# 针对S698系列处理器的Windows平台集成开发环境

## Orion4.0的设计和应用

珠海欧比特控制工程股份有限公司　龚永红　王超　李付海　颜军

目前S698系列处理器在电子、通信以及航空航天等领域有着广泛的应用，但还没有基于这种架构的Windows平台集成开发环境，Orion4.0正好填补了这一空白。本文主要介绍如何利用可扩展的Java开发平台插件机制实现Orion4.0，同时演示如何利用这个集成开发环境进行高效率的开发。

### 传统的开发流程

文本编辑器编写程序→选择编译工具编译→选择调试工具调试(Debug)→再编译→再调试……编译通过→连接→运行。如果要烧写程序，还需要选用额外的烧写工具。

这种开发流程的缺点是程序非常复杂，而且调试困难。它的操作都是通过命令行完成的，让人有种难以接近的感觉。举个简单的例子：当执行完编译操作后，控制台只是麻木地把编译信息打出，告诉你第几行出了错误，你要返回去逐行检查，找到出现错误的行数，然后再调试；或者你发现错误的原因只是把一个关键字给拼写错了……这样无形中延长了开发的时间，也影响了开发人员的情绪。

### 新的开发流程

用工程管理器选择开发的工程类型→源码编辑器编写代码(自动编译)→调试→运行→烧写。这就是使用Orion4.0进行嵌入式开发的开发流程，它为针对S698系列处理器的嵌入式软件开发提供了一整套的解决方案。

Orion4.0的操作都是图形化的，不需要使用命令行，对新手来说，非常容易上手。

流程的简化，得益于以下这些功能部件的有机结合。

工程管理器：图形化的工程管理工具，负责应用源程序的文件组织和管理，自动帮你选择好需要的编译、连接工具。

源码编辑器：标准的文本编辑功能，支持语法关键字、关键字色彩显示等（这样就会提醒你关键字的录入是否有错误），还可以显示行数、支持函数和编译的搜索，这对大型开发项目来说非常重要。

编译工具：专门针对S698系列处理器的sparc-rtems-gcc编译器和GNU的GCC编译器，并经过优化和严格测试。

调试器：源码级调试，提供了图形和命令行两种调试方式，可以进行断点设置、单步执行、异常处理，也可以查看修改内存、寄存器、变量等，还可以查看堆栈和进行反汇编等。在跟踪调试时可以任意切换C语言级调试或汇编级调试。这些功能已经完全可以满足一般使用者的要求，但Orion4.0并不仅限于此，它还提供了强大的软件模拟器让你在没有硬件平台的情况下也能进行软件调试开发。

### 简单为美的设计理念

软件学向来推崇以简单为美，因为复杂的东西是难以操纵的，如何简化开发的流程是Orion设计之初最先考虑的问题。

下面用Orion4.0的工程向导（Wizard）部分举例，说明流程是如何被简化的：在新建工程时，用户需要选择工程类型，如图1所示，不同的工程类型对应不同的工具链，如图2所示，工

具链会指定在工程运行的时候所需要的编译器、连接器、调试器等工具，以及这些工具的默认最优参数。也就是说，用户在开发的整个过程中对工具的选择和配置只需要操作一次，这些信息会一直保存，直到工程被删除。

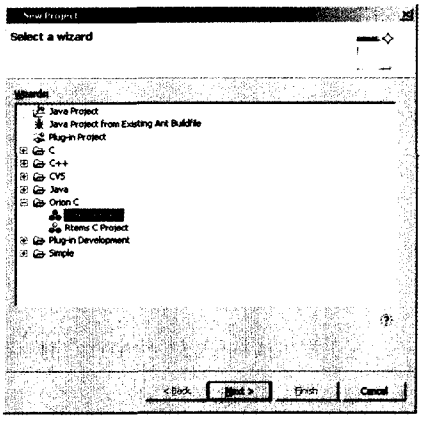


**图1 工程管理器**

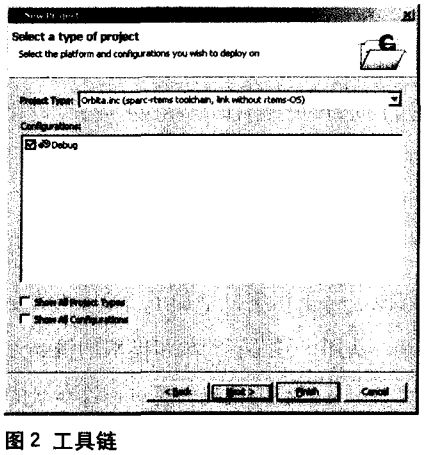


**图2 工具链**

为了方便用户的使用，Orion4.0还设计了很多辅助功能，比如：图形化界面设置RTEMS参数，图形化烧写参数设置，一键烧写功能等。

当然，Orion4.0还提供了详细的帮助文档。

**1 解决方案的设计模型**

用户操作如何传递给编译器？编译完成后错误和警告标志如何能正确无误的出现在对应的代码行上？这些都需要非常复杂的处理过程。但大致来讲，它们都是按照图3所示的设计模型完成的。

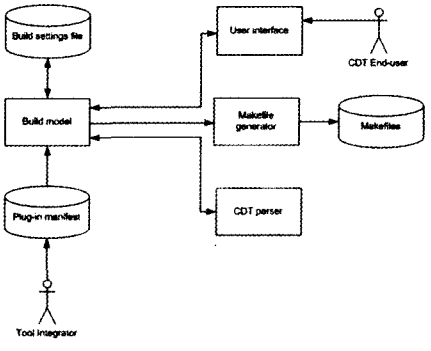


**图3 工具链配置管理模型图(Build Model schema)**

根据图3所示模型图，可以推断出使用工具链(ToolChain)管理的设计思想。以下是对模型图的简要说明：

①UI是用户接口，这部分是交给用户自己来控制影响Build model，最简单的例子就是在同一个工程里面，可以通过图形界面指定哪些文件编译，哪些文件不编译；

②Makefile generator是Makefile生成器；

③CDT Parser是二进制解析器，这是Windows下用GNU工具编译程序所必须的；

④Tool Integrator是GNU工具链集成器和其他东西，比如编译器sparc-rtems-gcc。

设计工作主要是依照这样一个模型来完成，模型中有要处理的工程的配置管理信息，然后Makefile generator采集这些信息，让它的Project工程中每个子目录生成对应的Makefile，模型的配置是可以变化和影响的，而默认的配置是由plugin.xml文件指定的。

**2 开发应用实例**

**①具体的应用例子**

现在简单列举一个编程例子来演示此开发平台的功能效果。按照一般教程的惯例，我们选择列举Hello World这个例子，向世界问好。

如果不想为一个入门的例子特意去找一个硬件平台，可以使用软件模拟器来模拟S698系列处理器的硬件平台；如果程序涉及外设硬件操作，就只能连接硬件平台并使用硬件调试器了。

首先，打开开发环境并选定一个工作区，参照图1，在Orion C下选择Bare C Project，新建一个test工程，然后单击“NEXT”按钮。

Bare C Project选项生成的工程，就是纯粹的在SPARC架构平台上运行的C工程；而Rtems C Project选项生成的工程，就是带RTEMS操作系统运行的C工程。它们的本质区别就是工具链的配置不同。而用户可以通过自己的配置来影响编译配置，这也印证了图3工具链配置管理模型图所表现的UI功能。

在Project Type中已经默认选择了工具链Orbta.inc(sparc-rtems toolchain, link whitout rtems-OS)，继续单击“NEXT”按钮，然后单击Finish按钮，建立工程。该工程里有自动生成的代码，用户可以修改编辑。就这样，用户没有写一行代码就生成了想要的程序。

现在可以看到的是编辑器支持语法关键字的色彩显示。其实编写程序代码的过程中，编辑器具体的显示色彩可由

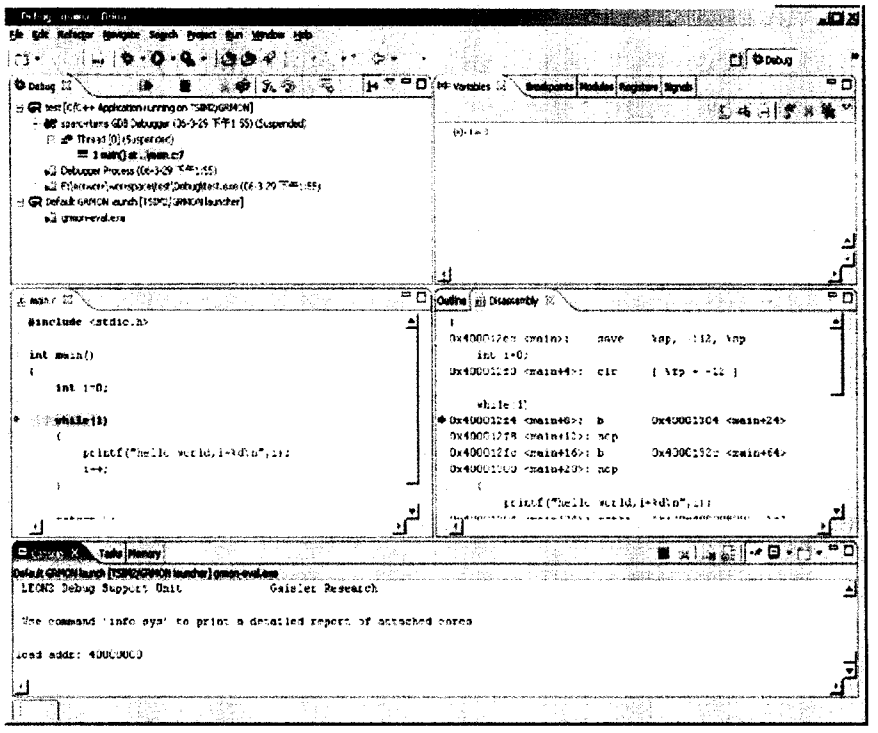


图 4 调试环境设置

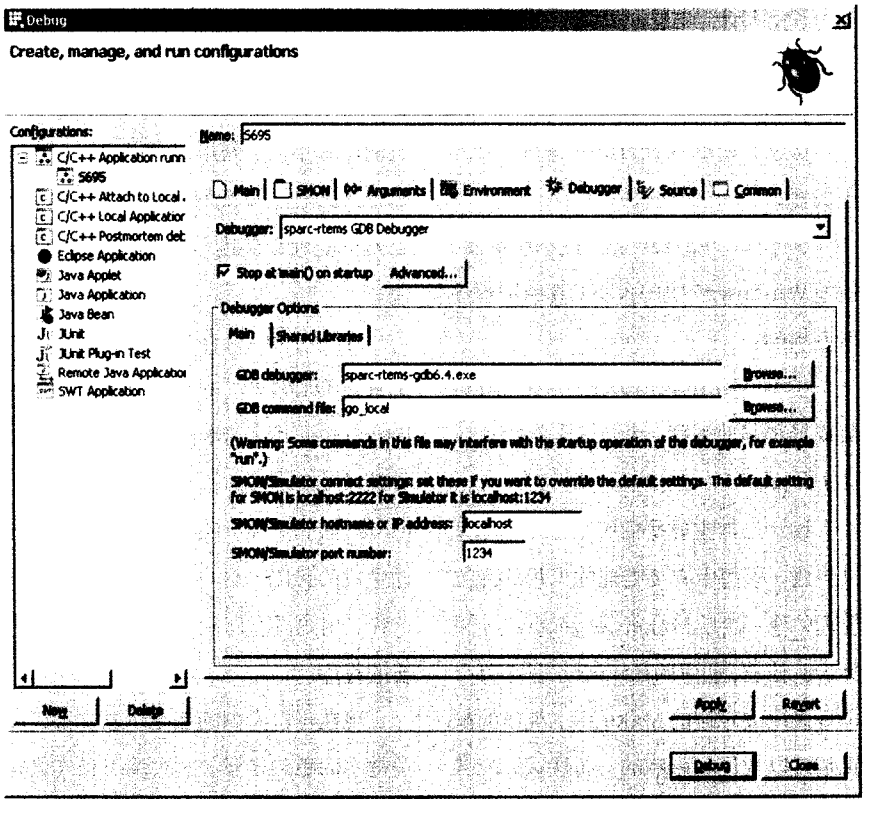


图 5 debug 界面

用户自由编辑，同时具有程序语法的自动纠错功能。这样既方便了代码编写，又提高了代码的编写质量。

如果程序没有出现预期的结果，就需要进行调试。Orion4.0强大的调试功能允许用户对程序进行单步跟踪，设置断点，观察变量，察看堆栈等。

Orion4.0 支持以下的调试方式：

- Simulator 调试模式；
- SMON 调试模式；
- Debug monitor 调试模式。

**② 调试**

现在大概了解一下调试的方式。

在工程生成的EXE文件中点右键，选择“Debug As”，选择“Debug”，在弹出窗口中的“C/C++ Application running on Simulator/SMON”项中右键选择New（或双击）以新建调试环境，然后根据不同的调试模式做好相应的设置，设置完成后直接在“Debug”设置窗口单击“Debug”按扭启动调试，如图 4 所示。

因为要展现工具的功能，我们稍微修改了一下程序，增加了一个 int 的变量。从图 5 可以看到单步调试的浅绿色亮条，反汇编的代码，还有追踪变量的窗口。当然这些只是Orion4.0的一小部分的功能。

## 结束语

作为术开发人员，总是希望自己手中的工具既方便又强大，从而摆脱琐碎的操作环节，专注于编程，而Orion4.0就能够实现这个想法。另外，该开发平台是可以扩展的，你可以在这个开发平台上使用任何一种编程语言。 EPC