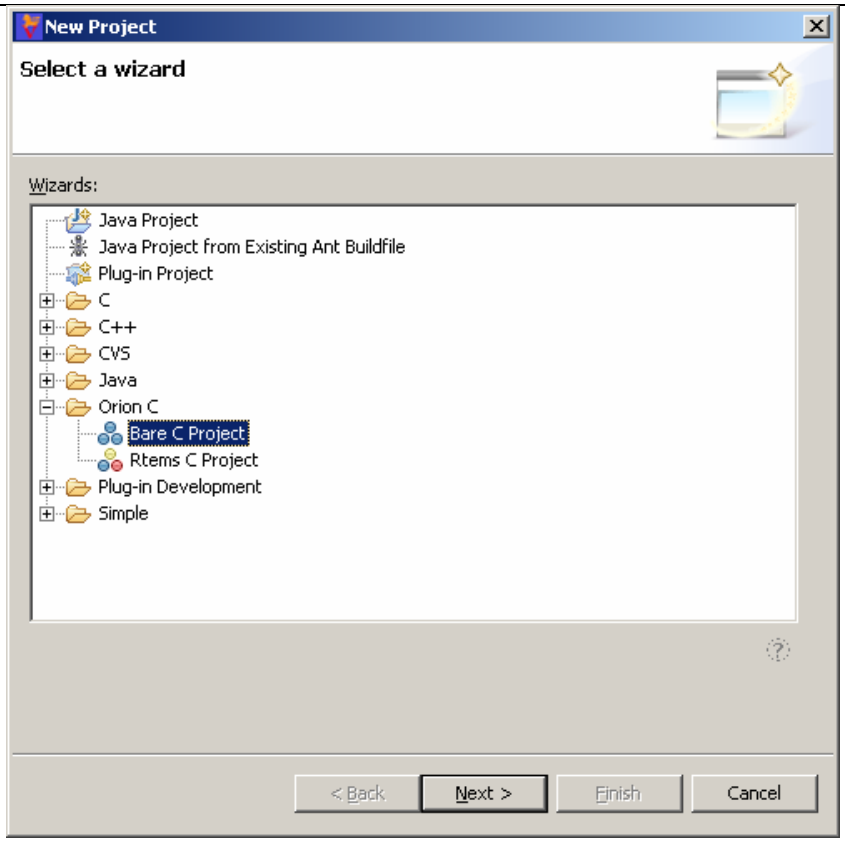


图 5-2 新建工程

补充说明：

Bare C Project 生成的工程，是不带操作系统的标准 C 工程。但也可以通过更改编译配置来带 RTEMS 操作系统运行，具体见 6.2 编译选项的配置。

RTEMS C Project 生成的工程，就是带 RTEMS 操作系统的 C 工程。

2. 在工程名输入框中输入工程名（例图中为 test），单击 next 按钮。**注意：工程名中不允许包含空格等特殊字符。**

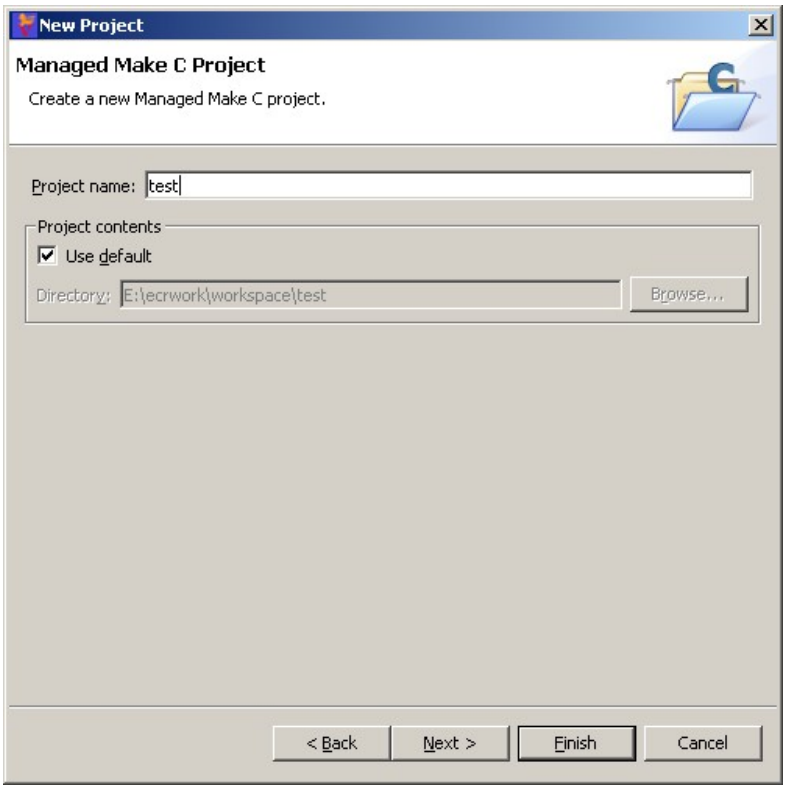


图 5-3 指定工程名称

3. 在 Project Type 中已经默认选择了 Orbita.inc (sparc-rtems toolchain，link without rtems-OS)，单击 next 按钮。

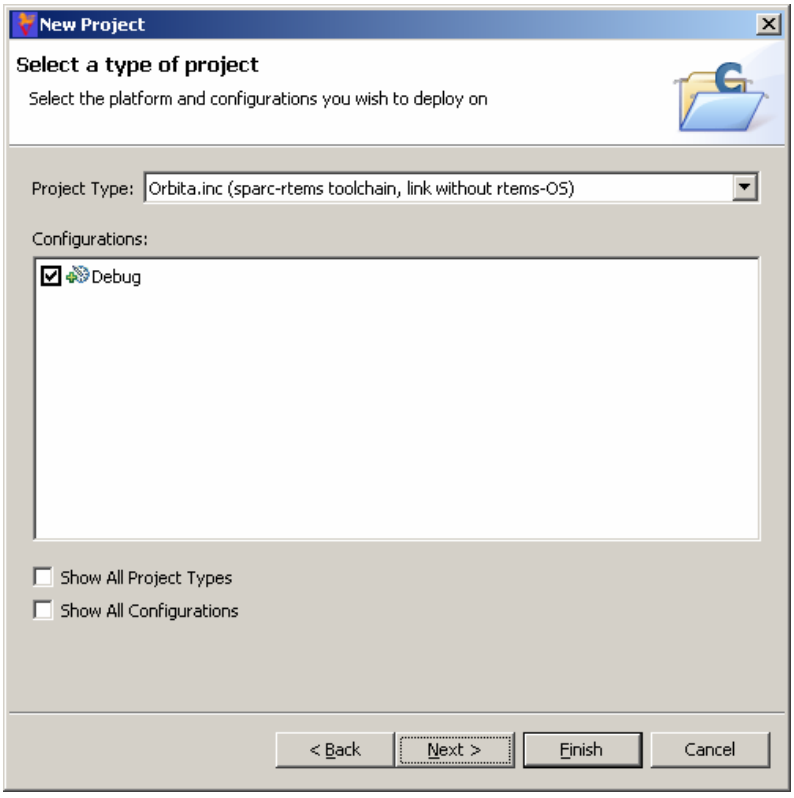


图 5-4 选择类别

4. 单击 Finish 按钮，建立工程完毕，系统自动生成了 Main 文件和 main 函数，用户可以修改编辑。

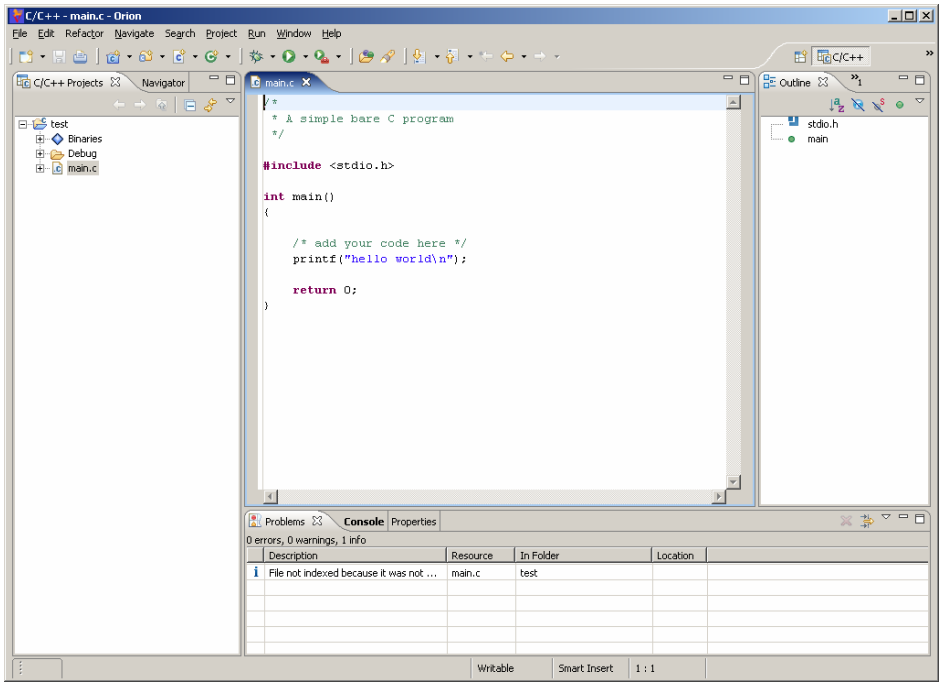


图 5-5 生成工程

5. 新增文件：在工程管理视图中点击右键，选择 New→Source File，输入文件名，然后单击 Finish 按钮；如果要新建多个源文件，可重复以上步骤依次添加；

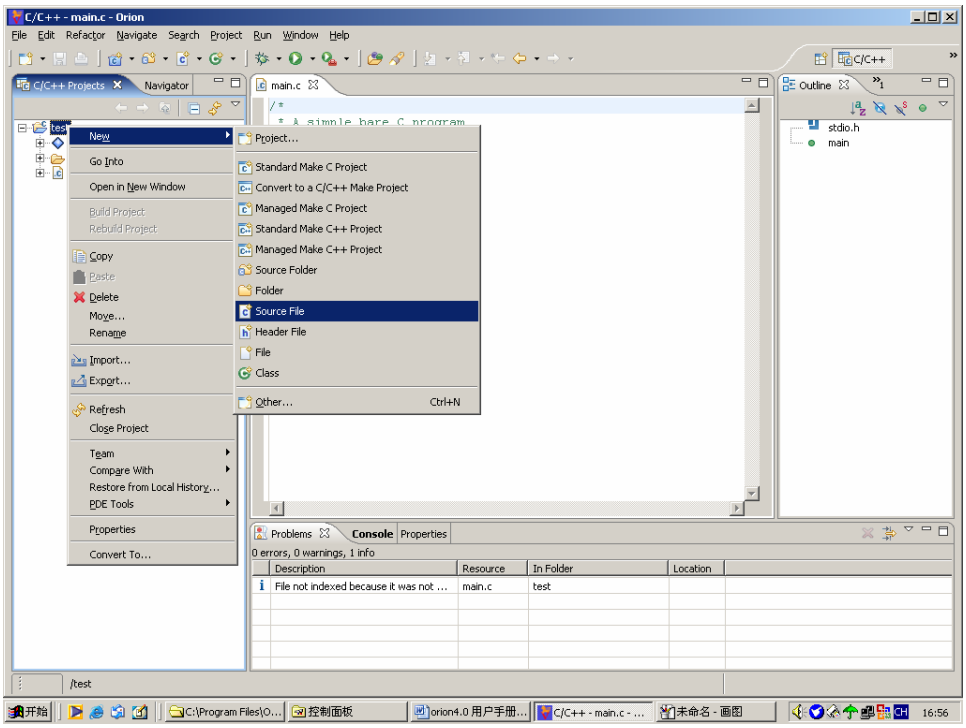


图 5-6 通过浮动菜单新建源文件

5.3 关闭工程

操作步骤：

1. 在工程管理视图中选择该工程。
2. 单击弹出菜单上的“Close Project”。

关闭工程后，就再也不能在工作区中更改它，并且它的资源不再出现在工作区中，但这些资源仍会驻留在本地文件系统上。已关闭的项目需要的内存很少。此外，因为在编译期间不会检查这些关闭的工程，所以关闭工程可以缩短其他打开工程的编译时间。

如果要重新打开工程，操作步骤如下：

1. 在工程管理视图中选择该工程。
2. 单击弹出菜单上的“Open Project”。

5.4 删除工程

1. 在工程管理视图中选择该工程；
2. 单击弹出菜单上的删除；
3. 在打开的对话框中选择删除方式：
   - 从工作区删除工程的同时将工程中的内容从硬盘中删除；
   - 不删除工程中的内容；
4. 单击 Yes 按钮；

5.5 导入工程

Orion4.0 允许将其它工作区的工程或已经删除的工程导入到本工作区中，操作步骤如下：

1、 在工程管理视图中点右键，浮动菜单中选择“Import”项，弹出文件选择对话框；

2、 Select 页：选择 Existing Projects into Workspace 项，单击 next 按钮；

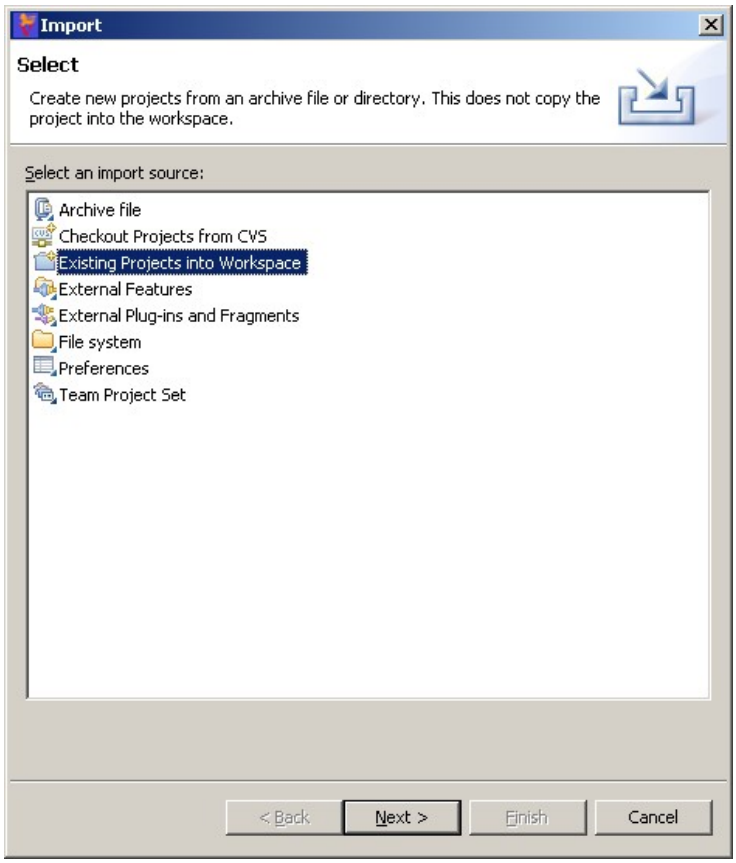


图 5-7 选择导入类型

3、 Import Projects 页：单击 Browse 按钮选择要导入的工程的文件夹，单击 Finish 按钮结束。

5.6 向工程中导入文件

要将已有的文件添加到工程中，可以使用文件导入功能，操作步骤如下：

1、 在工程管理视图中点右键，浮动菜单中选择“Import”项，弹出文件选择对话框；

2、 Select 页：选择 File system 项，单击 next 按钮；

3、 File system 页：单击 Browse 按钮选择目标文件夹，在右边的文件选择栏中勾选要添加到工程中的文件，单击 Finish 按钮结束。

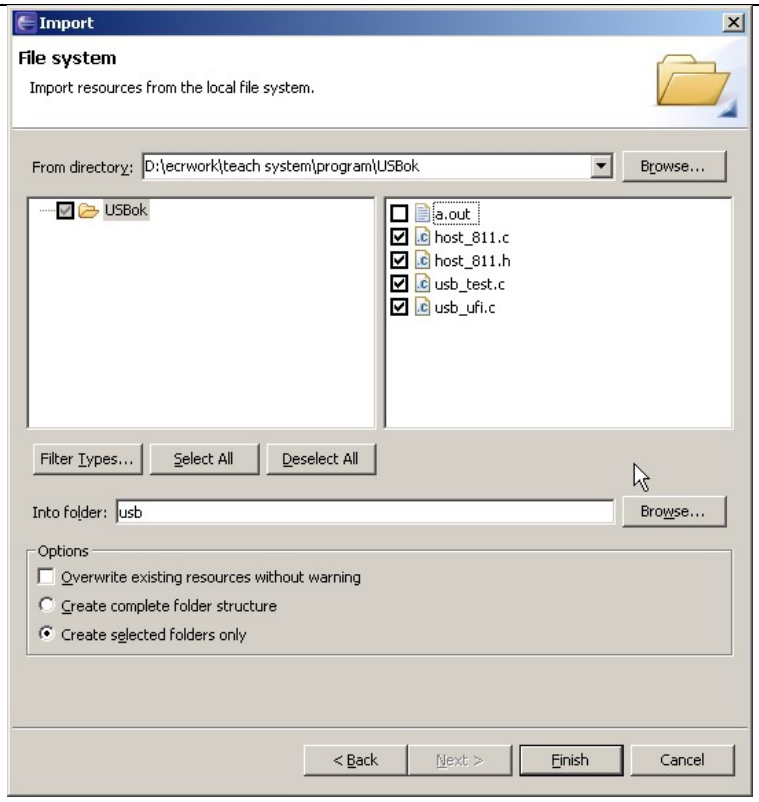


图 5-8 向工程中添加文件

5.7 从工程中删除文件

选中文件后点击右键，选择“Delete”可以从工程中删除文件。

6 编辑与编译

6.1 代码编辑

在编写程序代码的过程中，编辑器具有支持语法关键字的色彩显示，具体的显示色彩可由用户自由编辑。这样就方便了用户代码编写，提高了代码的编写质量。编辑器还提供了代码协助功能。在编辑器中输入 C/C++的关键词，然后按 ALT+/，再根据提示选择你想要的协助代码。除了 Orion4.0 自带的关键词支持，你可以自定义代码协助模板或者编辑现存的代码协助模板(在菜单 Window→Preferences→C/C++的 Templates 设置页里面设置)。

编辑菜单支持标准的 Windows 文本编辑功能：撤消键入、重复键入、剪切、复制、粘贴、清除、全选等。在编辑窗口中，还可通过鼠标选中文本进行拖拉移动操作。快捷键设置和 Windows 文本编辑器的快捷键设置基本一致:

CTRL+C:复制

CTRL+X:剪切

CTRL+V:粘贴

CTRL+F:查找/替换

CTRL+D:删除整行

要查看全部的快捷键定义:CTRL+SHIFT+L

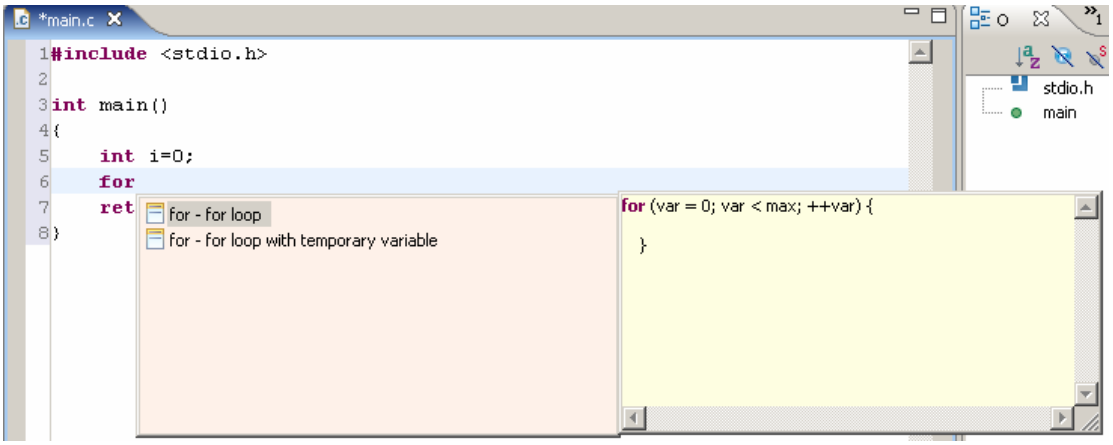


图 6-1 代码协助功能

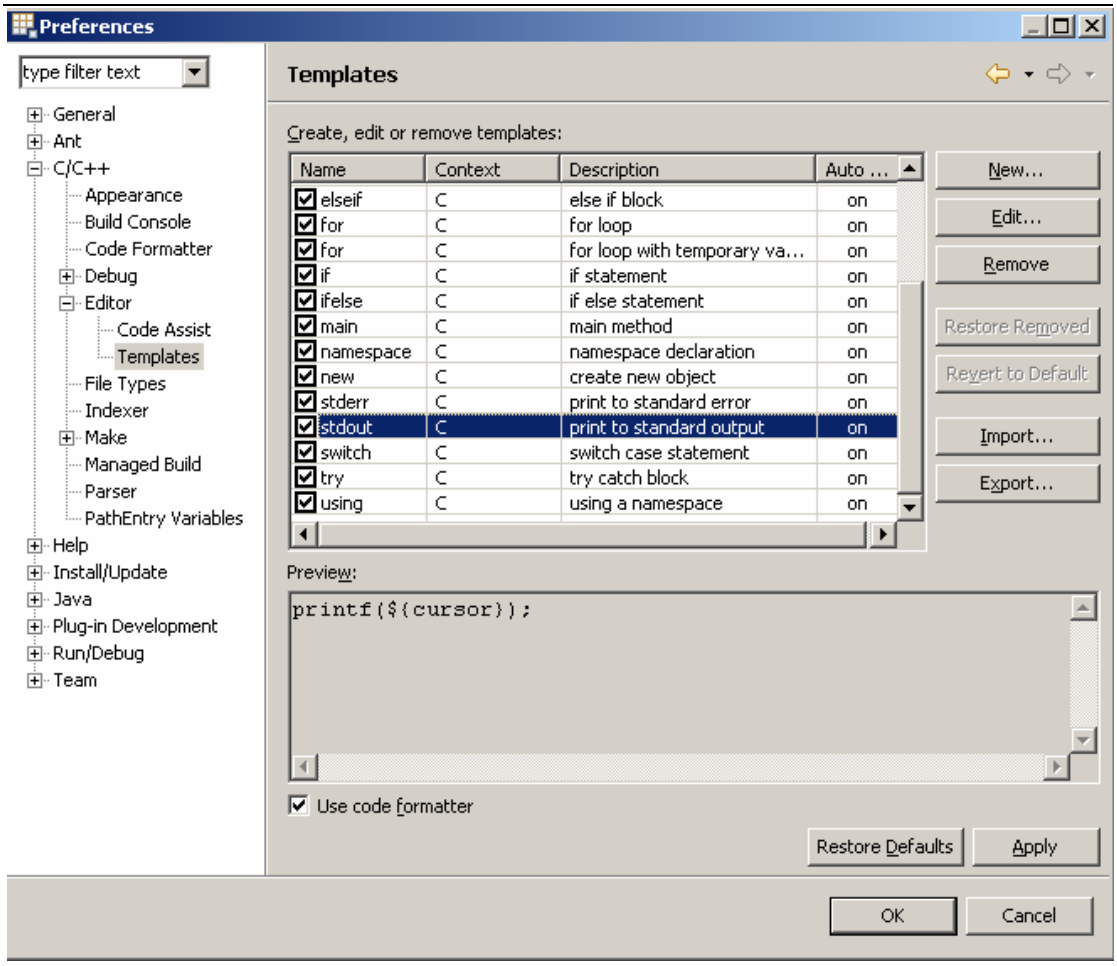


图 6-2 设置代码协助模板

6.2 编译选项的配置

正确配置编译选项是成功通过编译的前提。

配置编译环境方式：选中工程，右键选择“Properties”，再选中左边的 C/C++ Build 项，编译选项设置如图：

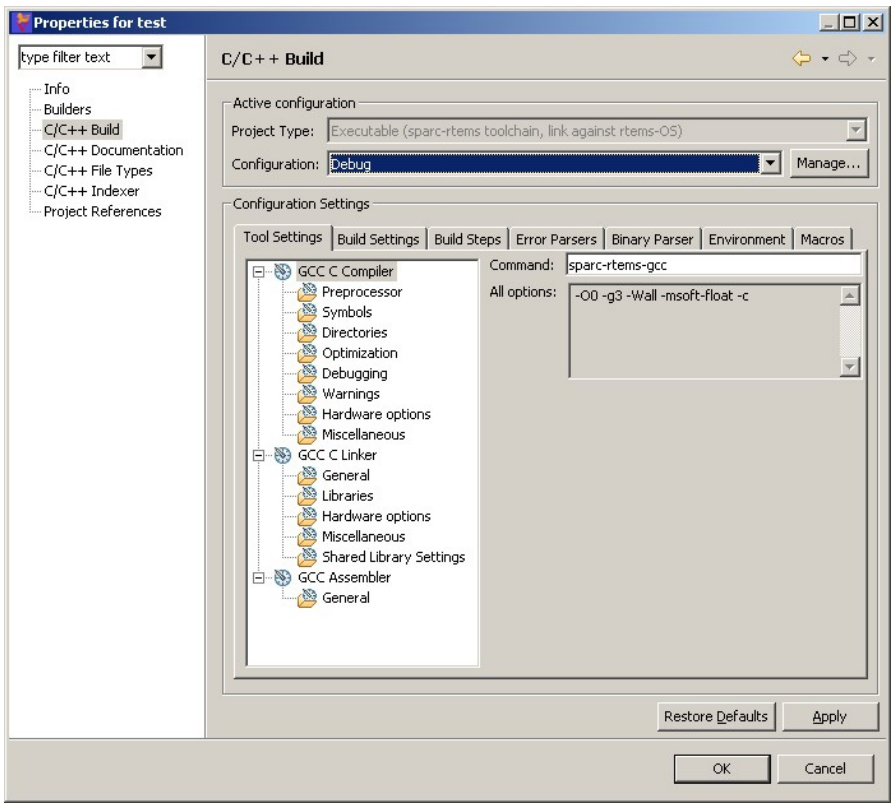


图 6-3 配置编译环境

Configuration 项选择 Debug，其它选项都有默认值，**针对 S698 系列处理器的标准 C 程序可直接使用默认值编译**；单击 Apply 按钮，再单击 OK 按钮完成设置。

部分情况下，需要更改编译选项，例如：

1、 **编译针对 TSC695F 平台的程序**，需分别在 GCC C Compiler 和 GCC C Linker 配置类中的 Miscellaneous 项中分别勾选第四项和第二项(“Generate TSC695 executable”)。

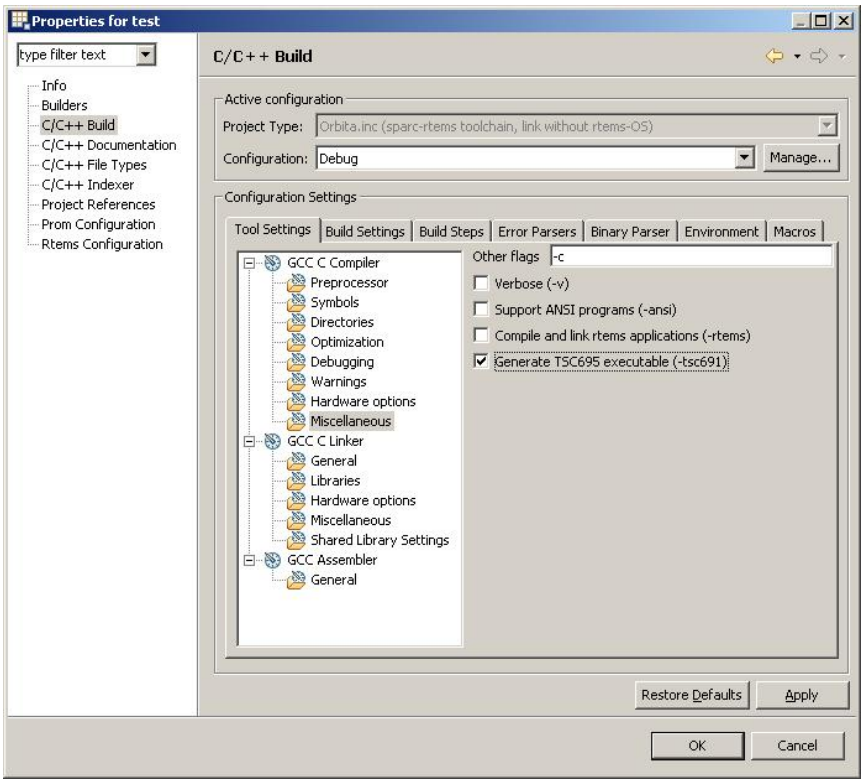


图 6-4 配置 TSC695F 编译参数

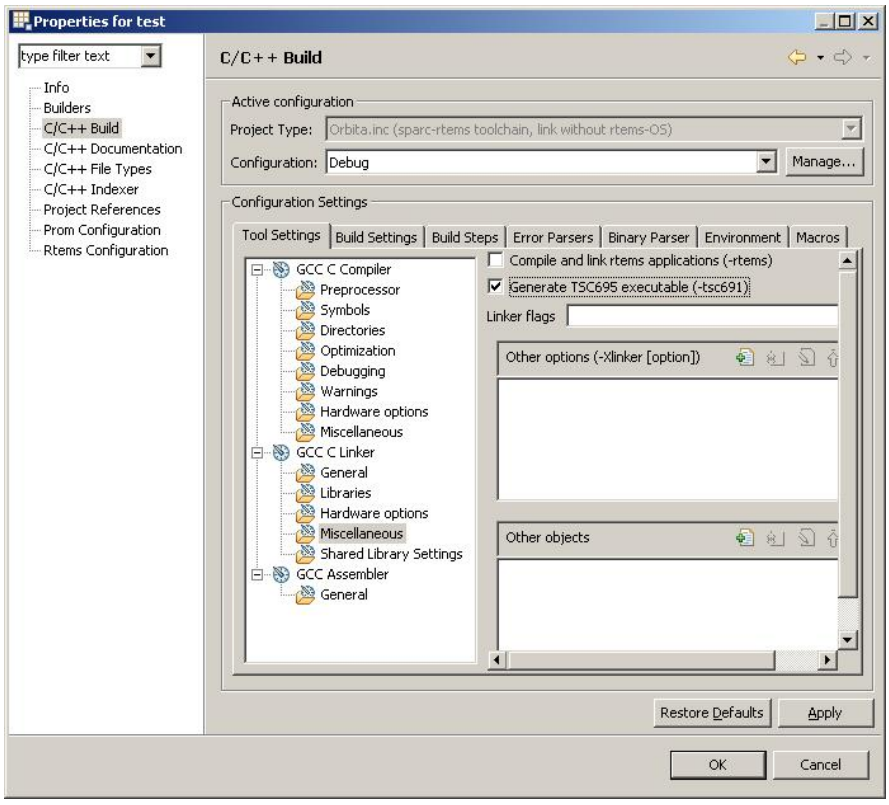


图 6-5 配置 TSC695F 链接参数

2、 程序中有数学函数被调用（如：cos,sin 等），则编译链接时需要添加数学库，添加方法是:在 GCC C Linker 配置类中的 Libraries 项的-I 配置框中点增加按钮，弹出的输入框中输入 m 后点 OK。

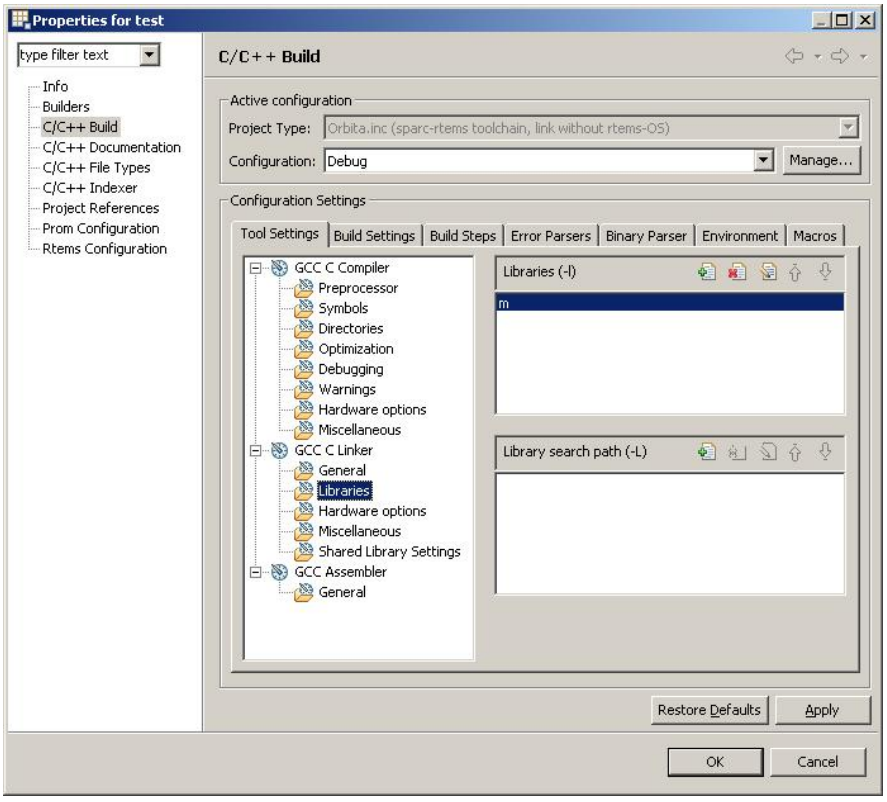


图 6-6 添加数学库

3、 新建的 Bare-c 工程需要更改为带 RTEMS 操作系统的编译，需分别在 GCC C Compiler 和 GCC C Linker 配置类中的 Miscellaneous 选项中分别勾选第三项和第一项(“Compile and link rtems applications”)。**如果是直接生成的 RTEMS C Project，此处无须手工添加，系统已经默认添加。**

其它各编译参数更详细信息见编译器手册 gcc.pdf。

6.3 RTEMS 操作系统的配置

如果用户开发的程序是基于 RTEMS 操作系统的，那么就肯定要涉及到对 RTEMS 的配置，因为 RTEMS 操作系统和其他嵌入式操作系统一样都是可裁减、配置的。用户可以将不需要的模块裁减掉，来减小目标程序的大小；也可以修改感兴趣的配置项，来满足自己的特定需要。这些配置项包括时钟、定时器、文件系统等常用选项。

我们只能对 RTEMS C 类型的工程进行此类配置。具体操作流程如下：

1. 点击右键来选中要配置的工程，在弹出的菜单中选中“Properties”

2. 点中“Properties”后出现的界面中选中“Rtems Configuration”

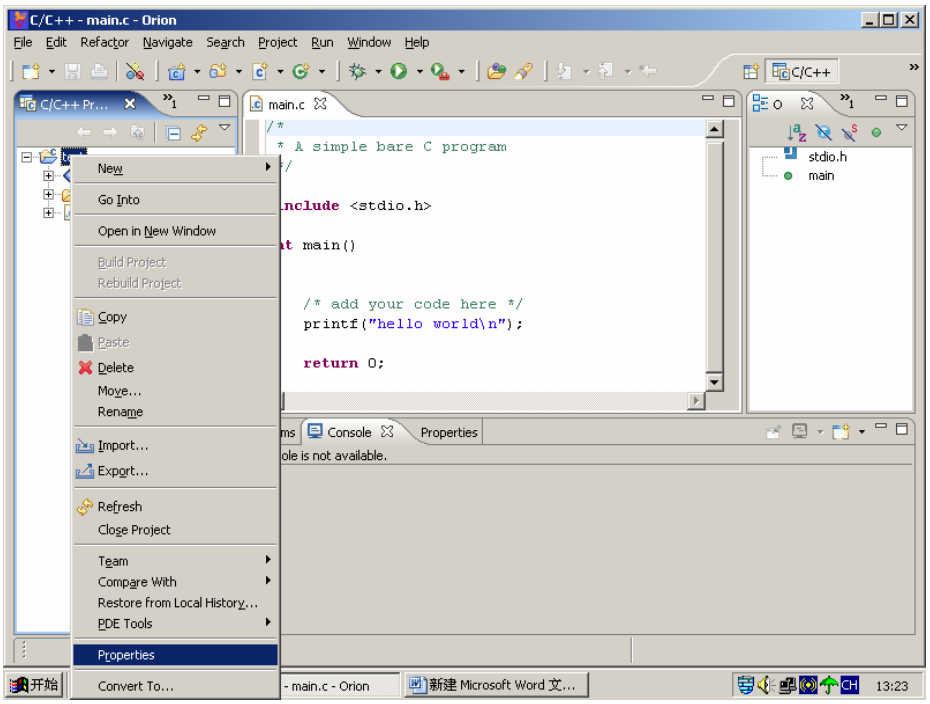


图 6-7 选中“Properties”

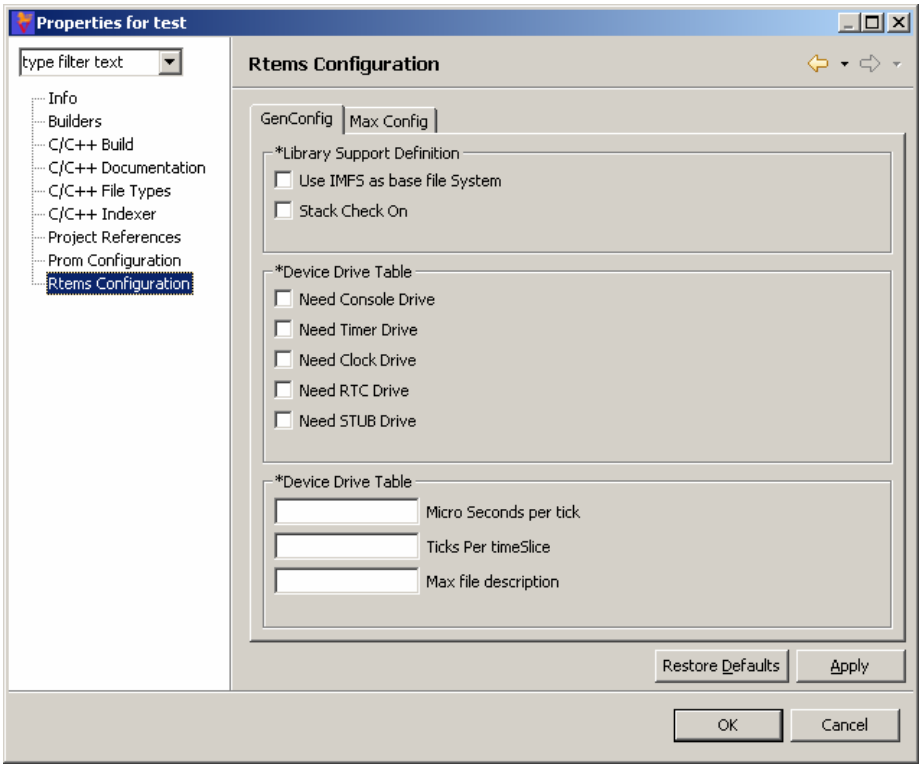


图 6-8 RTEMS 配置页

进入配置页后就可以根据自己的需要配置 RTEMS，点击 Restore Defaults 按钮可以将各个 RTEMS 选项配置为默认值；点击 Apply 按钮可以保存当前的配置状态；点击 OK 按钮也

可以保存当前的配置状态，同时关闭当前的配置窗口；点击 Cancel 按钮可以放弃当前的配置操作。

6.4 程序的编译

程序编辑完成后，点击工具栏中的 Build All 按钮  来编译工程，可以从 Console 控制台中观察到编译信息，编译产生的错误和警告可以从 Problems 视图中察看。

Orion4.0 支持“自动编译”，既保存文件时触发编译，可以点击主菜单 Project 下的 Build Automatically 选项打开自动编译功能。

一般情况下，系统是侦测到代码有改动时才允许编译，若要强制编译可点 Project 菜单下 Clean 项，使系统清空目标文件后再重新编译。

7 程序调试

7.1 调试模式介绍

调试是嵌入式软件开发中的重要环节，Orion4.0 强大的调试功能允许您对程序进行单步跟踪，设置断点，观察变量，察看堆栈等等。

Orion4.0 支持以下的调试方式：

1. Simulator 调试模式（S698、TSC695F 平台适用）；
2. SMON 调试模式（S698 平台适用）；
3. Debug monitor 调试模式（TSC695F 平台适用）。

1. Simulator 调试模式：

调用软件模拟器调试程序，这种方式无需硬件平台，调试方便、简捷，但要注意大多数涉及到硬件处理的程序都无法通过模拟器调试。Orion4.0 开发环境中，针对 S698 系列处理器的模拟器为 Sim-698，针对 TSC695F 处理器的模拟器为 Sim-695。

2. SMON 调试模式：

用于 S698 平台的硬件调试。搭建宿主机/目标板调试环境，主机通过串口与目标板连接，利用 S698 系列处理器的内部 DSU 调试单元，下载程序并运行，以完成交叉调试。这种方式采用真实的运行环境，能及时准确的定位问题。调试时仅需一根串口线，无须外接仿真器，这也正是 S698 系列处理器的一大优势。

采用这种模式时，Orion4.0 将默认调用硬件调试器 V8mon.exe，用户也可以使用第三方硬件调试器 GRMON，具体配置方法见第 7.4 节。

图 7-1 宿主机/目标板调试环境

3. Debug monitor 调试模式：

用于 TSC695F 平台的硬件调试。因为 TSC695F 处理器内部无 DSU 调试单元，需要事先在开发板的 ROM 中烧写 Monitor 程序，调试时 Monitor 负责与主机通讯，完成主机交付的各种调试指令。采用这种模式，主机与目标板之间需要两根串口线连接，一个负责调试通讯，一个负责打印输出。

7.2 调试环境设置

7.2.1 SMON 调试模式

1. 设置硬件调试器 SMON

(1) 选择 Run→External Tools→External Tools.

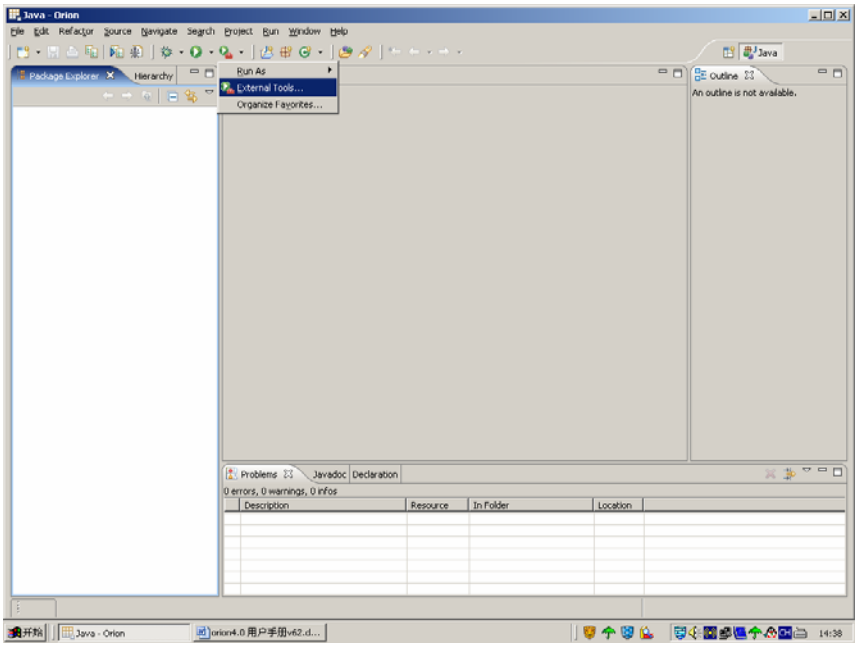


图 7-2 点 External Tools 菜单

(2) 左边的 Configurations 选择框中选中 Default SMON launch(如果没有，需双击 Simulator/SMON launcher 新建，右边 Name 输入框中可任意更改工具名称)，在右边的 Main 页下拉框中选 SMON，检查 Executable 栏中默认值是否：V8mon，Arguments 栏中是否：-gdb -leon2 -i -u。

说明：针对以下这些处理器型号的调试，此处需要增加-leon2 参数：S698、S698-M、

S698-MIL、S698-ECR、S698-S。

注意：如果目标板在下载之前需要用配置文件配置，则 Arguments 栏中需要增加 -c 参数。例如：配置文件为 d:\8，则 Arguments 栏中需输入：**-c d:\8 -gdb -leon2 -i –u**。

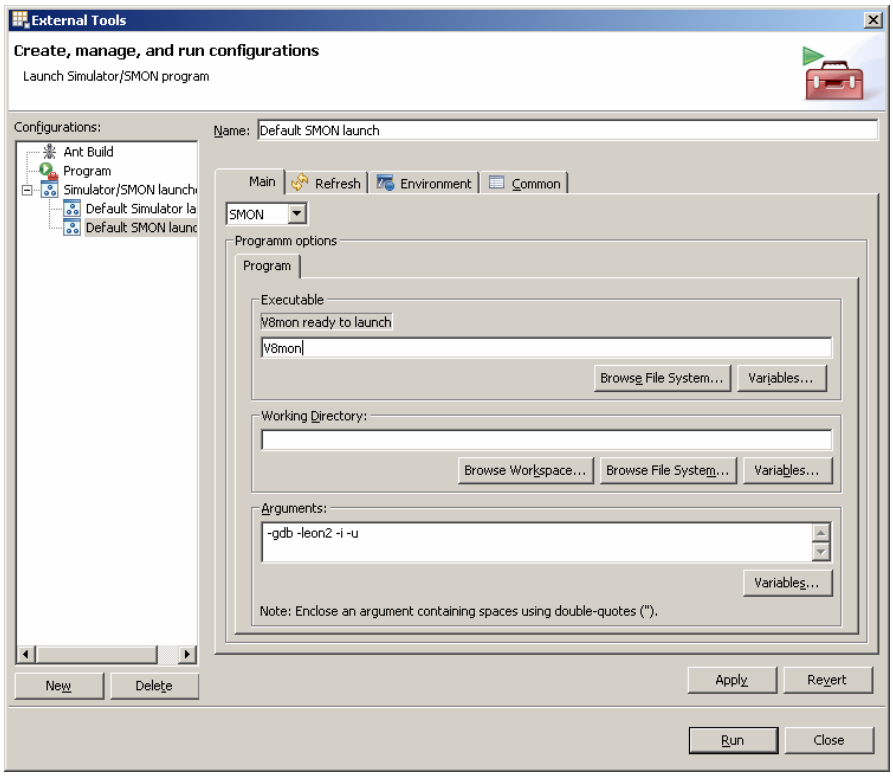


图 7-3 配置 SMON

（3）先点 Apply 按钮，再点 Close 按钮结束设置。

2. 设置调试环境

点击主菜单 Run->Debug…(或者在工程管理视图中点右键，选择 Debug As->Debug…), 系统弹出调试配置对话框；在弹出窗体中的 C/C++ Application running on Simulator/SMON 项中右键选择“New”（或双击）以新建调试环境，调试环境具体设置及说明如下：

(1) Name 输入框中输入新建调试环境的名称，Orion4.0 允许建立多个调试用例，不同的用例之间根据用户自定义的名称区分；

(2) Main 设置页中指定要调试的工程和可执行文件：Project 输入框中输入工程名，一般用默认值；C/C++ Application 输入框中输入可执行文件名，可点击后面的 Search Project… 按钮选择文件；

注意：如果同时有 Debug 和 Release 两个可执行文件时请注意选择，Release 目录下的可

执行文件不包含调试信息，运行时不能进行设断点、单步跟踪等操作；

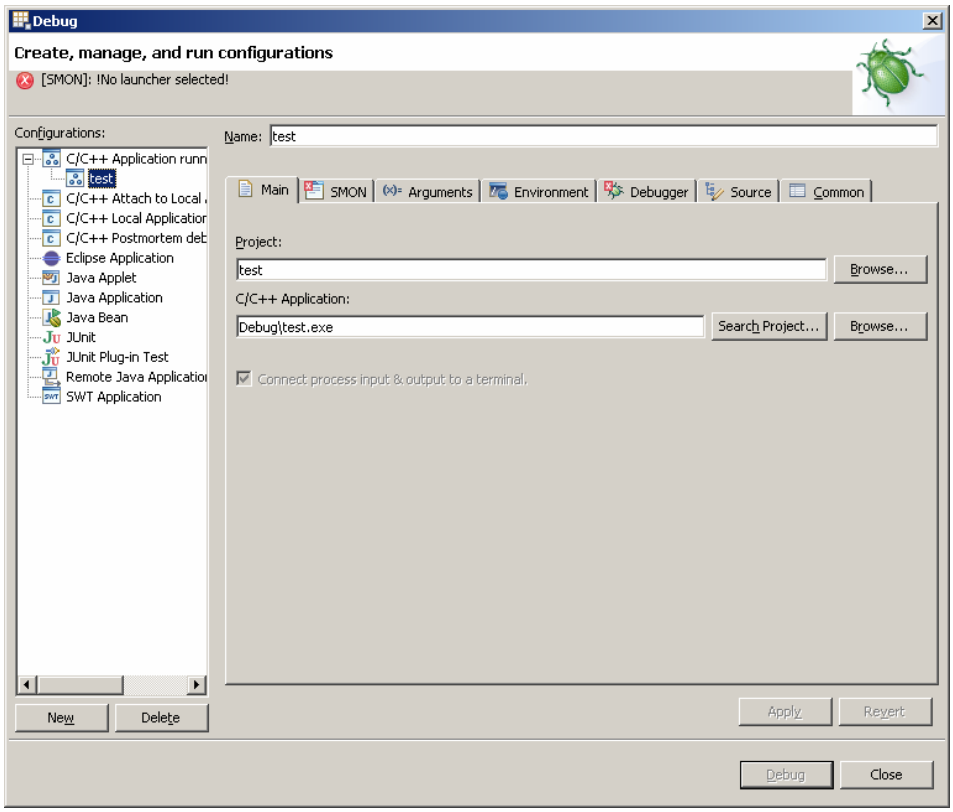


图 7-4 Debug 设置窗中 Main 设置页

(3) SMON 设置页中指定调试方式，以及启动调试时是否内部自动启动 SMON。一般情况下，Start Mode 应选择 Start Simulator/SMON inside Orion，也就是调试时自动启动 SMON；其下的选择框中选中上文配置的硬件调试工具名，如：Default SMON launch；`Terminate SMON/Simulator after debug session terminates` 项打勾。如果选择外部启动 Simulator/SMON，则此处要求输入 Host 和 Port 值，Host 中输入值一般为 localhost；Port 中输入值为 2222。

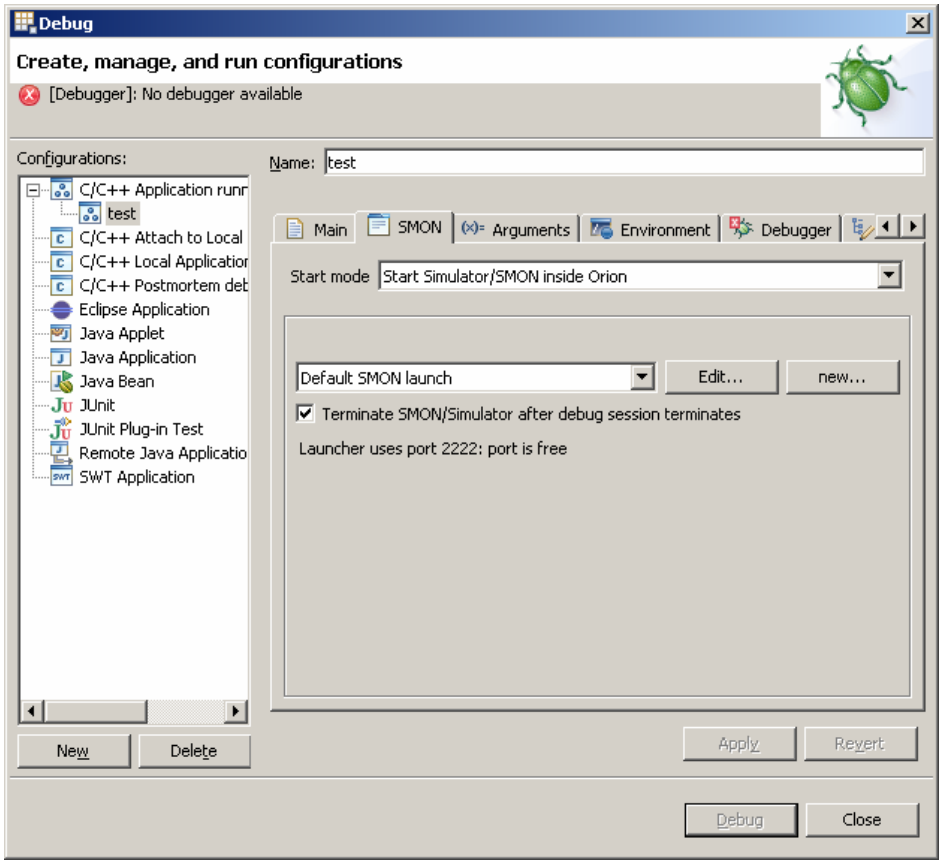


图 7-5 Debug 设置窗中 SMON 设置页

(4) Debugger 设置页中指定了 gdb 调试器的一些参数，具体设置如下：

- Debugger 选择框中选 sparc-rtems GDB Debugger；
- SMON/Simulator port number 中输入 gdb 通讯时对应的 Socket 端口号，端口号更改为 2222；
- 其它文中未提到的设置项请用默认值。

注：勾中 stop at main() on startup 选项表明调试启动后，程序将自动停止在 main()函数的第一条语句处，等待用户的调试；如果想调试启动后直接运行完程序，Debugger 设置页中可以不选中该项。

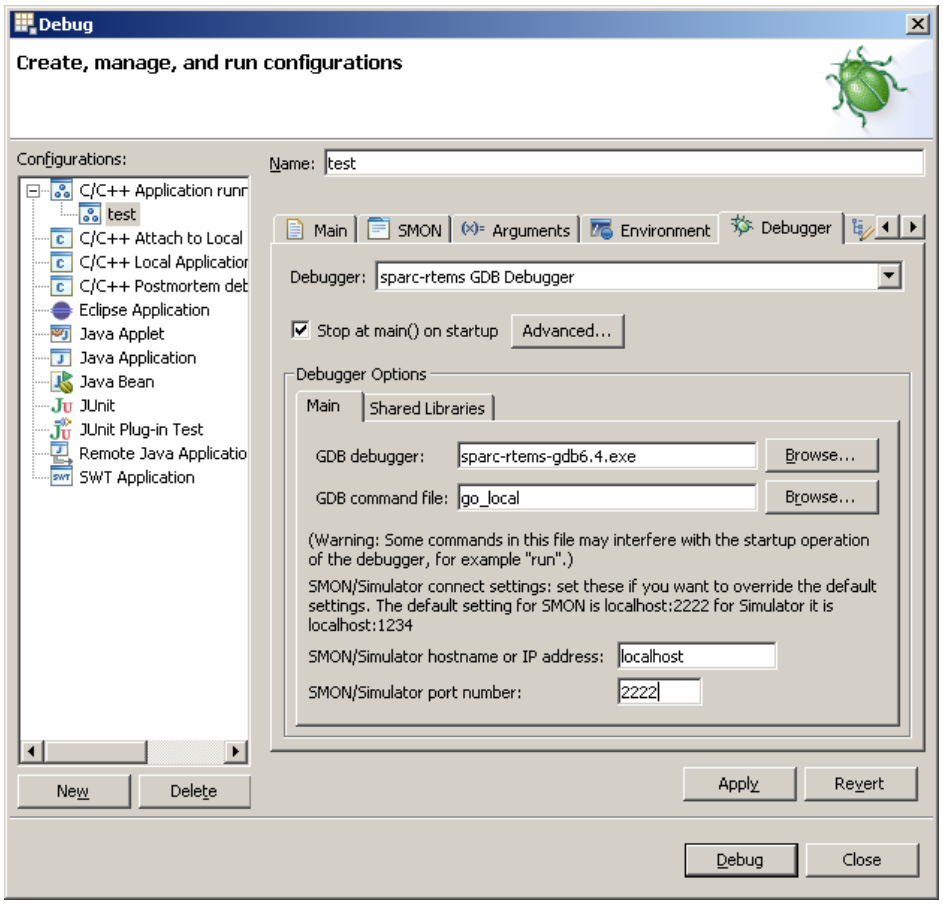


图 7-6 Debug 设置窗中 Debugger 设置页

(5) 先点 Apply 按钮，再点 Close 按钮结束设置。

3. 启动调试

调试环境设置好后可以直接在 Debug 设置窗口单击 Debug 按钮启动调试，也可以在工具栏中点 Debug 按钮  启动调试，此时主界面切换到 Debug 透视图，主界面下方的 Console 输出信息框中提示调试连接的状况，如果连接失败请检查设置选项。**注意：硬件调试前请正确连接硬件目标板并将目标板上电启动。**

7.2.2 Simulator 调试模式

1. 设置软件模拟器 Simulator

（1） 选择 Run→External Tools→External Tools；

（2） 左边的 Configurations 选择框中选中 Default Simulator launch(如果没有，需双击 Simulator/SMON launcher 新建，右边 Name 输入框中可任意更改工具名称)， 在右

边的 Main 页下拉框中选 Simulator，Executable 栏中输入可执行文件名：Sim-698.exe（**TSC695F 的模拟器文件名应为：Sim-695.exe**），在 Arguments 栏中输入 –gdb ；其它项保持默认值；也可以双击 Simulator /SMON launcher 新建。

（3） 先点 Apply 按钮，再点 Close 按钮结束设置。

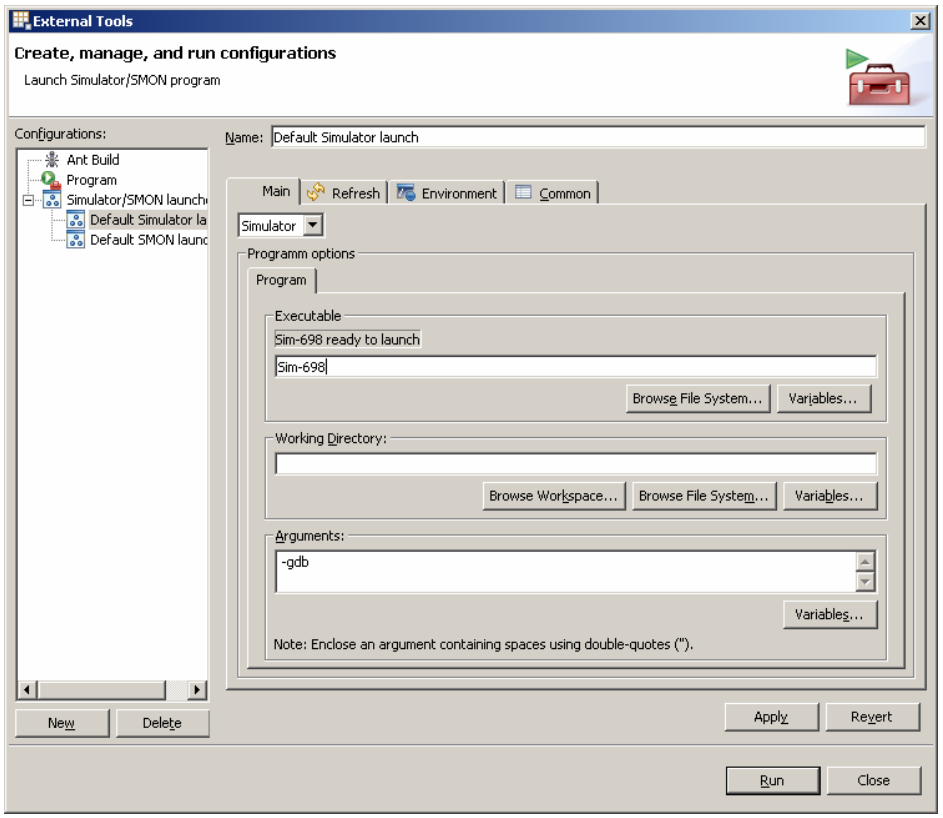


图 7-7 设置 Simulator

2. 设置调试环境

点击主菜单 Run->Debug…(或者在工程管理视图中点右键，选择 Debug As->Debug…)，系统弹出调试配置对话框；在弹出窗体中的 C/C++ Application running on Simulator/SMON 项中右键选择“New”（或双击）以新建调试环境，调试环境具体设置及说明如下：

(1) Name 输入框中输入新建调试环境的名称；

(2) Main 设置页中指定要调试的工程和可执行文件：Project 输入框中输入工程名，一般用默认值；C/C++ Application 输入框中输入可执行文件名，可点击后面的 Search Project… 按钮选择文件；

(3) SMON 设置页中的 Start Mode 应选择 Start Simulator/SMON inside Orion；其下的选择框中选中模拟调试工具名：**Default Simulator launch**（默认项）；`Terminate SMON/Simulator after debug session terminates` 项打勾。

(4) Debugger 设置页中指定了 gdb 调试器的一些参数，具体设置如下：

- Debugger 选择框中选 sparc-rtems GDB Debugger；
- SMON/Simulator port number 中输入 gdb 通讯时对应的 Socket 端口号，端口号更改为 1234；
- 其它文中未提到的设置项请用默认值。

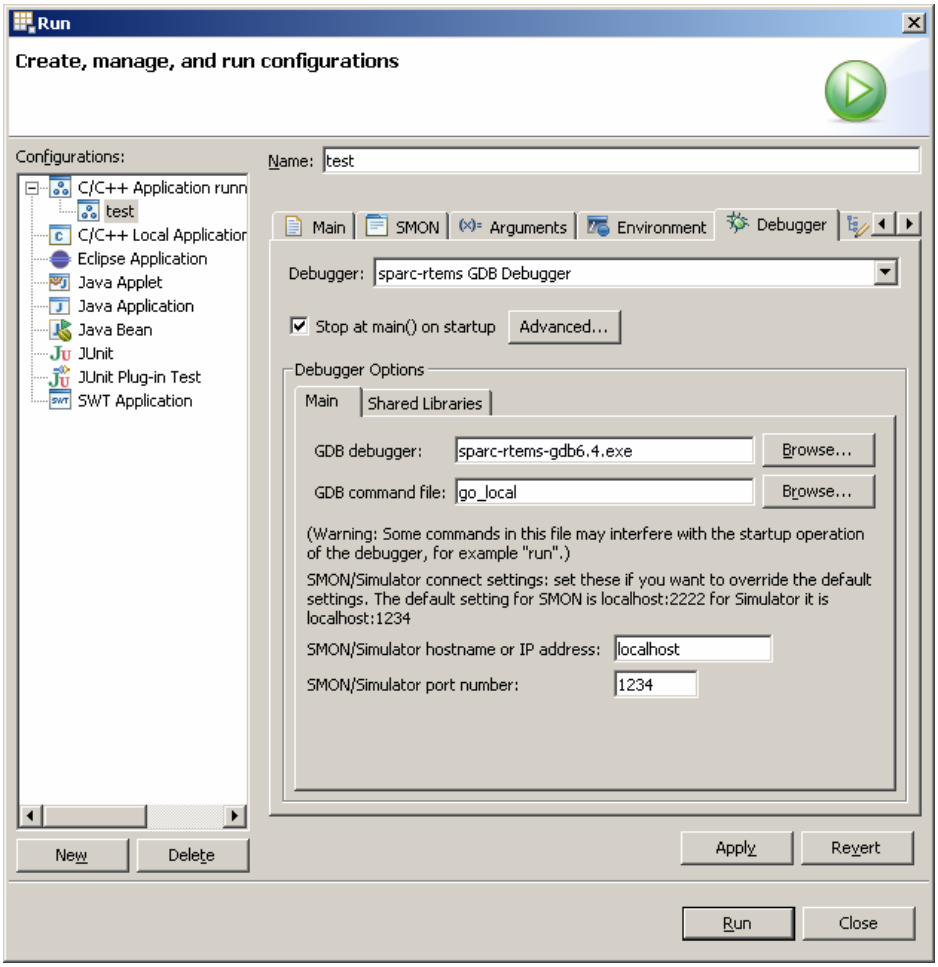


图 7-8 Simulator 调试模式时调试环境的设置

3. 启动调试

调试环境设置好后可以直接在 Debug 设置窗口单击 Debug 按钮启动调试，也可以在工具栏中点 Debug 按钮  启动调试，此时主界面切换到 Debug 透视图，主界面下方的 Console 输出信息框中提示调试连接的状况，如果连接失败请检查设置选项。

7.2.3 Debug monitor 调试模式

1. 设置调试环境

点击主菜单 Run->Debug…(或者在工程管理视图中点右键，选择 Debug As->Debug…)，系统弹出调试配置对话框；在弹出窗体中的 C/C++ Application running on Simulator/SMON 项中右键选择“New”（或双击）以新建调试环境，调试环境具体设置及说明如下：

(1) Name 输入框中输入新建调试环境的名称；

(2) Main 设置页中指定要调试的工程和可执行文件：Project 输入框中输入工程名，一般用默认值；C/C++ Application 输入框中输入可执行文件名，可点击后面的 Search Project… 按钮选择文件；

(3) SMON 设置页：Start Mode 选择 Connect to external Simulator/SMON or a uart debug monitor；Type 框选择 tty；Device 框中输入 PC 机串口一的设备号/dev/ttyS0(若使用串口二则设备号为/dev/ttyS1)；Baudrate 框中输入通讯波特率，波特率应与 Debug Monitor 相匹配。

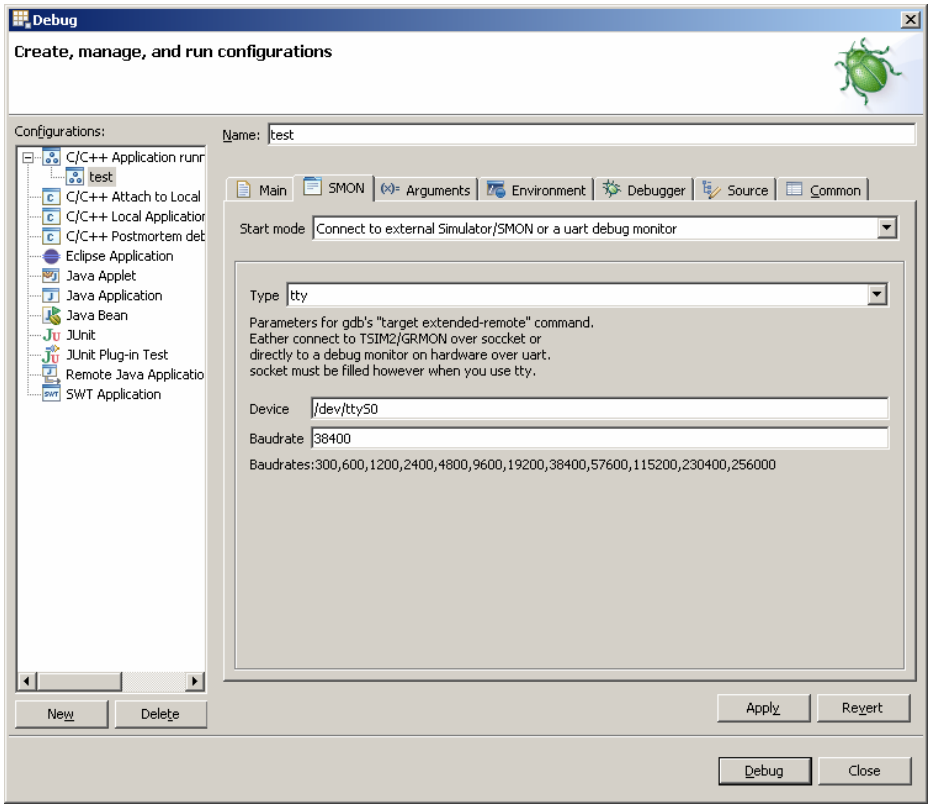


图 7-9 Debug monitor 调试模式时调试环境的设置 1

(4) Debugger 设置页中指定了 gdb 调试器的一些参数，具体设置如下：

- Debugger 选择框中选 sparc-rtems GDB Debugger；
- 其它文中未提到的设置项请用默认值。

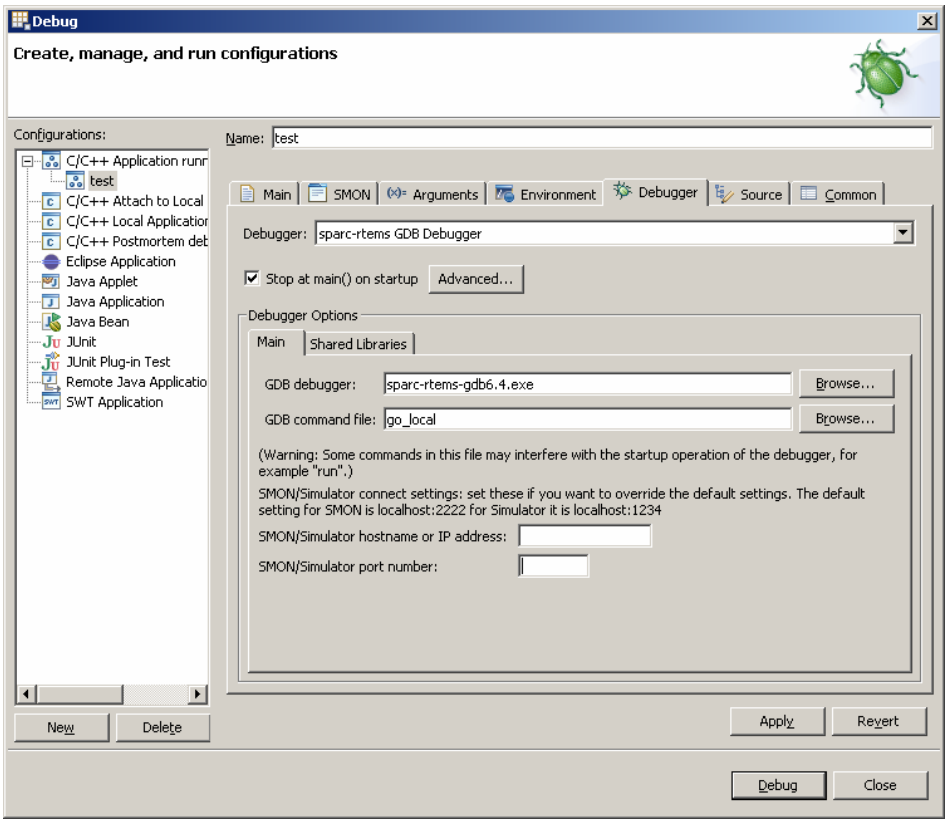


图 7-10 Debug monitor 调试模式时调试环境的设置 2

2. 启动调试

调试环境设置好后可以直接在 Debug 设置窗口单击 Debug 按钮启动调试，也可以在工具栏中点 Debug 按钮  启动调试，此时主界面切换到 Debug 透视图，主界面下方的 Console 输出信息框中提示调试连接的状况，如果连接失败请检查设置选项。

注意：调试之前要对开发板进行正确的连接，并观察 Debug Monitor 是否已经正确启动（可先借助串口助手打开 PC 机的串口二，在开发板上电或复位时观察有无提示信息输入）。

7.3 调试方法

1. 设置断点

设置断点通常有以下几种方法（以下(2)、(3)在 Debug 透视图下才有效）：

(1) 将鼠标移动到目标代码行左边的标记栏，点右键，在浮动菜单中选择 Toggle Breakpoint；

(2) 将光标移动到需要设置断点的代码行处，选择菜单 Run/Toggle Line Breakpoint；

(3) 将光标移动到需要设置断点的代码行处，按快捷键 CTRL+SHIFT+B；

删除断点的方法如此类似，当程序运行到断点处时，会停止在有效断点处，源程序左边出现指示箭头；建议调试开始之前先设置好断点，可防止一启动调试后程序就立刻运行完毕。

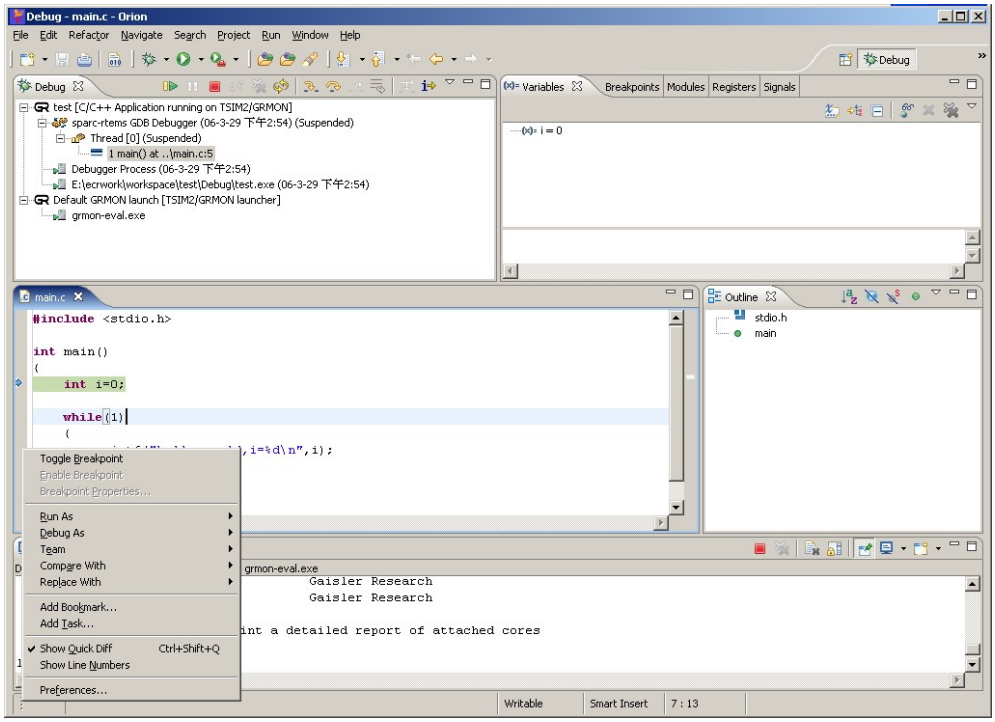


图 7-11 通过浮动菜单设置断点

2. 单步运行

程序运行到断点处后，可单击 Step Into 按钮  或 Step Over 按钮  进行单步运行，其中 Step Into 是运行到函数处会进入函数内部，Step Over 则不会进入函数内部直接运行完函数；源程序左边的箭头指示当前正待运行的程序行。

3. 切换到汇编调试

调试过程中，单击 Debug 视图上的 Instruction Stepping Mode 按钮 将出现反汇编窗口，这时进行单步操作将执行汇编语句单步；再次单击 Instruction Stepping Mode 按钮将切换到 C 语言语句单步。

4. 观察变量

选择 Windows/Show View/Variables 项，可打开变量观察窗，在其中点击右键可进行添加删除观察变量的操作，Format 项用来选择数据显示的方式（10 进制、16 进制）；如果所观察的变量值改变，则系统以红色显示该变量。

5. 观察内存

选择 Windows/Show View/Memory 项，可打开内存观察窗，单击 Add Memory Monitor 按钮输入要观察的内存起始地址（建议输入十六进制如：0x40000000），点 OK 按钮后，右边界面将显示当前内存地址的值，数据以 16 个字节为一行依次显示，拖动滚动条系统将按照对应地址提取数据显示，内存显示如图：

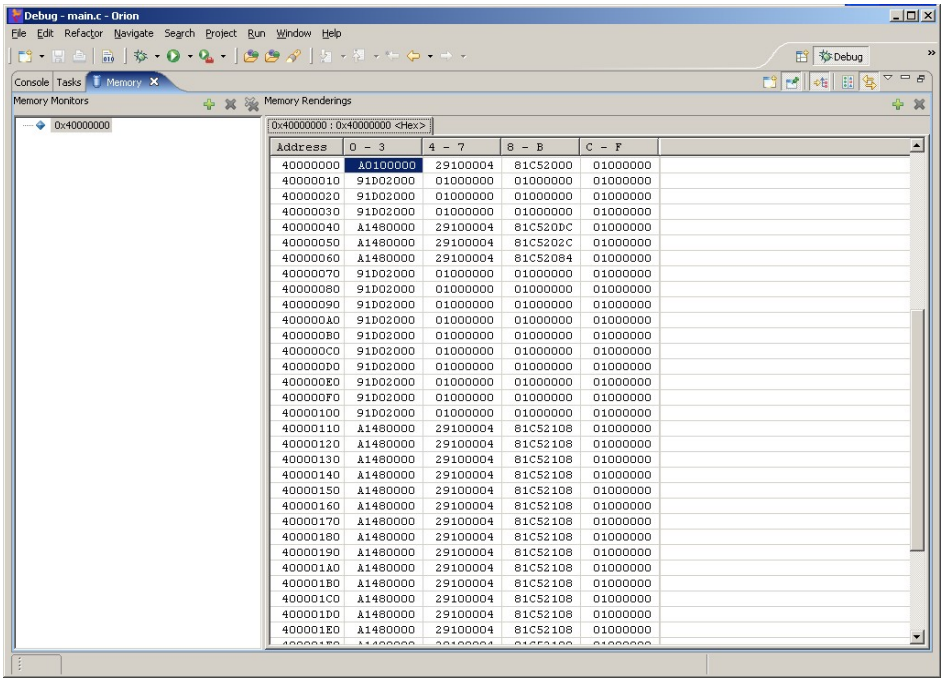


图 7-12 观察内存

6. 观察寄存器

选择 Windows/Show View/Registers 项，可打开寄存器观察窗，如下图；随着程序运行

到不同的程序行，可以观察到各寄存器所产生的变化；如果寄存器值改变，则系统以红色显示该寄存器。

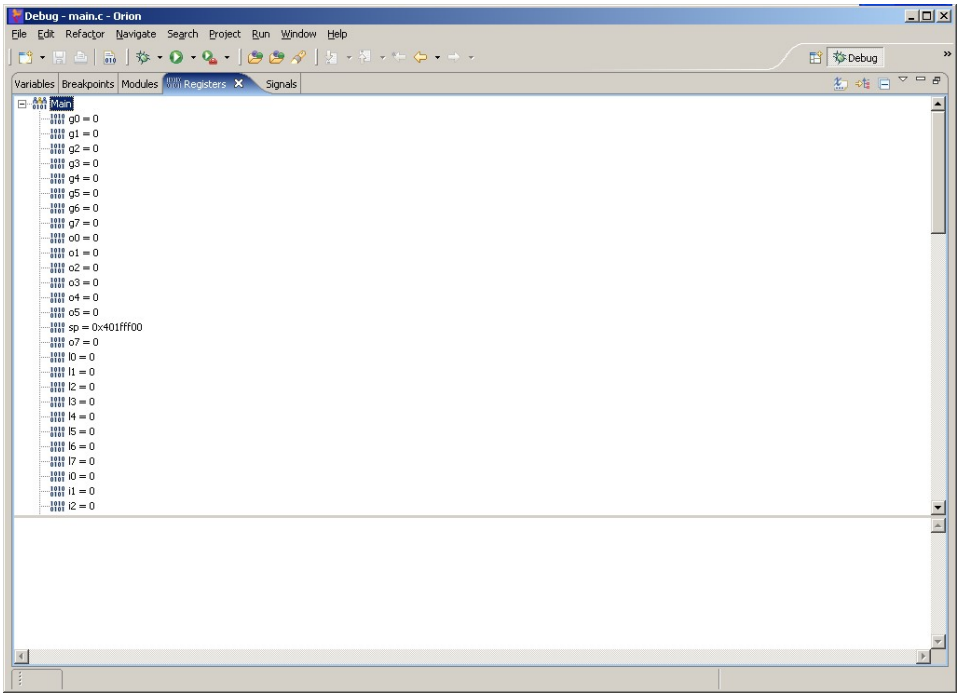


图 7-13 观察寄存器

7. 观察程序运行结果

程序调试\运行过程中单击控制台右上角的 Display Selected Console 按钮  中右方的三角符，在浮动菜单中选择 Sim-698 / Sim-695/V8mon，可以观察程序的执行时输出的结果；点 Pin Console 按钮  可将控制台锁定在输出显示。

8. 删除调试进程

调试完成或者中途需要重新开始调试时要将当前的调试进程关闭；单击 Debug 窗口或者 Console 窗口右上方的 Remove All Terminated Launches 按钮  可完成删除操作，如果调试进程仍在运行需要先单击 terminate 按钮  终止调试后再进行删除。

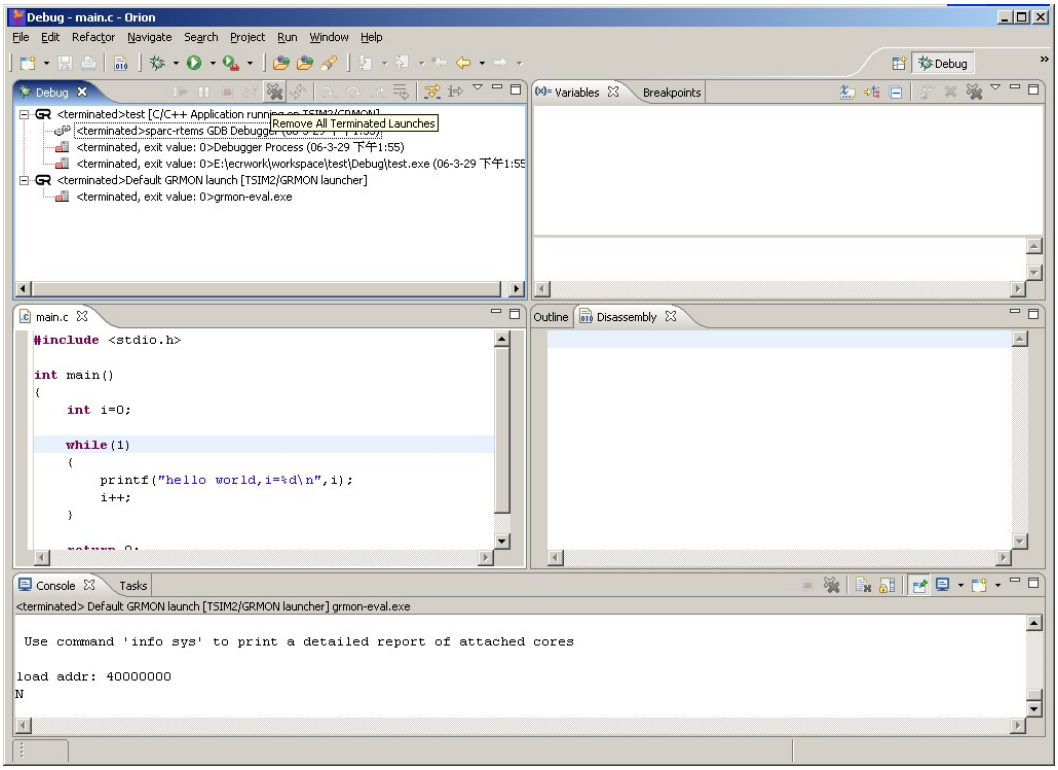


图 7-14 删除已经启动的调试

7.4 对 GRMON 的支持

7.4.1 GRMON 简介

GRMON 是一款针对 SPARC V8 架构处理器的硬件调试器，目前在国际上得到了较广泛的使用，GRMON 包括以下功能：

- 读/写访问所有系统寄存器和存储器；
- 内置的反汇编功能和跟踪缓冲管理；
- 下载和执行应用程序；
- 设置断点和查看断点管理器；
- 远程连接 GNU 调试器(gdb)。

Orion4.0 中可以支持 Windows 平台下的 GRMON 评估版和专业版。

7.4.2 针对 GRMON 的配置

使用 GRMON 调试时，Orion4.0 将 GRMON 作为外部工具自动调用，其配置方法是：

(1) 打开 External Tools 外部工具配置页；

(2) 左边的 Configurations 选择框中选中 Default SMON launch(如果没有，需双击 Simulator/SMON launcher 新建，右边 Name 输入框中可任意更改工具名称)，在右边的 Main 页下拉框中选 SMON，Executable 栏中输入 GRMON 的可执行文件名：专业版为 grmon.exe，评估版为 grmon-eval.exe。

(3) 先点 Apply 按钮，再点 Close 按钮结束设置。

其他操作步骤参考 SMON 调试模式操作说明。

注意：使用之前需确认 GRMON 已安装成功，且系统路径已正确添加。

8 ROM 映像文件的生成和烧写

代码调试无误后，就可以生成 ROM 映像文件烧制到目标系统的 ROM 或者 Flash 芯片中。烧写完成后，重新上电，程序可自动运行。

从编译生成的可执行文件到 ROM 映像文件还需要一些过程，Orion4.0 具备自动生成 ROM 映像文件和在线烧写的功能。

使用 Orion4.0 生成和烧写 ROM 映像文件可以有图形界面和命令行两种方式。下文的操作步骤中假设使用的硬件系统为 Orbita SPARC V8 开发系统，编译完成的目标文件为 test.exe。

8.1 ROM 映像文件功能原理

ROM 映像文件的 boot 部分在系统启动后首先被运行，它负责处理以下的工作：

- 初始化 IU & FPU；
- 通过 CPU 的控制寄存器，初始化内存参数；
- 把程序解压缩到 ram(使用修改过 LZSS 算法)；
- 设置内存顶部，堆栈指针，然后运行程序。

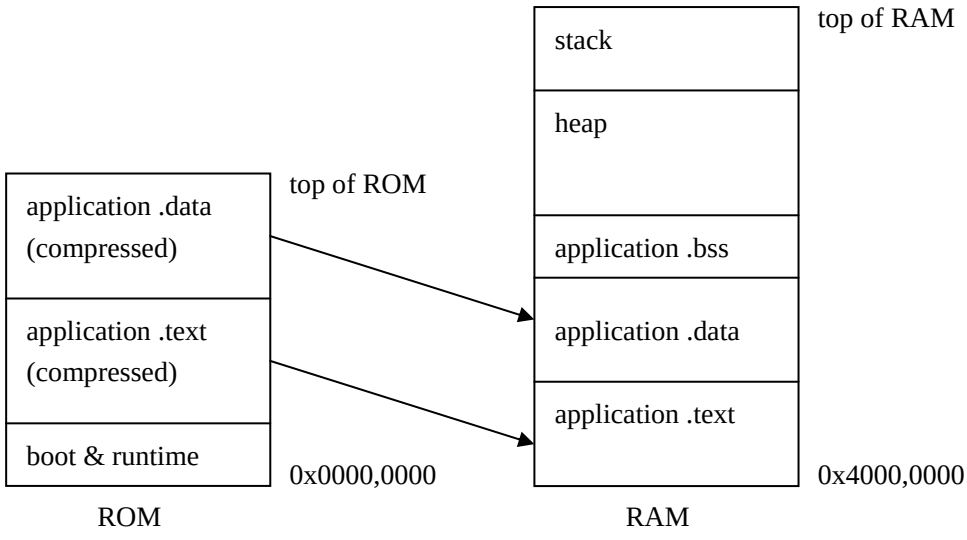


图 8-1 ROM 映像文件内存使用示意图

8.2 命令行方式

首先双击桌面的快捷方式 cygwin.bat 打开 Cygwin 命令行。

ROM 映像文件的生成、烧写过程如下（下文中‘**$**’表明是 Cygwin 命令，‘v8mon>’表明是 V8mon 中的执行命令）：

1. 将目标文件转换为 prom 文件

命令：

```
$ mkprom.exe -freq 20 -baud 38400 -ramsize 1024 -ramcs 2 -ramws 15 -ramwidth 8 -romsize 512 -romwidth 8 test.exe -o a.prom
```

参数说明：详见 mkprom-1.3.6.pdf

2. 将 prom 文件转换为 bin 格式

命令：

```
$ sparc-rtems-objcopy -O binary a.prom a.bin
```

3. 生成烧写文件

命令：

```
$ mkflash -sst -width 8 a.bin -o flash.out
```

参数说明：使用命令 mkflash --help 查看

4. 程序烧写

命令：

A．S698 平台：通过 V8mon 下载、运行烧写文件

```
$ V8mon.exe -i –u
v8mon> lo flash.out
v8mon> ru
```

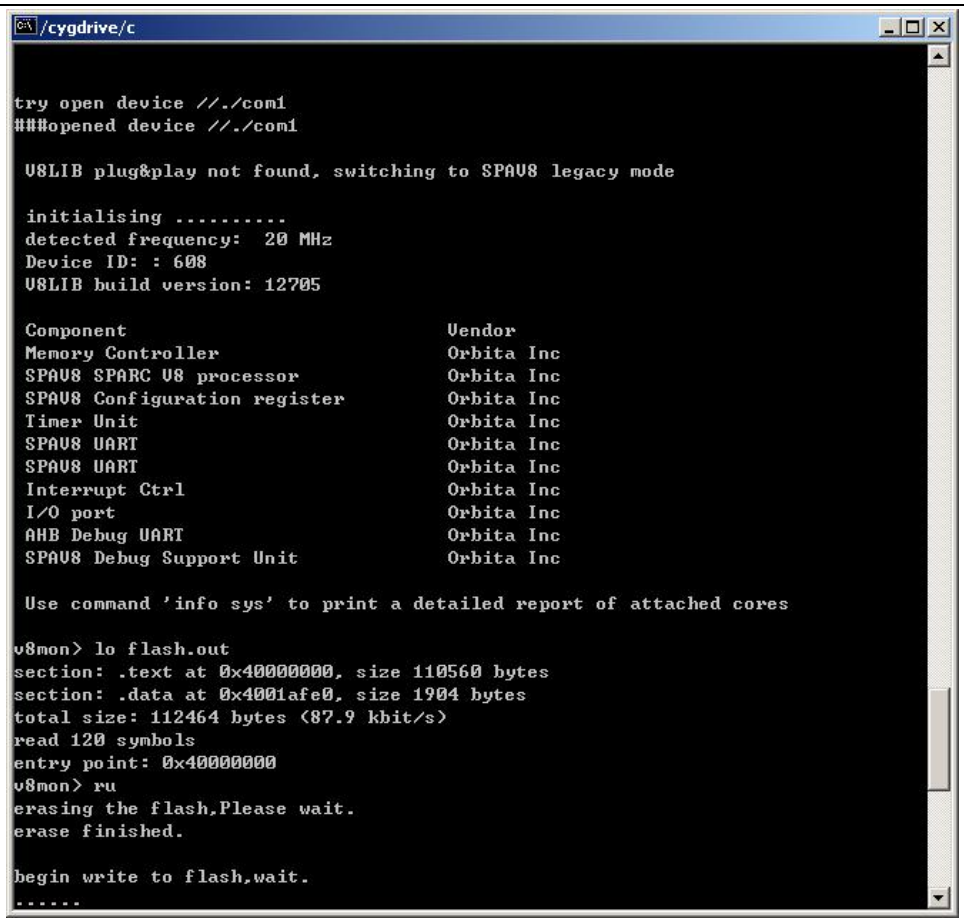


图 8-2 通过 V8mon 烧写 ROM 映像文件

B．TSC695F 平台：通过 gdb 下载、运行烧写文件

```
$ sparc-rtems-gdb
gdb> file flash.out
gdb> set remotebaud 38400
gdb> target extended-remote /dev/ttyS0
gdb> load
gdb> run
```

8.3 图形界面方式

1、参数配置

1) 点击右键来选中要配置的工程，在弹出的菜单中选中“Properties”。如下图所示：

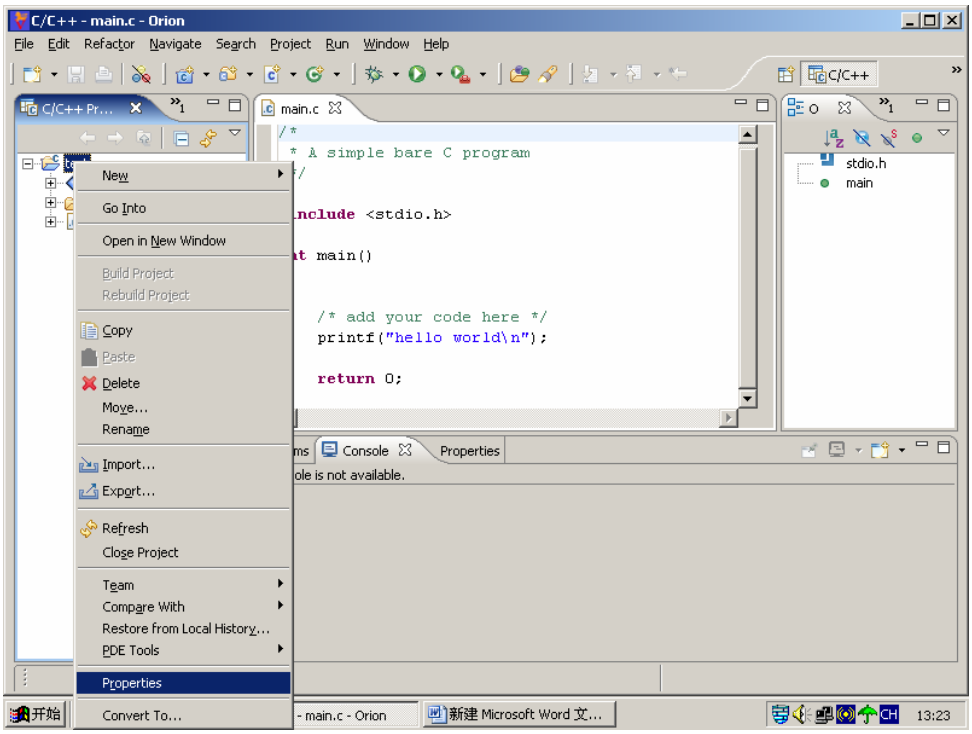


图 8-3　选中“Properties”

2)　点中“Properties”后的界面中选中“Prom Configuration”项，进入了 Prom 的配置页。

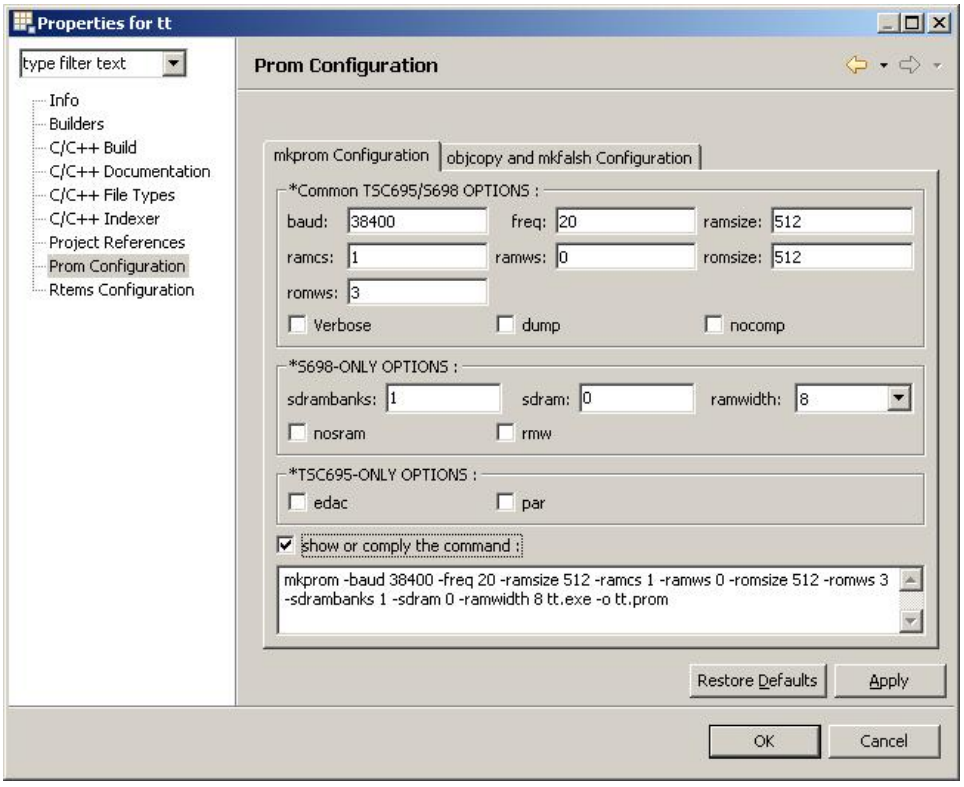


图 8-4 Prom 的配置页

3)　在此界面中有两个标签页“mkprom Configuration”和“objcopy and mkflash

Configuration”，前者用来配置 mkprom 命令和其参数，后者用来配置 objcopy 和 mkflash 命令和其参数。命令常用的选项我们都可以在界面中找到，如果用户自己编辑命令和参数，例如要编辑 objcopy 命令，可以选中“show or comply the objcopy command”项，在其下的文本框中编辑，如下图所示：

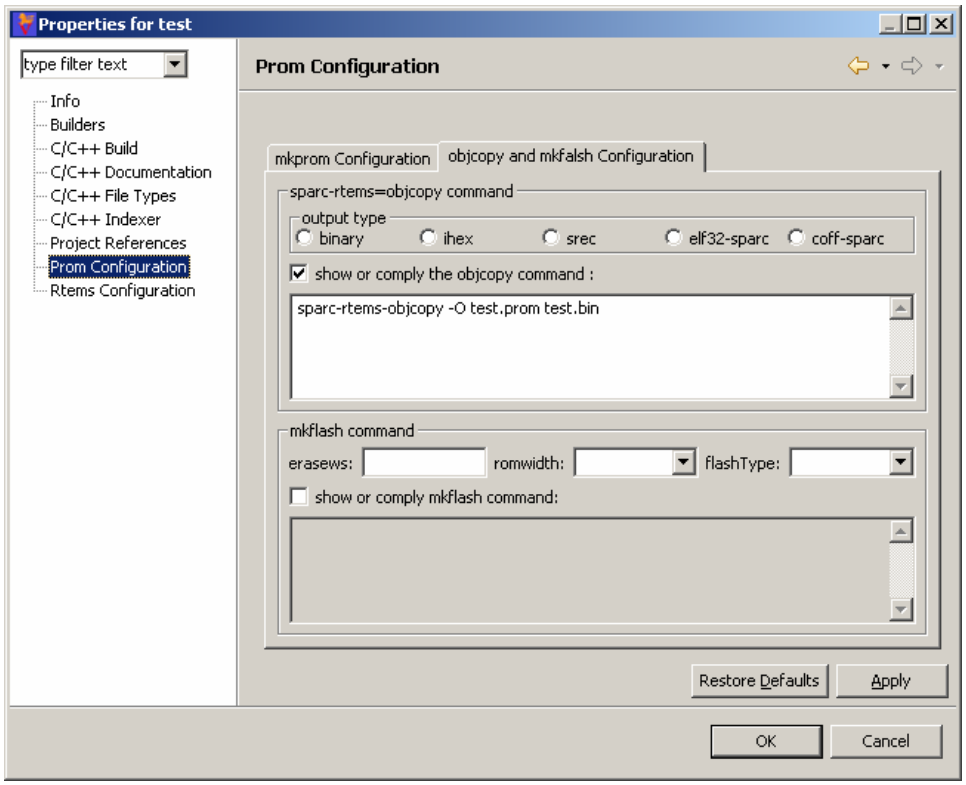


图 8-5 手工配置 objcopy 命令

4) 各个命令配置完毕后，点击 Apply 按钮保存当前命令为相应的脚本文件：flash.bat ，batch.bat，dsu.bat。

2、文件生成与烧写

1） 选中 Project 菜单下的 Burn Flash 子菜单，系统弹出 Flash 烧写操作界面；

2） 勾选要操作的工程(如图 8-6 所示)，单击 Generate Image 按钮就能生成最终的烧写程序 flash.out，通过 Console 控制台可以观察到文件生成过程，同时在 Debug/ 目录下可以看到已经生成的 flash.out；

单击 Burn 按钮系统会自动将 flash.out 文件下载到目标板并运行，请先确定已经用 PC 的串口一连接了目标板，并打开了电源。这时会弹出一个 Dos 窗口来显示运行状况，执行完毕就会自动关闭。**注意：**TSC695F 平台不支持图形界面自动烧写功能。

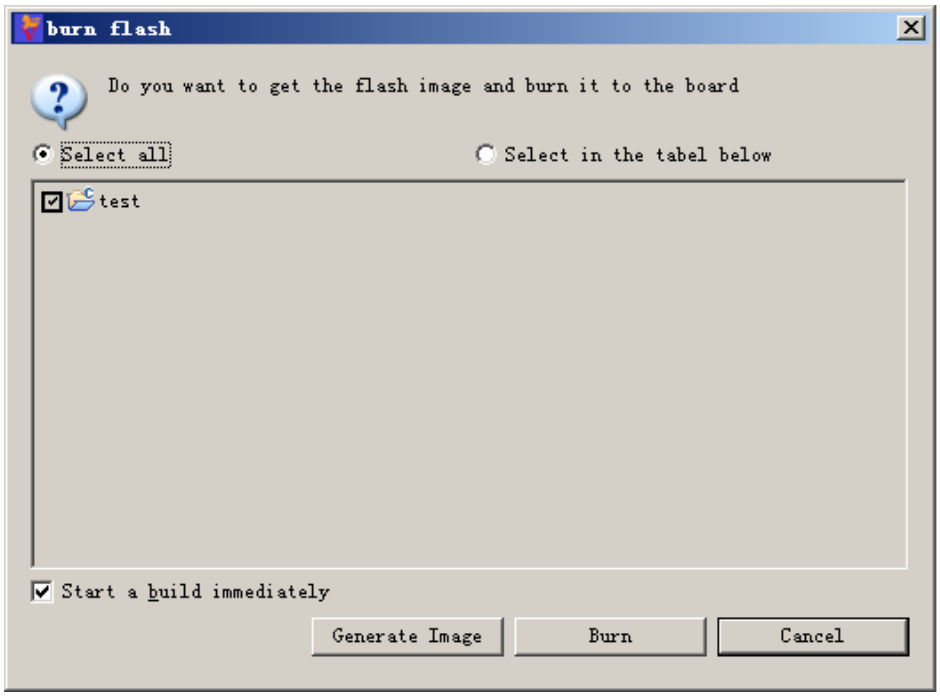


图 8-6 程序烧写操作界面

9 开发实例

下面我们用一个开发实例来展示 Orion4.0 的开发流程和操作步骤。这个实例计划运行于 S698 硬件平台，不带 RTEMS 操作系统，其运行结果是循环打印字符串“Just a test”。

9.1 新建工程

操作步骤：

(1) 点击主菜单 File->New->Project，弹出工程创建对话框；

(2) Select a wizard 页：选中 Orion C 中的 **Bare C Project**，单击 next 按钮；

(3) Managed Make C Project 页：输入工程名“Test”，单击 next 按钮；

(4) Select a type of project 页：单击 next 按钮；

(5) Additional Project Settings 页：单击 Finish 按钮。

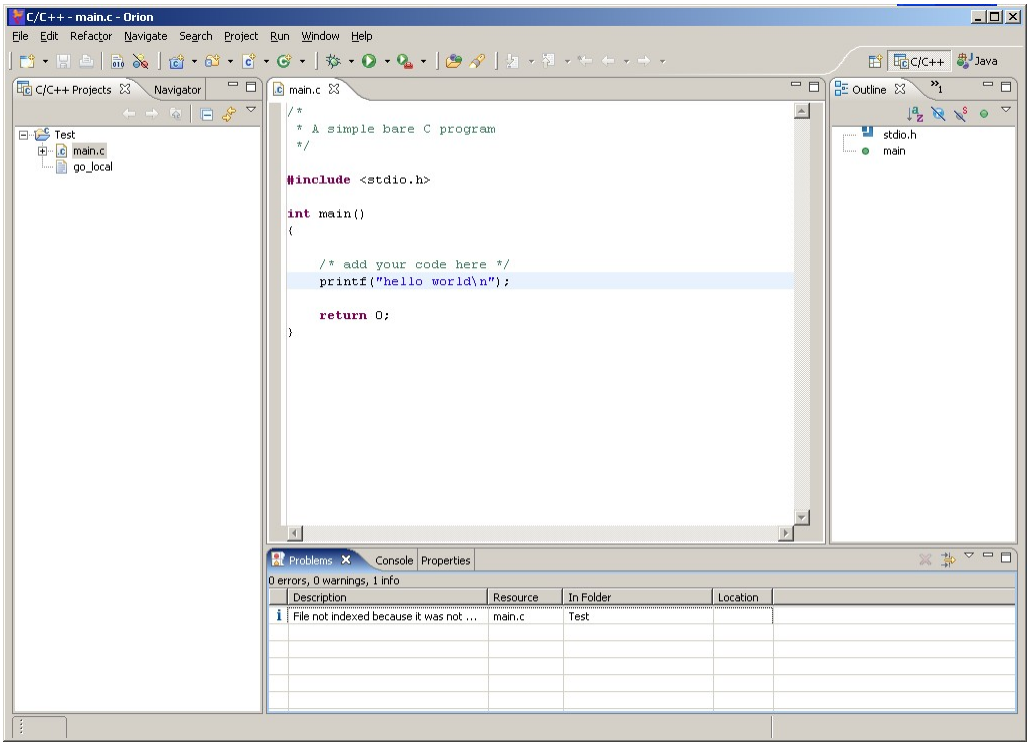


图 9-1 新建的工程

9.2 程序编辑

新建完成后，系统自动添加了 main.c 文件，这里我们把 main 函数作如下修改：

```
int main()
{
    /* add your code here */
    int i=0;

    for(i=0;i<100;i++)
    {
       printf("Just a test\n");
    }

    return 0;
}
```

9.3 程序编译

因为是针对 S698 平台的 Bare-c 工程，编译之前无须配置编译参数，直接点击工具栏中的 Build All 按钮来编译工程。

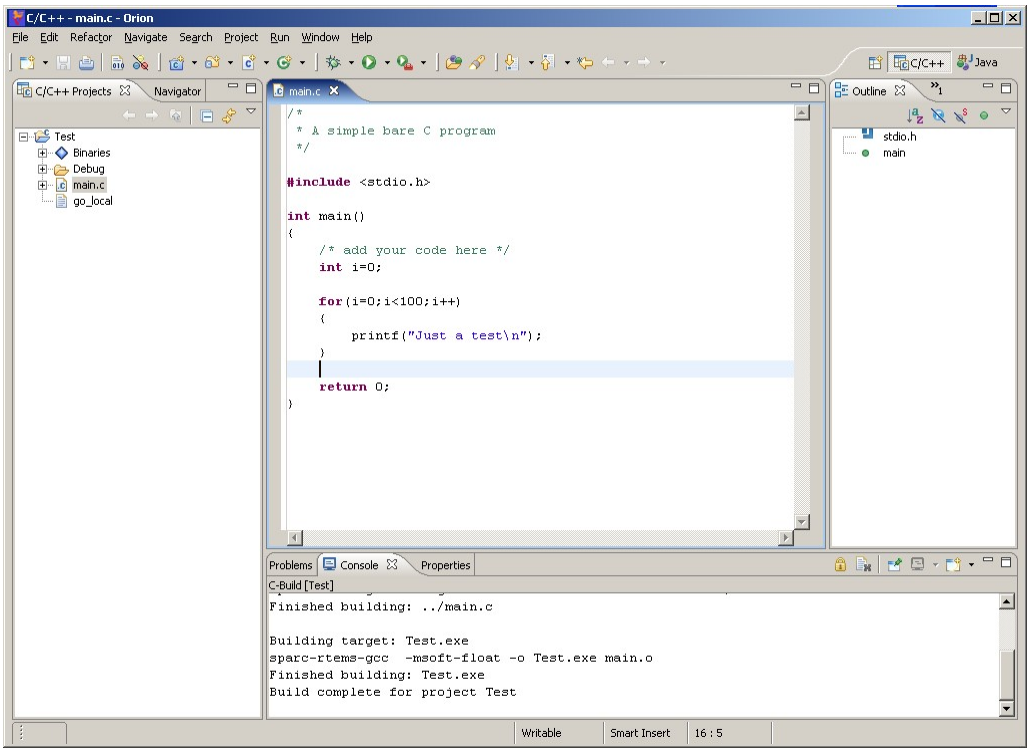


图 9-2 编译完成的工程

9.4 调试环境设置

编译完成后，我们采用 SMON 模式对程序进行硬件在线调试，调试之前需要对调试环

境进行一些配置。

操作步骤：

(1) 点击主菜单 Run->Debug…，弹出调试配置对话框；

(2) 在左边的 Configurations 列表中双击 **C/C++ Application running on Simulator/SMON** 项，系统新建了调试环境 Test；

(3) Main 设置页：点击 Search Project…按钮，选择待调试的可执行文件 **Test.exe**；

(4) SMON 设置页：**`Terminate SMON/Simulator after debug session terminates`** 项打勾，其上方的选择框中选 **Default SMON launch**；

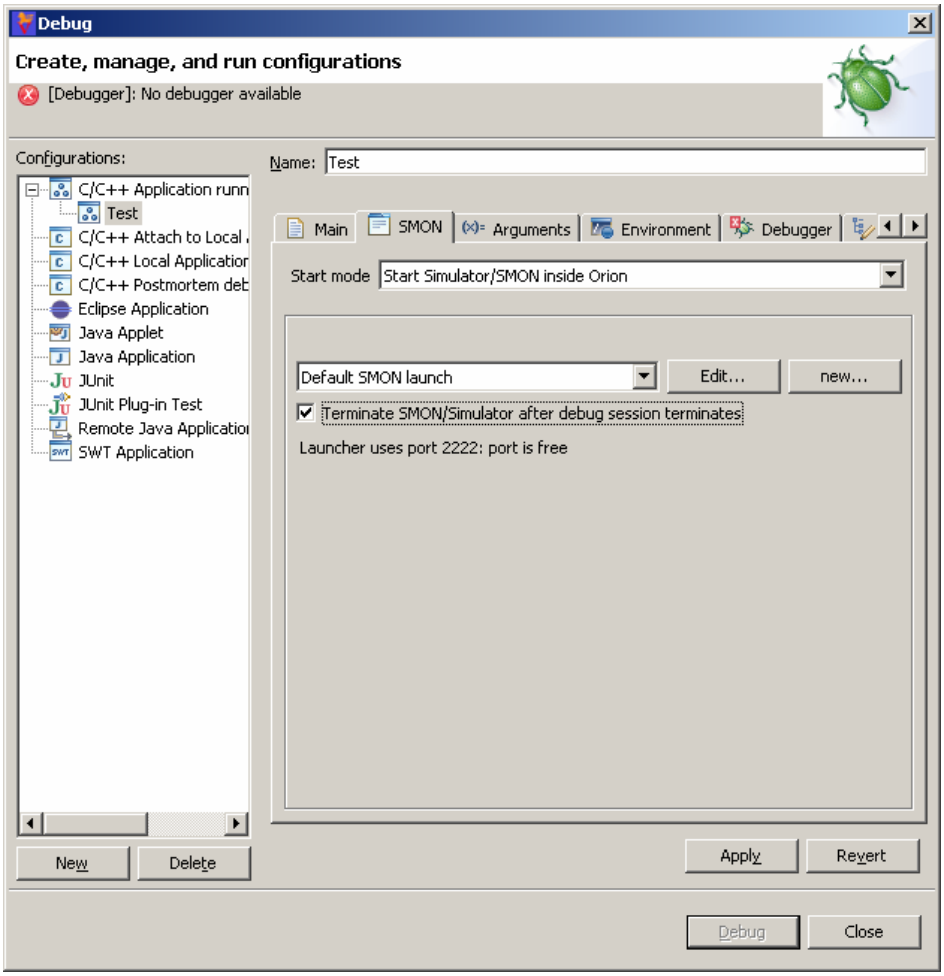


图 9-3 SMON 设置页

(5) Debugger 设置页：Debugger 选择框中选 **sparc-rtems GDB Debugger**，最下方的 SMON/Simulator port number 输入框中输入 **2222**；

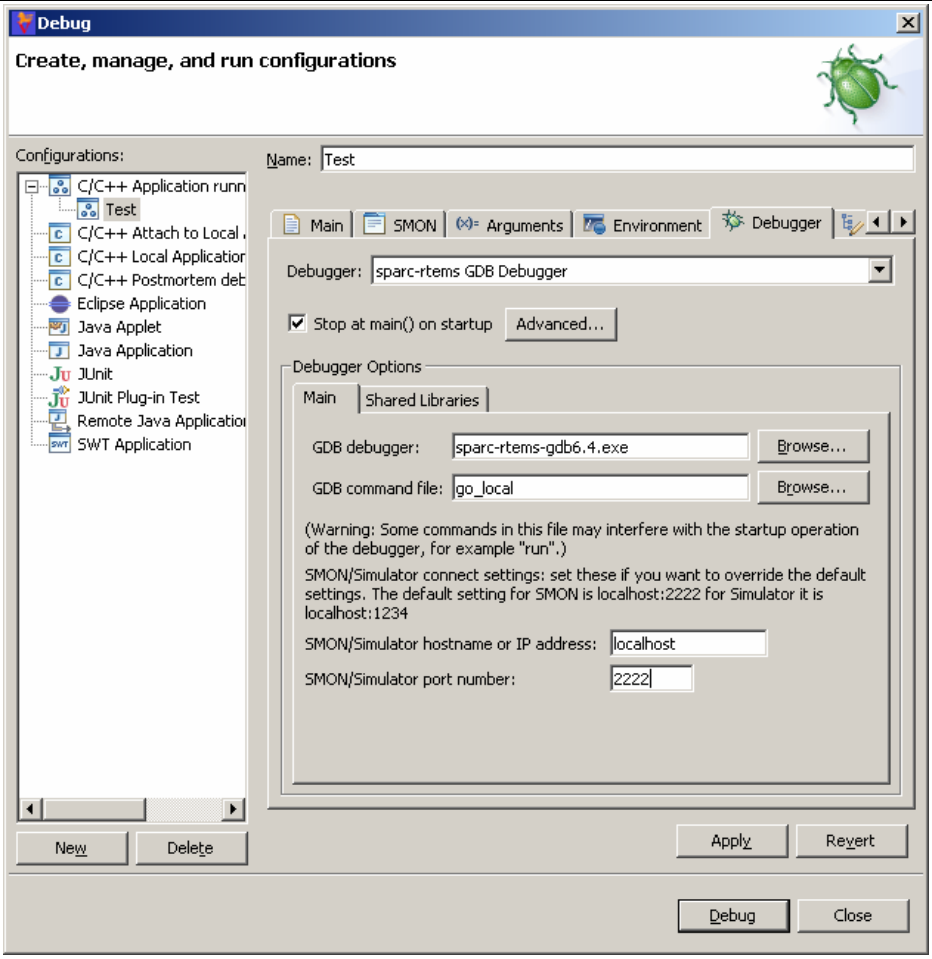


图 9-4 Debugger 设置页

(6) 点击 Apply 按钮结束设置。

9.5 在线调试

在调试之前必须首先保证：

(1) 目标板已经和 PC 机正确连接（**目标板的 DSU 口连接 PC 机的 COM1**）；

(2) 目标板已经上电启动；

(3) PC 机的 COM1 没有被占用（可能占用串口的程序有：V8mon、串口助手、超级终端等）。

以下是示例调试步骤：

(1) 启动调试：点击调试配置对话框中的 Debug 按钮启动调试；系统联机、程序下载需要一定的时间；

(2) 切换到 Debug 透视图：系统提示是否切换到 Debug 透视图，点击 Yes 按钮；

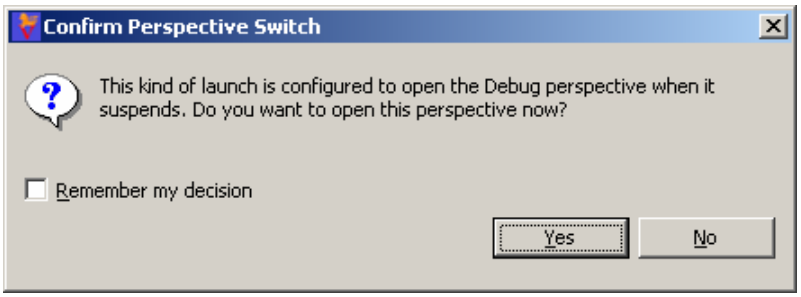


图 9-5 透视图切换提示框

(3) 确认调试启动：调试启动成功，程序暂停在 Main 函数的第一行语句，如图 9-6；

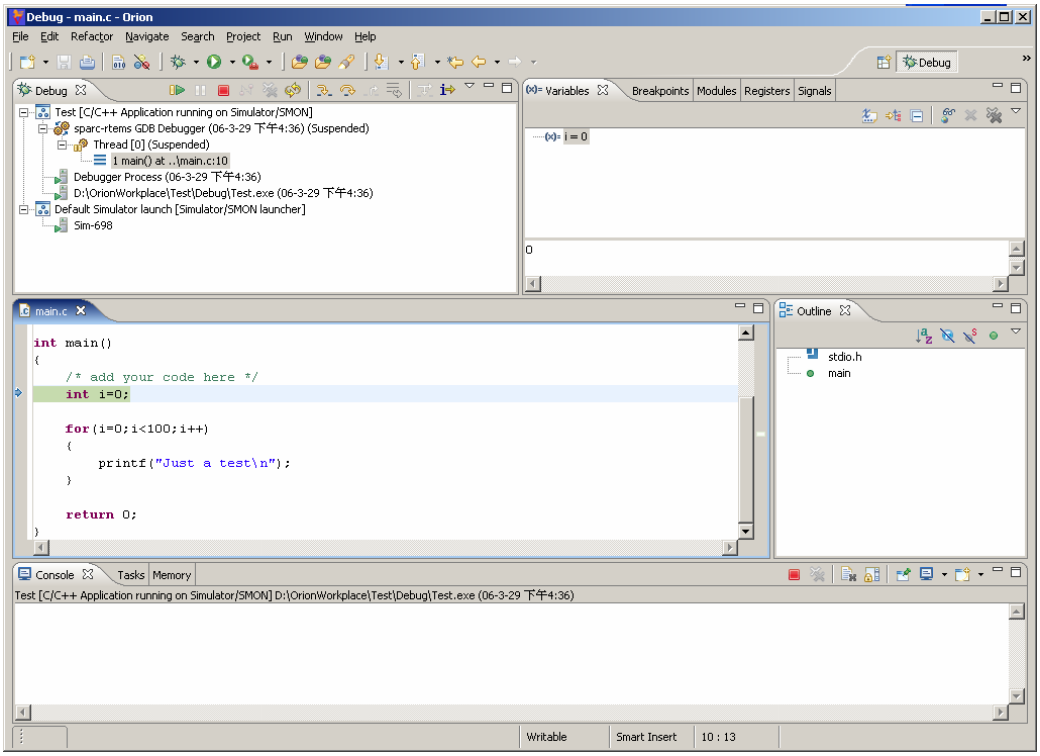


图 9-6 调试启动成功

(4) 单步运行：单击 Step Over 按钮  ，程序单步运行了一条语句；

(5) 观察变量：变量观察窗口自动锁定程序中正在运行的变量，多次重复单步操作，随着循环次数的增加，可以看到变量 i 的变化；

(6) 设置断点：我们在最后一行设置断点，先将编辑光标移动到目标位置，点主菜单 Run->Toggle Line Breakpoint；

(7) 运行到断点处：点击 Resume 按钮  ，程序将从当前位置运行到断点处；

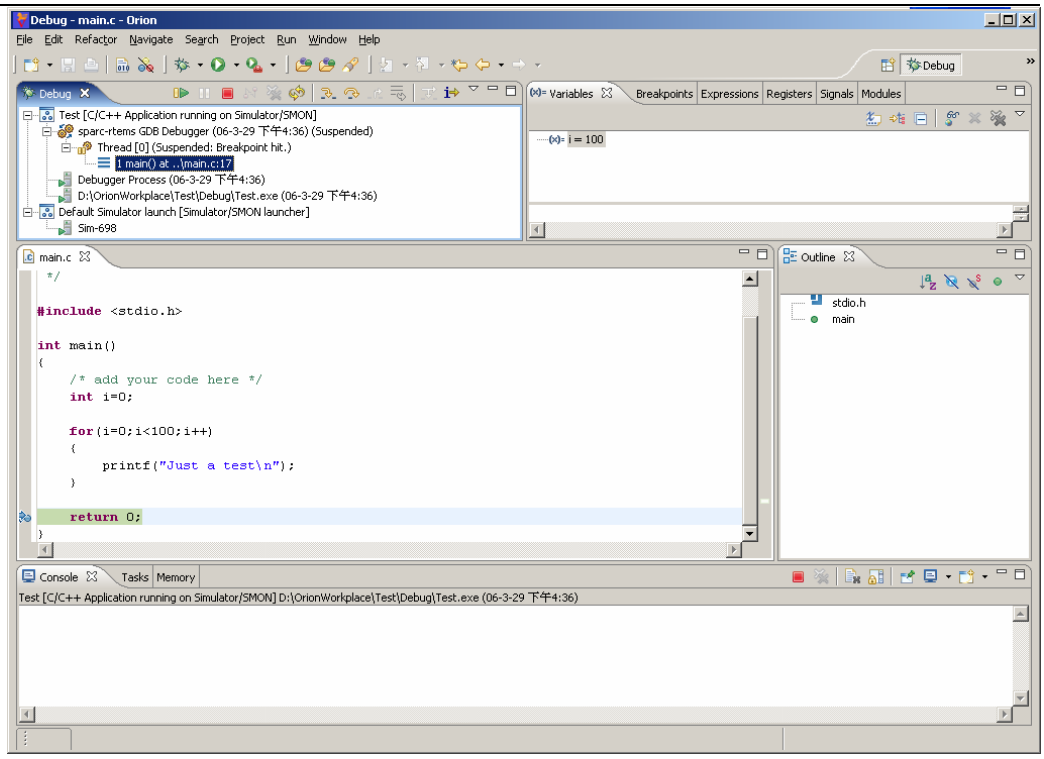


图 9-7 程序运行到断点处

(8) 运行完程序：继续点击 Resume 按钮  ；

(9) 观察运行结果：单击控制台右上角的 Display Selected Console 按钮  中右方的三角符选择第 3 项，可以在 Console 控制台中观察到程序运行过程中的所有打印输出值；

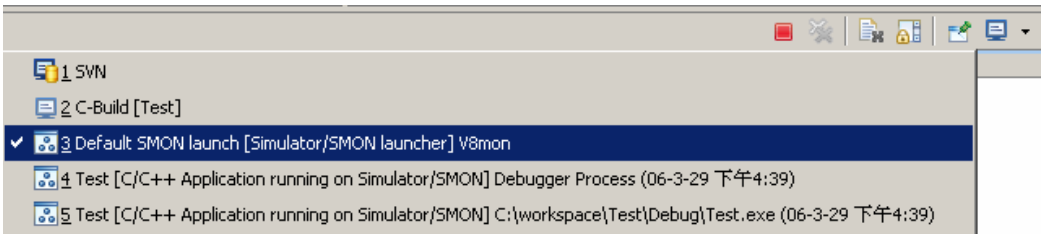


图 9-8 选择 V8mon 控制台观察程序输出

(10) 结束调试：点击 Remove All Terminated Launches 按钮  删除所有的调试进程；

(11) 重新切换到 C/C++透视图：点击主窗体右上角的 Open Perspective 按钮  ，选择 C/C++项，界面切换到 C/C++透视图，可以重新对程序进行编辑编译。

9.6 下载运行

程序调试无误后，可以通过命令行的方式直接运行程序。与连机调试相同，首先要

检查目标板是否与 PC 机正确连接，目标板是否已上电启动以及 PC 机 COM1 是否被占用。

命令操作步骤如下：

(1) 打开 Cygwin 命令行：点击桌面的快捷方式 cygwin.bat；

(2) 切换到可执行文件所在目录：

```
$cd d:
$cd OrionWorkplace/Test/Debug/
```

(3) 启动 V8mon 连接目标板：

```
$V8mon –i -u
```

如果连接成功就会出现如图 9-9 的提示信息，否则表示连接失败；

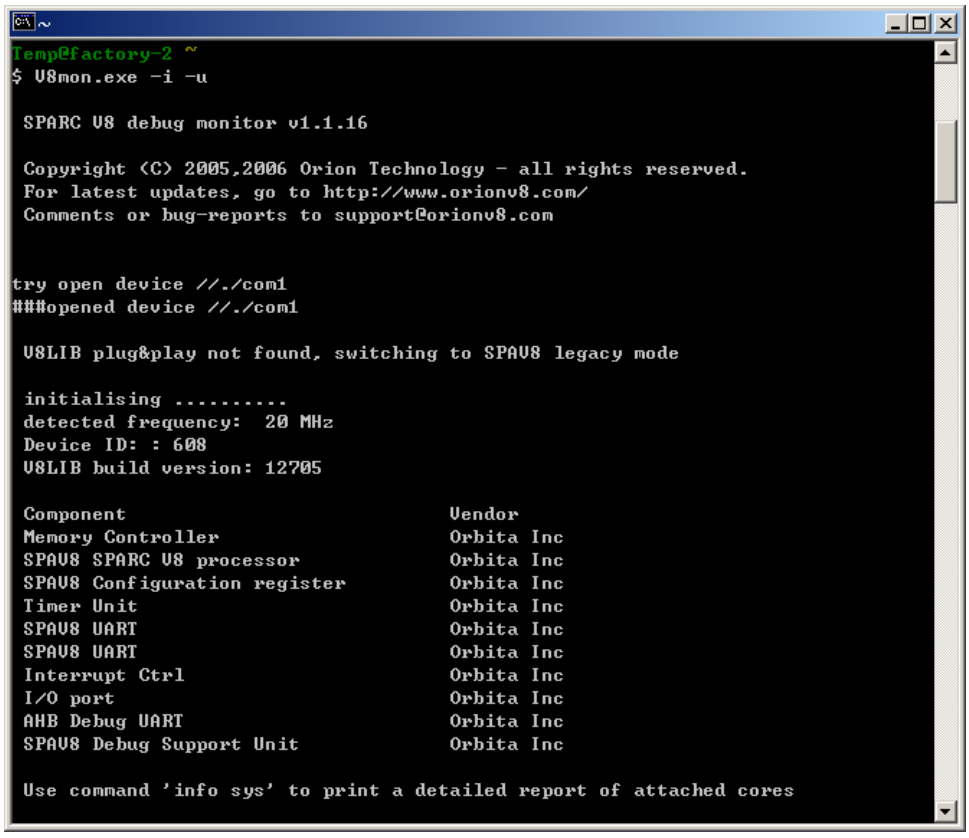


图 9-9 V8mon 提示信息

(4) 下载程序：

```
v8mon>lo test.exe
```

(5) 运行程序：

```
v8mon>run
```

在图 9-10 中我们可以看到循环打印出来的字符串“Just a test”；

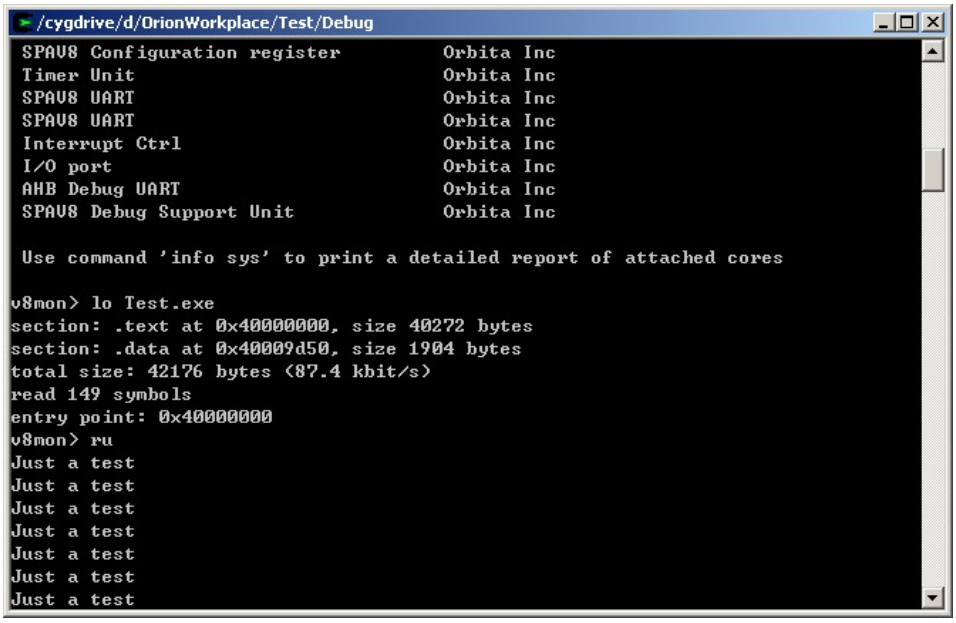


图 9-10 程序运行

(6) V8mon 退出：使用快捷键 Ctrl+c 可终止程序的运行，再次使用快捷键 Ctrl+c 则退出 V8mon，同时释放 COM1。

10 技术服务

公司：珠海欧比特控制工程股份有限公司

地址：广东省珠海市港湾大道白沙路 1 号欧比特科技园

电话：0756-3391979

传真：0756-3391980

邮编：519080

网址：www.myorbita.net

附录 I 常见问题及解决办法

1、 Orion4.0 安装后所有程序都无法编译

原因：Orion 在编译程序时需要调用 Cygwin 下的 make 命令文件；但如果电脑中事先已经安装有自带 make 文件的开发软件（如 Tornado、C++builder 等），由于环境变量的设置问题将导致 Orion 找不到自己的 make，编译时可能调用了其他的 make；

办法：首先查找硬盘中是否有多个 make.exe 文件，如果有则可以在环境变量设置中暂时将包含有其它 make 文件的路径删除。

2、 初次启动 SMON 调试时报错，无法启动

原因：在连接参数都设置正确的情况下，可能是目标板连接不正确；

办法：确定连接，可以先在 Cygwin 中使用 V8mon 命令测试连接(V8mon 使用完后一定要退出，不然它将一直占用串口)。

3、 再次启动调试时报错，无法启动

原因： （1）上次调试后没有做删除调试操作，可能 SMON/Simulator 没有关闭；

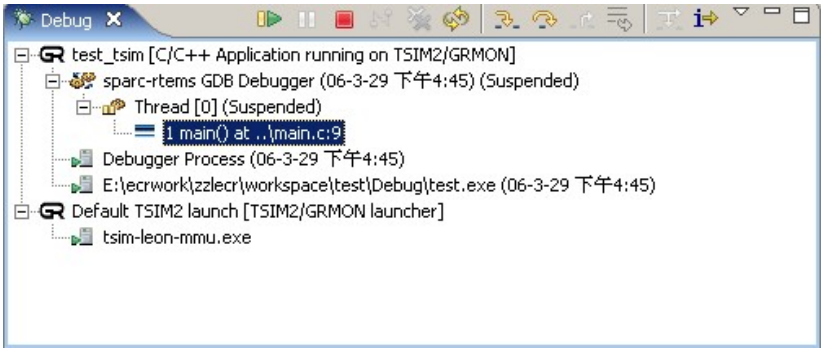
（2）上次调试程序产生异常，导致目标板 CPU 处于非正常状态；

办法： （1）确认没有 V8mon 、SMON/Simulator 等服务程序处于连接状态；

（2）目标板复位后重试。

4、 调试启动后代码窗左边没有出现指令停止箭头

办法：Debug 视图中层层打开代表调试线程的树型结构，点击标识有主函数名的线程。



附图 I -1 Debug 视图

5、 单步跟踪延时语句时等待时间过长

原因：类似” for(i=0; i<10000;i++);”这样的延时语句在单步跟踪时由于循环次数太多，会出现点单步运行后长时间没有响应的情况；

办法：尽量不要单步跟踪这种语句，可以将它封装成延时函数，或者在调试时使用 Resume 直接跳过这种语句。

6、 模拟器调试过程中出现调试错误

原因：程序中有些语句是直接对硬件的操作，这类语句在模拟器中运行时会产生错误；

办法：调试硬件驱动类程序时请用 SMON 调试模式。

7、 Properties 窗体中无法设置编译参数

办法：在左边的选择框中先点选 C/C++ Build 外的其它项，再点中 C/C++ Build 项进行设置。