VisualDSP++中文手册

目录：

（一）开发工具及其特点

1、开发工具概述

Visual DSP++是 ADI 公司针对 ADI 公司的 DSP 器件而专门开发的一种使用方便的开发平台，它支持 ADI 公司所有系列的 DSP 处理器，包括 Blackfin 系列和 ADSP-21XX 系列定点处理器、SHARC 系列和 TigerSHARC 系列的浮点处理器的各种型号处理器。

Visual DSP++通过图像窗口的方式与用户进行信息交换。VisualDSP++采用直观的、易于使用的用户界面，针对处理器进行操作。ViSualDSP++集成了两大部分：集成的开发环境(Integrated Development Environment，IDE)和调试器(Debugger)，称为IDDE(IntegratedDevelopment and Debugging Environment)，提供了更强大的程序开发和调试功能。VisualDSP++具有灵活的管理体系，为处理器应用程序和项目的开发提供了一整套工具。VisualDSP++包含生成和管理处理器项目必须的所有工具。

Visual DSP++从推出至今已经经历了 1.0、2.0、3.0、3.5、4.0、4.5 及 5.0 七种版本，相应的 DSP 开发和调试功能也不断增强。下面以常用的 VisualDSP++的 4.5 版本进行介绍。

Visual DSP++开发工具包中集成了开发 DSP 程序所需要的各种工具组件，根据用户所购买的软件，VisualDSP++包含下列组件中的一个或多个组件。

- 与 Visual DSP++一体化的集成开发和调试环境（IDDE）
- 带有实时运行库的 C/C++语言最优化编译器
- 汇编程序、连接器、预处理器和档案库
- 程序加载器、分割器
- 模拟器
- EZ—KIT Lite 评估系统（必须单独购买）
- 仿真器（必须单独购买，推荐安诺电子的 AN 系列 ADI DSP 仿真器 http://www.analogcn.com/Shop/shop1/Index.html）
- 程序实例

一下是 Visual DSP++的基本特点。

（1）源文件编辑特点

Visual DSP++简化了源文件的操作任务，可以非常容易地实现创建^查看、打印、移动和信息定位等相关文件操作。

- 编辑文本文件。创建和修改源文件，查看由代码开发工具生成的文件。
- 源文件按是 DSP 工程开发的重要组成部分，可以采用 C / C++语言或汇编语言进行编写。如果 DSP 开发工程的源代码文件采用汇编程序进行编写，那么 DSP 开发工程中还应当包含链接描述文件(. LDF 文件)和一些相关的数据文件，而如果 DSP 开发工程的源代码文件采用 C / C++语言进行编写，那么相应的工程则可不必包含链接描述文件。
- 编辑窗口。Visual DSP++编辑器是一个完整的代码书写工具，用于编辑文本文件。查看和编辑多个编辑窗口的相关文件，也可为一个文件打开多个编辑窗口。
- 与上下文相关的表达式评价。将鼠标指示移至一个变量上 j 在一定范围内可以查看变量的值。
- 状态图标。用于指明断点，书签和处理器当前执行程序指令的在源代码文件中的位置。
- 查看错误信息和违规代码。通过突出显示错误代码(如 cc0251 等)或按 F1 键，可以在输出窗口的生成视图中看到错误的详细信息，而且双击错误行可跳至编辑窗口中的违规代码。

（2）工程管理特点

Visual DSP++为处理器应用程序的开发提供了灵活的工程管理，包括创建、定义和编译等处理器项目所必须的操作。

- 定义和管理工程。管理用户工程编译时所需的相关的文件和相关的开发工具。对工程的定义只需一次即可，在开发的过程中用户可以根据需要对工程灵活地进行修改。
- 查看和管理代码开发工具。配置选项中确定了代码开发工具如何处理输人文件和生成输出文件。对于代码开发工具，工具设置类似于命令行转换。工程的配置选

项可以在工程定义的时候进行设置，也可以在工程开发的过程中进行修改。

- 查看工程编译结果。在进行工程编译的过程中可以随时查看编译状态，并且根据用户需要，用户可以随时停止工程的编译过程。查看工程编译结果时，如果工程编译存在错误，那么用户在输出窗口中双击错误信息则可以查看造成错误的源代码，或者重复错误信息。
- 管理源代码文件。根据工程窗口内管理项目中的源文件和文件的依赖项，可以依次显示文件之间的关系。VisualDSP++使用代码开发工具处理工程和生成处理器所需的程序。它也提供了源代码控制(SCC)界面，使得用户可以直接在 IDDE 环境下完成源代码的控制操作。

（3）调试特点

在调试一些工程的过程中，Visual DSP++提供了以下一些工具和功能为用户服务：

- 产看 C/C++语言和汇编语言的联合编程的源代码文件。在汇编源代码中，行数和符号信息有利于用户在源文件上查看和调试汇编代码。
- 运行命令行脚本。通过使用脚本，用户可以使用它制定调试过程中的主要参数和特性。
- 使用存储器表达式。使用存储器相关的表达式。
- 利用断点查看寄存器和存储器。可以快速添加和移除断点，使能和使断点失灵。
- 设置模拟观察点。对堆栈、寄存器、存储器和图标设置观察点可以停止程序的执行，方便调试过程中观察相关信息。
- 统计描述目标处理器的指令执行数(该功能仅用于 JTAG 的仿真调试目标)。用户可以随意设置统计过程的取样，并将统计结果采用图形显示，根据统计结果可以轻松地观察到程序中最耗时的指令部分。
- 线性描述目标处理器的指令执行数(该功能仅适用 Visual DSP++的模拟调试目标)。对每个 DSP 的 PC 寄存器进行取样，统计它们的执行情况，并将结果采用图形显示。该功能与统计描述目标处理器的指令执行数功能类似，只不过该功能只能在 Visual DSP++模拟下使用，而统计描述目标处理器的指令执行数功能在 JTAG 仿真器方式下使用。

- 模拟 I / O 端口数据流、中断产生。通过该工具可以模拟串口或存储映射 I / O 进行数据流传输和模拟处理器通用中断的产生。
- 创建用户自定义的寄存器窗口。配置一个自定义的寄存器窗口来显示指定的寄存器组。
- 根据处理器存储器中的数值进行绘图。该工具将处理器存储器中的数据以图像的形式进行显示，并且用户可以根据自己的需要选择多样的绘图风格、典型数据处理功能和外观显示方式。
- 跟踪程序运行历史，跟踪用户程序，可以获得用户程序是如何达到用户设置的特定的程序点，并显示读、写和符号名称等相关信息。
- 查看汇编指令的流水线深度。通过流水线界面可以查看目标处理器流水线的阶段。

（4）VDK 特点

Visual DSP++核（VDK）是一种可扩展的软件执行程序，专门用于高效开发 ADI 公司 DSP 处理器的操作。Visual DSP++软件集成了 Visual DSP++核。

VDK 能够方便用户从软件中获取硬件实现的详细信息，使得用户可以更加专心的完成处理算法的实现。

VDK 为所有处理器应用开发过程阶段提供了基本模块，它们的性能描述如下：

- 自动化。VisualDSP++可以根据用户指定的语言自动生成源代码框架。
- 确定性。VisualDSP++明确指明 VDK 应用程序接口执行时间的确定性。
- 多任务处理。VDK 的任务问(线程)是相互独立的，并且每个线程都有自己的栈。
- 模块化。VDK 包含各种组件，并且在以后的版本中将会提供更多的功能。
- 方便移植性。大部分的核组件可以用 ANSI 标准的 C 或 C++语言编写，这将方便地实现代码在不同处理器之间的移植。
- 优先性。VDK 的优先级的调度表可以使高优先级的线程无需等待信号运行，随时可以执行。
- 原型化。VDK 和 VisualDSP++包含模板文件，可以方便用户创建原始文件，且整个应用程序是原型化的，需要用户根据需要进行测试和修改。
- 可靠性。VDK 提供实时运行过程中的错误检查。

- 可扩展性。如果某个项目不包括该属性，那么目标系统中将不包含其相关的代码支持。

（二）DSP 程序开发方法

利用 Visual DSP++集成环境开发流程如图 5.1 所示。

DSP 程序开发有三个阶段：

- Simulation—利用 Visual DSP 提供的软件环境进行软模拟，不需要硬件；
- Evaluation—利用 EZ-KIT 板对程序进行测试和评估；
- Emulation—利用 JTAG 口对用户的目标系统进行仿真调试。

（1）模拟（simulation）阶段

工程师开发新硬件时，项目开发通常以模拟环境为开始，模拟系统存储器和 I / O，允许查看部分目标系统的硬件行为。模拟器是一种软件，用于模拟处理器的操作。由 VisualDSP++可以生成一个模拟目标(无物理处理器)运行、编辑和调试用户的程序。

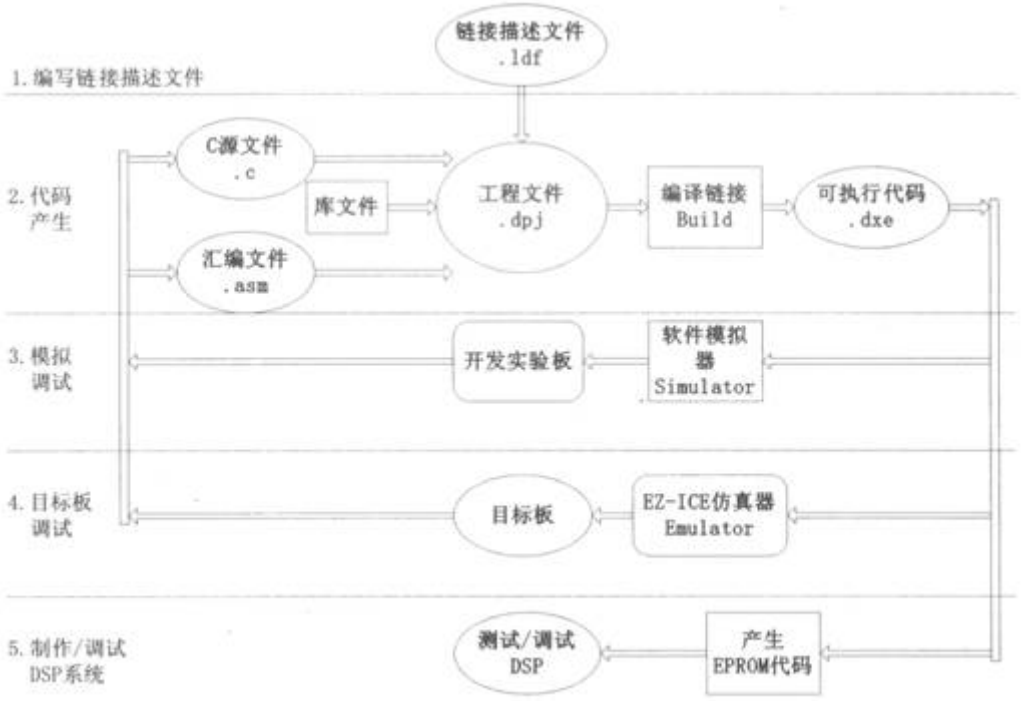

图 5-1　VisualDSP 程序开发流程

（2）评估（Evaluation）阶段

在项目设计的初期，使用 ADI 公司的 EZ．KIT Lite 评估系统确定处理器和验证用户设计的程序的部分功能，并进行评估。

（3）仿真（Emulation）阶段

用户目标系统的硬件设备准备完成后，用户可以通过 JTAG 仿真器将 PC 与用户的处理器目标板进行链接。仿真器为 PC 与实际处理器目标板之间提供了快捷通信，在 VisualDSP++环境下通过仿真器可以将用户的程序下载到处理器内部，然后让程序在用户目标系统的处理器上运行。采用仿真器进行调试，处理器实际上是工作在用户的目标系统中，PC 和仿真只是起到控制和监视作用，因此通过仿真器在用户目标系统上调试的程序基本上是符合用户目标系统在实际工作中的程序的。

在完成仿真阶段的程序设计和调试后，DSP 程序的开发基本上完成了，剩下的工作，用户只需要将开发的程序生成加载文件提供给用户的目标系统，用户的目标系统按照设计的加载方式对处理器进行加载，那么处理器就能够按照用户设计的程序运行了。

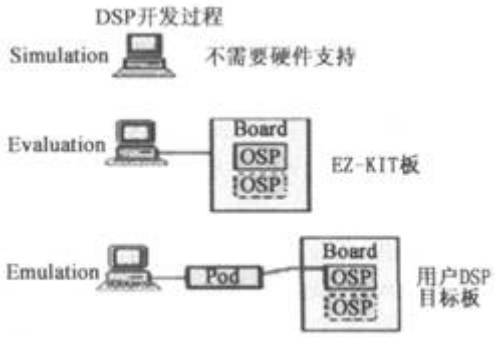

图 5-2　DSP 程序开发的过程示意图

DSP 程序开发的过程示意图如图 5-2 所示

在过程开发过程中，Vishal DSP++集成开发和调试环境中可利用的调试工具见表 5-1。

表 5-1　模拟、仿真、评估过程中的使用工具

调试工具	Simulation	Evaluation	Emulation
线性剖析(Linear profiles)	√		
中断模拟(Interrupts)	√		
数据流模拟(Streams)	√		
跟踪(Traces)①	√		
流水线查看(Pipeline Viewer)②	√		
Cache 查看工具(Cache Viewer)	√		
断点(Breakpoints)	√	√	√
检测点(Watchpoints)	√		
硬件条件中断(Hardware breakpoints)			√
绘图(Plotting)		√	√
统计剖析(Statistical profiles)			√

注：①该功能只针对 SHARC 系列处理器有效
②该功能不支持 SHARC 系列处理器

（三）利用集成开发和调试环境 IDDE 进行 DSP 程序开发

Visual DSP++的集成开发和调试程序界面主要由工程管理窗口、文本编辑窗口、反汇编窗口、输出窗口和一些辅助菜单组成，如图 5-3 所示。

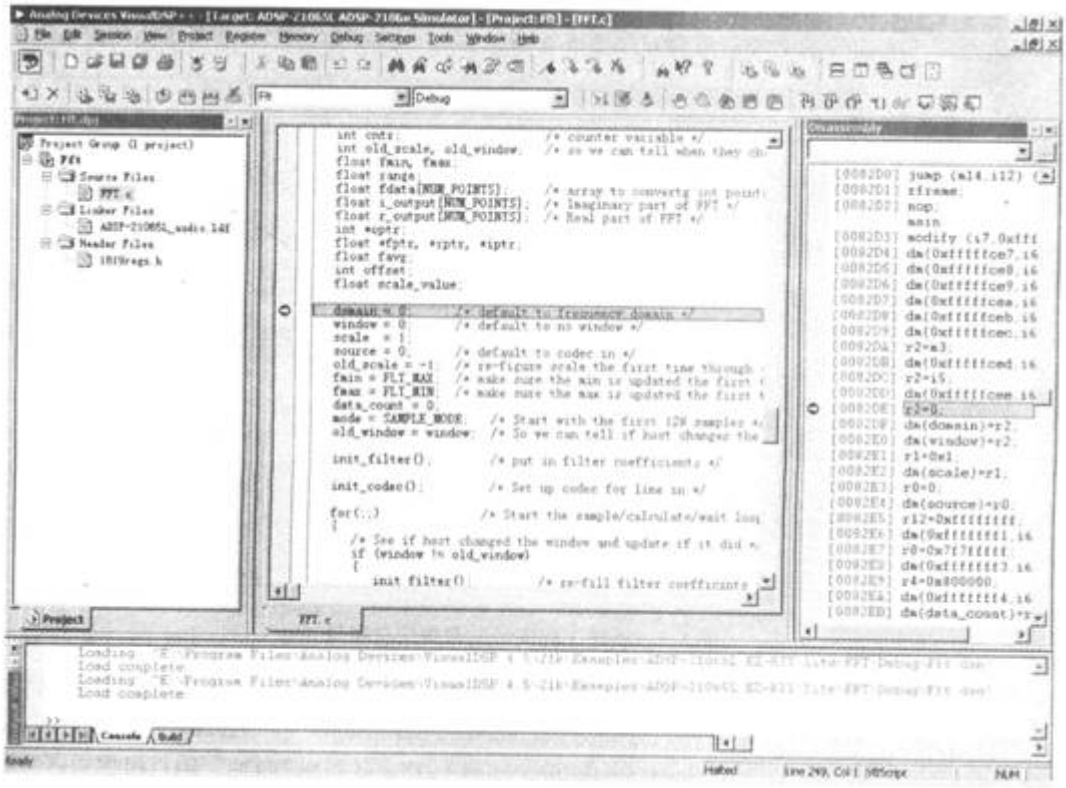

图 5-3　Visual DSP ++ 的集成开发和调试程序界面

集成开发和调试环境支持对 DSP 应用程序开发的整个过程，在 IDDE 的应用程序的开发一般都要经过如下几步：

第一步，创建一个新的工程。

第二步，设置工程选项。

第三步，编辑或添加工程源文件。

第四步，设置工程编译链接选项。

第五步，编译链接 Debug 版的工程，生成可执行文件。

第六步，建立 Debug Session 和加载可执行文件。

第七步，运行和调试（Debug）程序。

第八步，编译链接加载（Release）版本的工程。

通过以上这 8 步，就可以方便地完成整个 DSP 的应用开发，下面分别做介绍。

第一步 创建一个新的工程文件

在 Visual DSP++中，DSP 的所有应用开发都是基于工程的，所以创建一个工程文件是整个软件开发的第一步。工程文件(术．dpj)中存放程序的编译链接信息：源文件列表、其关

联关系信息和开发工具的选项设置等。

1）打开 IDDE：选中 WINDOWS’中的开始菜单“Start\Programs\Analog Devices\VisualDSP++ 4.5\VisualDSP++ Enviroment”，弹出 IDDE 主界面。

2）单击“File”下拉菜单中的“new”—>“Project”，Visual DSP++将启动新建工程向导，帮助用户逐步建立新工程。新建工程向导主要包含两个部分：“常规(General)”和“输出类型(Output Type)”。新建工程向导的常规信息窗口如图 5．4 所示

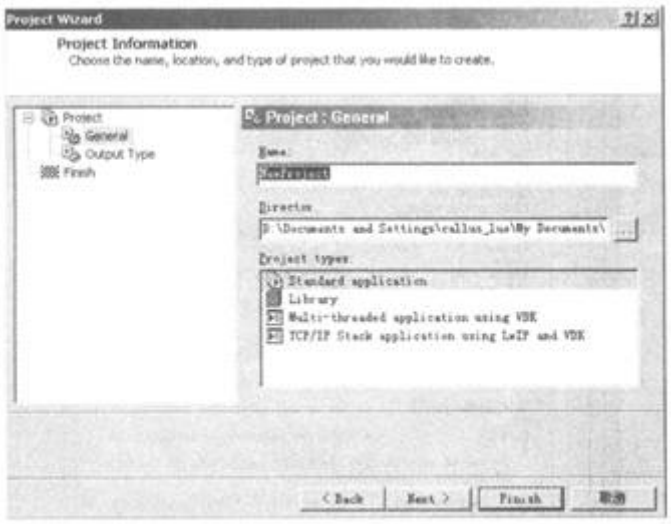

图 5-4　新建工程向导的常规信息窗口

在新建工程向导中主要包含了所建工程的名称、路径和工程类型等。工程的名称和路径由用户自己设置，工程的类型有四种，分别为：“Standard application”、“Library”、“Muhi—threaded applicaton using VDK”和“TCP／IP Stack application using LwIP and VDK”四种，用户根据自己需要进行选择，系统默认为“Standard application”，本书也将按照“Standard application”进行讲解。在将工程名称、路径和类型设置完成后，单击“NEXT”按钮，VisualDSP++将显示工程选项设置窗口，如图 5-5 所示。

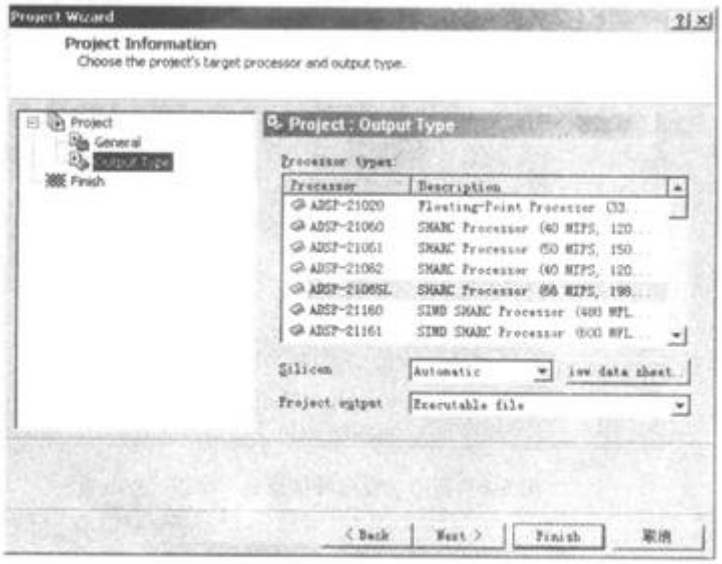

图 5-5　新建工程向导的输出类型窗口

在输出类型窗口中主要是设置工程采用的处理器类型(Processor types)、芯片版本号(Silicon)和工程输出文件类型(Project output)等。

处理器类型窗口用于可以选择相关的处理器，该窗口中包含了 ADI 公司 2007 年以前的所有处理器类型。

芯片版本号选项将随所选的处理器信号变化而变化，包含 2007 年所有处理器的芯片版本号，另外还附加了自动(Automatic)、无(NONE)和任意(any)三个选项，例如处理器 ADSP 201065L 有 0.1、0.2、0.3 三个版本，所以当处理器类型选择为 ADSP 21065L 时，相应的芯片版本号选项中则有 6 个选项：自动、无、0.1、0.2、0.3 和任意。用户需根据自己开发的处理器芯片版本号自行选择，在芯片版本号未知的情况下可以选择自动或者任意。

工程输出文件选项是用来设置工程输出的文件为处理器可执行的文件(Executable File)或者处理器加载文件(Load File)。如果用户的工程处在调试阶段，那么一般将该选项设置为处理器可执行的文件，以方便模拟器或者仿真器进行调试，而如果用户的工程处于开发完成阶段，那么一般将该选项设置成加载文件，加载文件可以用于对处理器进行程序加载。

在对新建工程向导的输出类型设置完成后，单击“NEXT”按钮则将显示出用户建立的工程信息，如图 5-6 所示。如果用户确认信息无误，则单击该窗口中的 “Finish”按钮将完成工程的建立，如果用户需要修改工程的某些参数，那么通过单击“Back”按钮可以回到前面的窗口重新进行设置。

另外，如果用户在建立工程完成后，通过工程选项窗口也可以对工程进行修改。

当新的工程建立完成后，在工程管理窗口中将显示出新的工程，并且在该工程下通常有三个文件夹分别为：源文件文件夹(Source Files)、链接文件夹(Linker Files)和头文件的文

件夹(Header Files)，它们分别用于存放工程中相关文件，如图 5-7 所示。

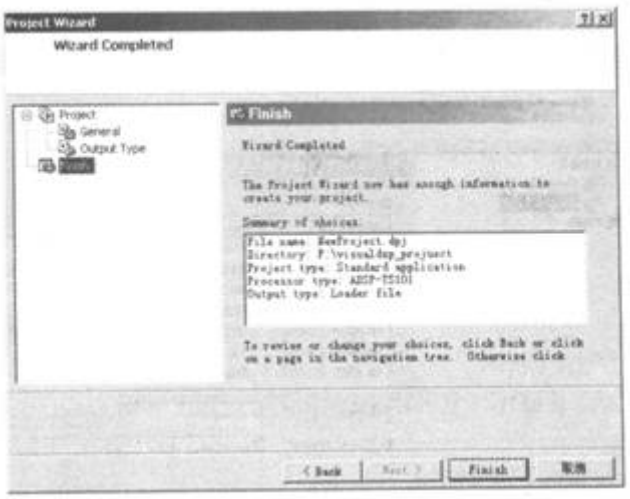

图 5-6　新建工程向导信息显示窗口

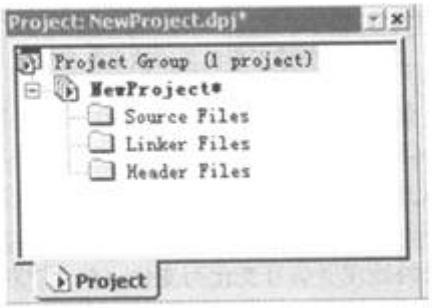

图 5-7　新建工程中的文件夹

第二步 设置工程选项

在新的工程建立完成后，用户可以通过工程选项窗口对工程修改，并对工程设置参数。用户通过单击 visualDSP++主界面中的“Project”下拉菜单，然后选择“Project Options”或者通过键盘快捷方式“Alt+F7’’可以显示出工程选项窗口，如图 5-8 所示。

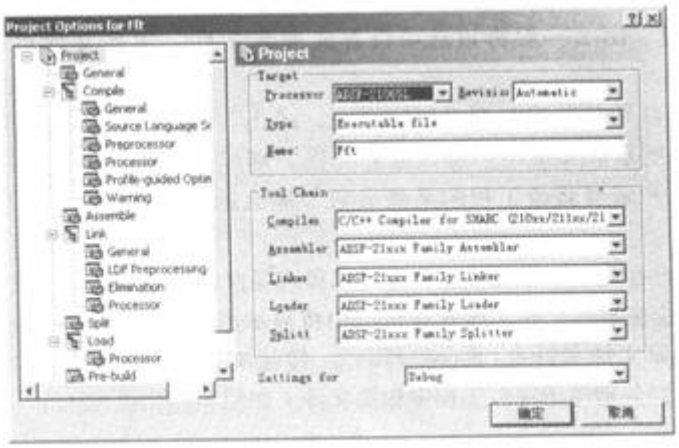

图 5-8　工程选项窗口

该窗口中主要包含 9 个部分，分别是：工程(Project)、常规(General)、编译(Compile)、汇编(Assemble)、链接(Link)、分割(Split)、加载(Load)、预编译(Pre. build)和后编译(Post-Bulid)等。

大部分选项栏的选项可以使用默认值，主要是 Project 选项栏的设置。它用于选择处理器类型和工程输出类型，其余选项可以使用默认值。Project 选项栏的各选项的意义如下。

目标（Target）

- Processor：用于设霉该工程中使用的处理器的类型，包含了 ADI 公司的绝大多数处理器(如 ADSP-21 160、ADSP-2106i、ADSP-21062、ADSP-21065L 等)。
- Type：IDDE 的输出文件类型，包括可执行文件(E 斌 utable File)、库文件(Library File)、加载文件(Load File)、目标文件(Object File)和分割文件(Split File)等。
- Name：输出文件的文件名，如 FFT。

工具链组（Tool Chain）

- Complier：指定 C 编译器；
- Assembler：指定汇编器；
- Linker：指定连接器；
- Loader：指定加载器；
- Splitter：指定加载方式的镜像文件管理器。

该组参数基本上使用默认即可。

设置（Setting for）

指定一个输出类型，有以下两种输出类型：

- Debug 类型，编译链接的工程文件可用来进行 Debug 调试。
- Release 类型，生成具有限制的或不能进行 Debug 调试的工程文件，Release 类型通常用来进行优化程序性能。

一般在调试过程中选择 Debug 类型，当程序调试好以后，选择 Release 类型。

第三步 编辑或添加工程源代码文件

一个工程文件--般包含--+或多个 C / C++或者汇编语言源代码文件。当创建了一个工程并在工程选项中指定了所用的处理器类型后，就可以编辑新的源代码文件或将已存在的源文件加入到该工程中。

（1）添加文件到工程中

Visual DSP++支持将多种类型的文件添加到工程中，当工程进行编译链接时，IDDE 能自动选择可识别的文件进行编译链接。

添加文件到工程中一般可以采用三种方法。

1）通过单击工具栏中的添加文件图标

2）选择工程下拉菜单中的“Add to project”->“File(s)…”;

3）在工程管理窗口中，选中所需添加文件的工程，然后单击鼠标右键，选择菜单中的“Add File（s） to Folder…”选项。

其他两种添加方式如图 5-9、图 5 一 l0 所示，无论采用那种方法进行文件添加，都将弹出文件选择窗口，如图 5—11 所示。

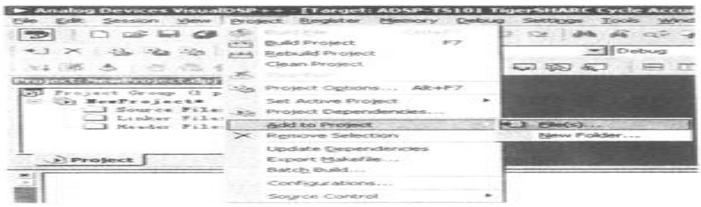

图 5-9　添加文件到工程方法 2

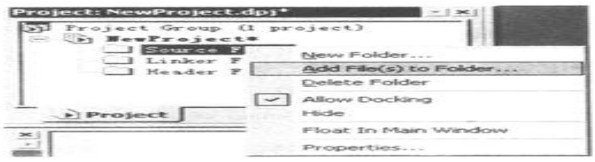

图 5-10　添加文件到工程方法 3

图 5-11　添加文件到工程窗口

在对话窗中可以查找所需的源文件，双击该文件后会自动添加到工程中。被添加的文件会自动出现在工程管理窗口的文件目录列表中，选择某个文件，然后单击鼠标右键就出现对该文件的操作栏。

（2）新建一个文本文件并把它加入到工程中

选 File＼New，或从工具栏中选择编辑新文件图标按钮，则会打开一个 WINDOWS 风格的编辑窗口，接下来就可以在里面进行编辑了。VisualDSP++的编辑器可以编辑任意名称的文本文件，VisualDSP++．文本编辑器将根据文件后缀名来判断文件类型并根据文件类型以不同的颜色显示源代码文件中的关键字。

当把一个文件添加到工程中后，它会自动更新工程窗口中的关系树。

（3）编辑文件

VisualDSP++的编辑功能是非常强大的，不但支持标准的编辑功能，还支持用户、DSP 指定语言语法的 Hightlighting(不同颜色显示)功能，还可以加入书签和进行列编辑操作等。

其他一些标准的编辑操作，如 copy、paste、cut 和书签等功能与其他的编辑器一样。注意编辑完成后，要把新编辑的文件存盘，并加入到工程中去。

（4）工程相关性

相关性(Dependency)用于描述工程中源文件之间的相互关系，它存放在后缀为．mak 的文件中，即哪一个文件需要用到另一个文件的信息，因此这决定了编译链接的顺序。

更新工程相关性可以通过 Project＼Update Dependencies 来实现。

第四步 设置工程配置选项

创建完工程、设置工程选项和添加源代码文件完成后，需要定义工程配置选项或按照默认的设置进行编译，之后才能生成处理器的可执行文件。通过下拉菜单 Project，选择 configurations 选项，将弹出 Projectconfigurations 窗口，如图 5-12 所示，在其中将指定编译链接其产生的工程的类型等。

工程类型决定工程编译链接后的类型，有两种选择：Debug 和 Release，默认的类型为 Debug。选择 Debug 类型，并且接受其他默认值时，编译器会产生一个包含有调试信息的目标文件，供调试使用。而选择 Release 类型，并接受其他默认值时，编译器则会产生一个不包含有调试信息的目标文件，并且会对代码进行优化。

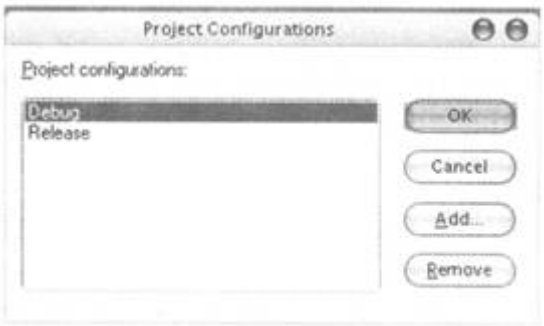

图 5-12 Projectconfigurations 窗口

在新建工程时，新建工程向导中也有对工程选项的设置。

第五步 编译链接 Debug 版的工程生成可执行文件

在对工程配置完成后，需要使用 Build 方式对工程和相关文件进行编译和链接。使用 Build 方式有多种方法。

1）通过使用工具栏上的图标或者键盘快捷方式“F7”来编译链接当前工程文件，如图

5-13 所示。

图 5-13　编译链接工程图标

2）通过 VisualDSP++的下拉菜单 Project.，单击菜单里面的 Build Project 来编译链接当前工程文件，如图 5．14 所示。

3）在工程管理窗口中，选择相应的工程，然后单击鼠标右键，在弹出的菜单中同样有“Bulid Project”相关选项，如图 5—15 所示，单击后完成对工程的编译链接。

图 5-14　工程菜单中的编辑链接

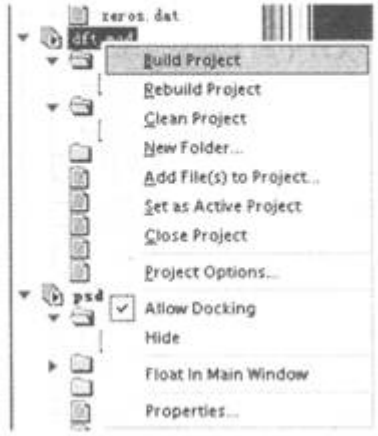

图 5-15　鼠标右键方式编译链接工程

在编译链接过程中，输出窗口中会显示状态信息。如果编译链接错误，输出窗口将会告知“编译链接失败(Build was unsuccessful)”，如图 5 一 16 所示，且显示出错信息和错误类型，用鼠标双击出错信息行，IDDE 会自动打开出错的源代码文件，并跳转到与错误信息相关的代码位置。具体的错误所在需要用户自己进行判断。

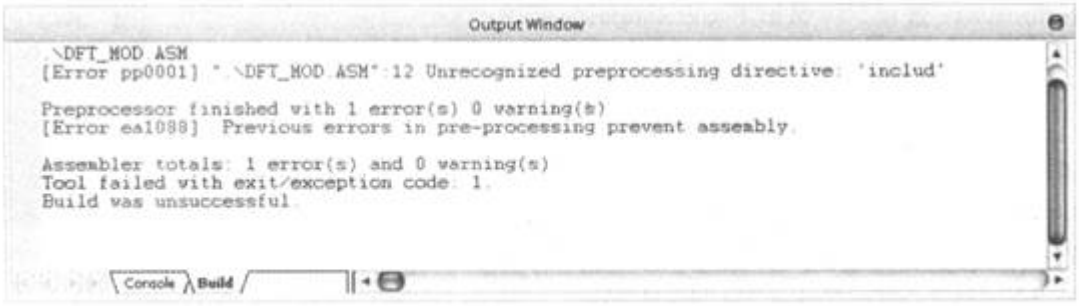

图 5-16　编译链接失败错误信息

而如果编译链接过程成功，那么在输出窗口的“编译链接(Build)”中将会显示“编译完成(Build bompleted successfully)”等相关信息，如图 5—17 所示。

值得注意的是，输出文件类型(工程选项中)必须指定为“Executable File”类型(＊. dxe)，且工程类型为 Debug 类型时，才能产生可进行 Debug 调试的输出文件。

第六步　建立调试会话(Debug Session)和加载可执行文件

在编译链接成功之后，VisualDSP++将生成处理器可执行文件。这种可执行文件可以在 VisualDSP++ 自带的模拟器环境下运行，也可以由计算机通过仿真器提供给目标板上的处理器，然后在处理器中执行。无论是在模拟器环境下还是在仿真器中，都需要建立调试会话，才能让该文件执行。建立会话完成后，由加载器将可执行文件调入模拟器或者通过仿真器加载给目标板上的处理器内进行执行。会话的相关操作请参见 5.4 节。

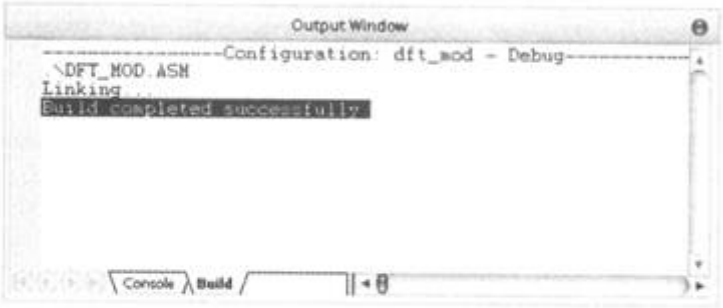

图 5-17　编译链接成功显示信息

VisualDSP++在工程编译链接完成后，加载器将根据用户建立的会话将编译链接生成的文件加载到模拟器或者通过仿真器加载给用户目标板上的处理器中。加载完成后，VisualDSP++将在输出窗口中显示加载完成信息，如图 5—18 所示。

图 5-18　加载器加载完成信息

第七步 运行和调试(Debug)程序

在加载器完成将生成的可执行文件的加载工作后，就可以用 VisualDSPC++中的 Debugger 工具来调试该工程了。

通过单击工具栏上的图标或选择 Debug 菜单中的子菜单，就可以对程序进行运行、停止等调试操作，如图 5-19 所示。用户根据需要不断对程序进行修改、完善和优化。

如果工程不是当前的(有过期的源文件或 Dependency 信息)，IDDE 会建议你先编译链接此工程，然后再启动 Debugger 工具并把可执行文件加载到 Debugger 工具中。

图 5-19　调试菜单

第八步　编译链接 Release 版的程序和生成加载文件

用户在对调试版的程序调试完成后，就基本完成了对 DSP 应用程序的基本开发，接下来，可以将调试版本的程序优化后生成正式版本程序。在生成正式版本程序后，如果用户开发的工程是需要应用在硬件平台上的，那么用户还需要将生成的正式版程序编译生成处理器的加载文件，提供给处理器系统中的程序加载方，实现处理器系统运行程序的加载。

生成正式版程序和加载文件可以通过以下简单的步骤完成。

1）选择 VisualDSP++主界面中的 Project 下拉菜单，然后单击 Project Configurations，将其设置成“Release”，或者在 Project 下拉菜单中选择 Option，将其中 setting for 的内容修改

为“Release”即可；

2）选择 VisualDSP++主界面中的 Project 下拉菜单中的 Options，将其中 Type 的内容修改为 Load File；

3）在 Options 窗口中，选择 Load 选项，根据用户系统的要求对所生成的加载文件进行设置，设置选项如图 5-20 所示；

4）编译链接该工程，VisualDSP++将生成正式版本的程序。

经过前面八个步骤，可以认为 DSP 处理器程序开发完成。

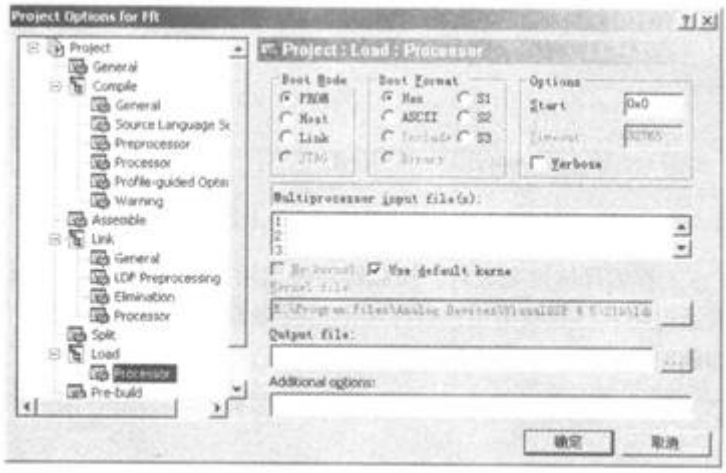

图 5-20　加载文件选项窗口

（四）Debugger 工具

Visual DSP++的 IDDE 中已经集成了 Debugger 工具。Debugger 是 WINDOWS 窗口操作界面，操作使用非常方便。在 Debugger 工具下，可以直接应用 ADI 公司的模拟器(Simulator)和仿真器(Emulator)工具。

4.1、设置调试会话

进行 Debugger 调试的第一步是必须先设置好调试会话(Debugging Sessions)。在调试会话中主要是设置调试的目标和调试所使用的工具的。Debugger 工具支持的会话类型包括硬件仿真调试会话和软件仿真调试会话。硬件调试会话必须有硬件系统的支持，也就是 VisualDSP++软件会检测所需调试的目标硬件系统，而软件调试会话不需要硬件系统支持，

是由 VisaulDSP++自带软件模拟器工具通过计算来模拟处理器的工作。

1、新建调试会话的设置

新建调试会话的设置步骤如下。

（1）单击 VisualDSP++主界面的。“会话(Session)”，然后选择“New Session''，IDDE 将弹出新建会话向导，如图 5-21 所示。

新建会话向导有三个步骤，分别是：处理器选择(Select Processor)、连接类型选择(Select Connection Type)和平台选择(Select Platform)。

处理器选择如图 5-21 所示，主要用于设置所建立的调试会话是针对何种处理器，处理器型号的选择等。在该窗口中，主要选项如下：

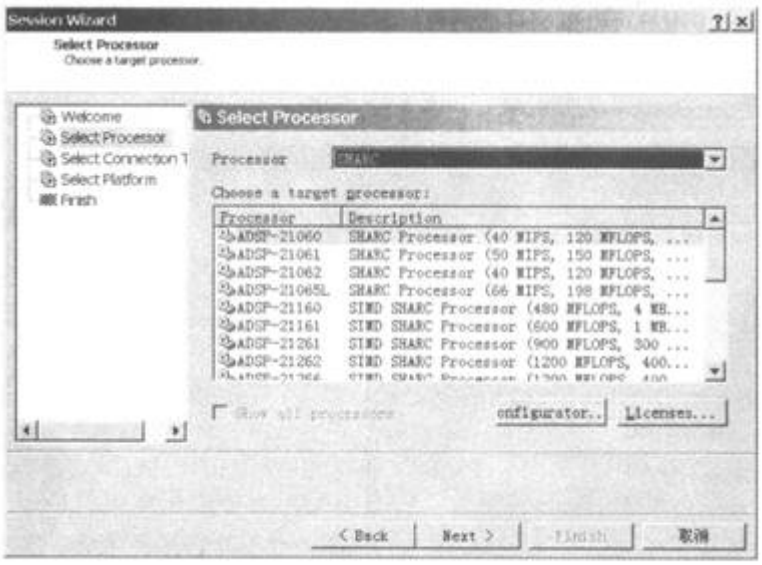

图 5-21　新建会话向导

1）Pmcessr 用于选择会话的目标处理器类型，有 Blackfin、SHARC 和 TigerSHARC 三种类型。用户需要根据所调试的目标处理器进行选择。

2）Choose a target processor——用于选择具体的处理器型号，该选项的内容将随 Processor 选项的处理器类型选择变化而变化，比如，如果 Processor 中选择为 TigerSHARC 系列，那么在 Choose a target processor 中将只会显示 TigerSHARC 系列处理器，只有 ADSP TSl01、ADSP TS201、ADSP TS202 和 ADSP TS203 等，同样如果选择为 SHARC 类型，那么该窗口中将显示 SHARC 系列的相关型号。

3）Configurator ——用于配置非模拟器环境下的会话。由于在模拟器环境下是通过计算机来模拟处理器的运行，所有并不需要使用硬件配置，但如果用户开发的是非模拟器的目

标，如 EZ-KIT 评估板或者用户设计的处理器硬件系统，那么就需要使用 Configurator。因此，对于非模拟器平台，只有在 Configurator 中定义了的非模拟器平台才可以在新建会话向导中选择，否则用户的目标系统将无法正常使用。VisualDSP++Configurator 窗口如图 5-22 所示。

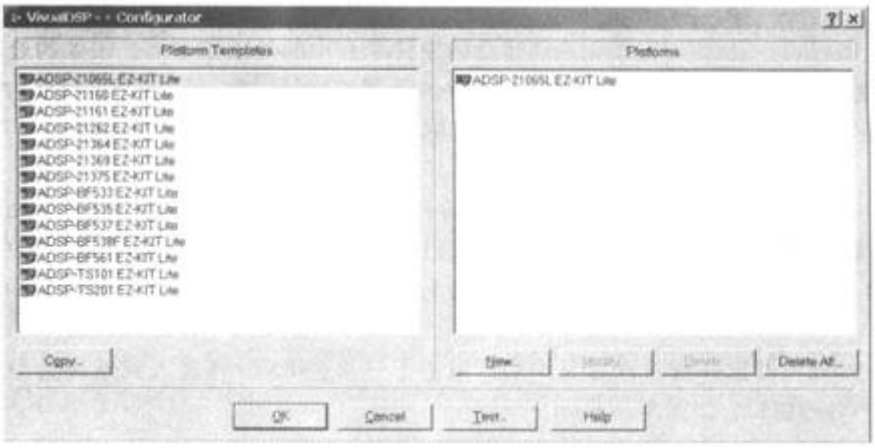

图 5-22　VisualDSP ++ Configurator 窗口

在 Configurator 中用户可以根据自己的目标系统设置所对应的平台，而且在 Configurator 中直接带有了各种处理器评估板系统的平台。用户如果建立了自己的目标系统平台，通过单击“新建(New)”按钮将弹出新建仿真平台窗 13，如图 5-23 所示。

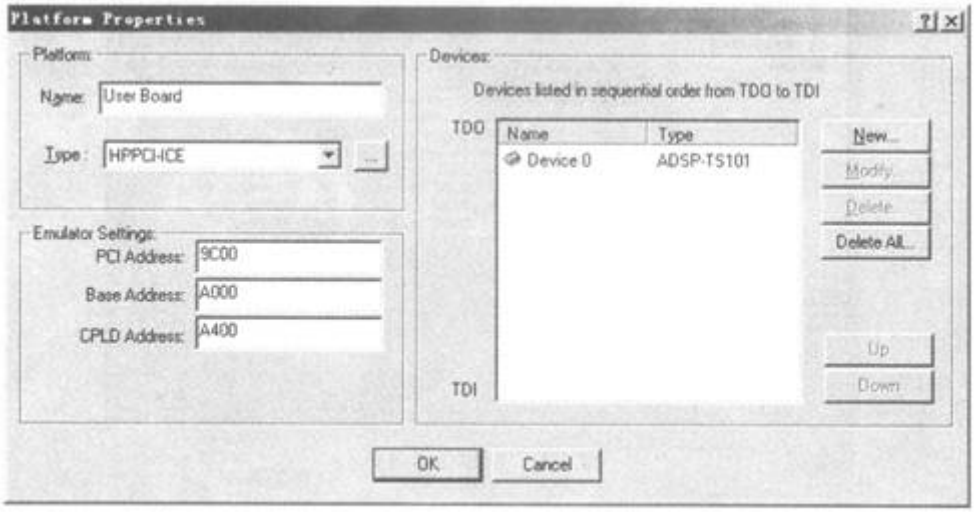

图 5-23　新建仿真平台窗口

在该窗口中，有三个参数组，分别是：平台(Platform)、仿真器设置(Emulation Settings)和器件（Devices）。

平台(Platform)——包含两个参数：名称(Name)和平台类型(Type)。名称用来为所建立的平台命名，用户可以随意命名。平台类型有：通过串口和 USB 方式连接的各种处理器的 EZ—KIT、HP PCI 仿真器、HP USB 仿真器等，该类型的选择需根据用户的硬件系统进行选择。

仿真器设置(Emulation Settings)——用于设置用户计算机上安装的仿真器的相关硬件地址，该参数随用户的仿真器选择不同而不同。在正常情况下，该窗口会自动识别计算机上安装的仿真器而直接提取仿真器相关的地址参数，因此基本上用户可以不用修改该部分内容。如果用户希望自己修改仿真器的地址，那么用户需要通过操作系统中的硬件设备管理来查询仿真器相关的地址。

器件(Devices)——用于设置用户所建立的硬件平台中的处理器型号，该处的处理器型号必须与用户硬件平台中的处理器、用户建立工程的处理器型号一致。可以通过新建(New)、修改(Modify)、删除(Delete)和全部删除(Delete A11)等方式。图 5-24 为新建仿真平台中的器件选择窗口。

4）License ——用于管理 VisualDSP++的授权信息，如图 5-25 所示。VisaulDSP++在安装后需要输入软件序列号，并进行授权验证后才能正常使用，否则 VisualDSP++将只能试用 30 天，且部分使用功能受限。在进行新会话设置的时候，对该窗口可以不予以设置。

（2）在对处理器选择窗口设置完毕后，单击下一步(NEXT)将进入到连接类型选择(Select Connection Type)窗口，如图 5-26 所示。

图 5-24　新建仿真平台中的器件选择窗口

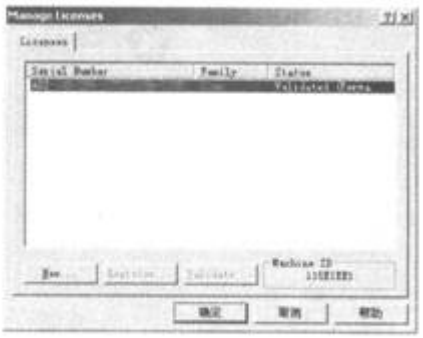

图 5-25　VisualDSP ++ 授权管理窗口

连接类型选择窗口主要是用于设置所建立的会话使用的连接方式，在一般正常安装的 VisualDSP++中会有四种连接方式：评估板系统(EZ—KIT Lite)、仿真器(Emulator)、模拟器(Simulator)和遗留目标(Legacy Target)。根据安装 VisualDSP++的授权不同，在该窗口中有些选项可能不能选择。

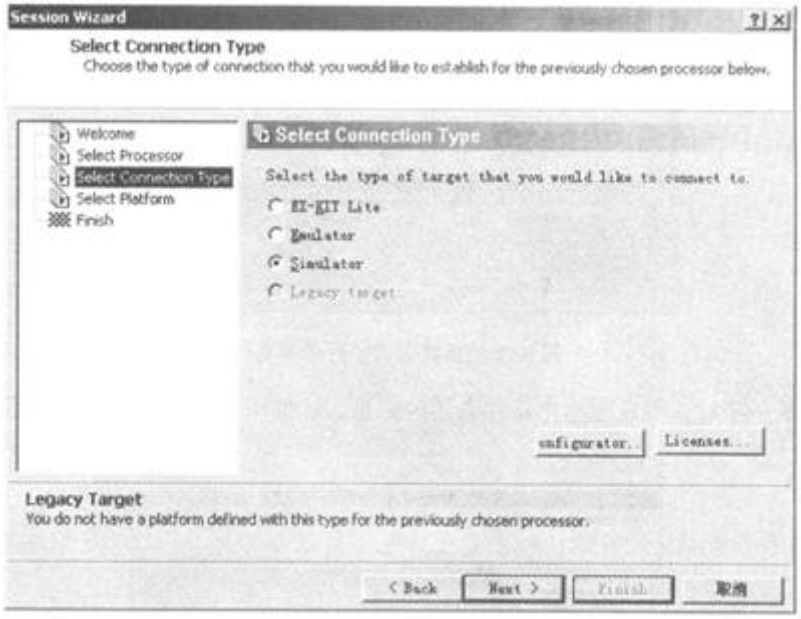

图 5-26　连接类型选择窗口

用户根据需要自行选择其中的一种即可。对于模拟器方式，不需要硬件支持，完全由计算机来模拟处理器的运行，而对于评估板系统和仿真器连接方式是需要用户提供硬件平台进行连接的。如果用户没有硬件平台，而在此处选择为评估板系统和仿真器连接方式，那么在建立会话完成后，由于 VisualDSP++检测不到硬件设备，VisualDSP++将会弹出错误信息，如图 5-27 所示，提示用户连接不上硬件平台。因此如果用户在没有硬件系统时，只能将类型选择作为模拟器方式。

图 5-27　会话连接失败提示信箱

（3）用户在确定会话的连接方式后，单击下一步，将显示所建立的会话使用到的平台，如图 5-28 所示。对于大多数处理器而言，除针对部分 Blackfin 系列的处理器建立的平台该窗口中有两个选项，该窗口中只有单一的选项。因此对于大多数情况，该窗口使用默认选项即可。

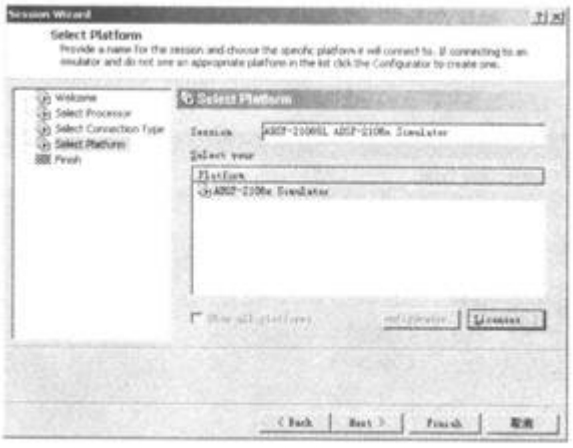

图 5-28　新建会话平台选择窗口

（4）单击下一步后，将显示用户新建会话的信息，以便用户检验所建立的会话是否正确，如图 5-29 所示。

图 5-29　会话信息窗口

用户在确定所建立的会话信息无误后，单击完成(Finish)后，VisualDSP++将根据用户的设置建立会话。

在会话建立完成后，Visual DSP++将把刚建立完成的会话作为当前的会话平台。另外用户可以从下拉菜单 Session 中的会话选择，在会话列表中查看到用户建立的会话和选择其他以前建立的会话。

2、打开已经存在的调试会话

如果用户已经建立过多个会话环境，用户可以对存在的所有会话进行管理并在其中进行选择。选择已经存在的会话，拥户只需要通过 VisualDSP++主界面的 “会话(Session)”下拉菜单，单击“会话选择(Select Session)”，然后在选择所需要的会话即可，如图 5-30 所示。

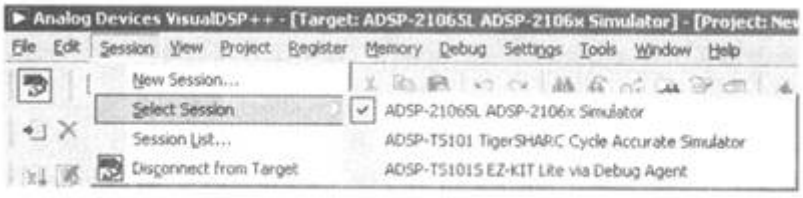

图 5-30　会话选择菜单

用户通过会话列表(Session List)可以完成对已经存在的会话进行管理和选择。单击会话列表(Session List)后，VisualDSP++将弹出会话管理窗口，如图 5-31 所示。在该窗口中可以新建、删除和激活已经存在的会话。

图 5-31　会话管理窗口

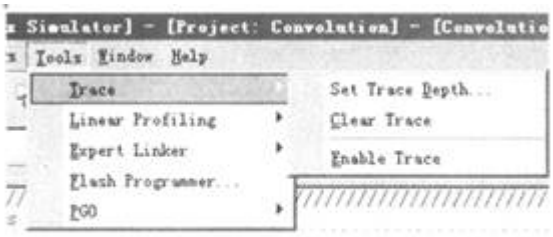

图 5-32　跟踪和线性剖析菜单

4.2、程序执行操作

Debugger 中的程序执行命令在 Debug 下拉菜单中，如图 5—19 所示。这些命令在工具栏中也有相应的快捷按钮。下面简单说明一些常用的执行命令。

1、运行（Run）：

运行程序直到遇到某种条件才停止，程序停止的条件可以是执行到断点或用户干预等。当程序处于停止状态时，在 VisualDSP++中所有的已经打开的各种相关窗 El 的内容都更新为程序运行后的值。

2、暂停（Halt）

程序在执行过程中，用户可以通过暂停功能来暂停程序的执行。当程序处于暂停状态时，在 VisualDSP++中所有已经打开的各种相关窗口的内容都更新为程序运行后的值，且状态条显示为当前程序停止的地址。

3、执行到光标所在位置（Run To Cursor）

该功能将使程序运行，直到程序执行到光标所在位置时，程序暂停。光标的位置可以在源文件窗或反汇编窗中设置。

4、执行 1 行程序（Step Over）

仅仅执行 1 行 C 语言程序行，仅用于 C 语言程序。

5、单步执行程序（Step into）

单步执行程序。每执行 1 步，所有已经打开的相关窗口的内容都更新。

6、单步执行当前函数（Step out of）

单步执行当前函数，直到返回到它的调用程序。仅用于 C 语言程序。

7、复位（Reset）

通过 VisualDSP++对处理器进行复位，如果当前会话处于模拟器状态，那么 VisualDSP++将使模拟器所有状态复位，而如果 VisualDSP++处于硬件会话环境下，那么 VisualDSP++将通过仿真器向目标系统上的处理器发送复位信号。在复位后，必须使用加载器将程序重新加载才可以运行。

4.3、程序性能分析操作

VisualDSP++调试器中提供了两个工具来分析程序的执行情况：跟踪(Trace)和线性剖析(Linear Profiling)。这两个命令都位于 VisualDSP++的“Tools”下拉菜单中，如图 5-32 所示。

1、跟踪（Trace）

提供对程序执行指令的跟踪，结果显示程序如何执行到某一地址上，显示程序的读、写和存储器访问。通过如下步骤来设置 Trace 功能并显示其结果。

- 1）单击 VisualDSP++主界面 Tool 下拉菜单中的 Trace，选择 Enable Trace 激活跟踪操作；
- 2）单击 VisualDSP++主界面 Tool 下拉菜单中的 Trace，选择 Set Trace Depth，然后设置 Trace Buffer Depth，如图 5—33 所示，选择用户定义的跟踪深度或最大跟踪

深度，VisualDSP++默认为最大跟踪深度；

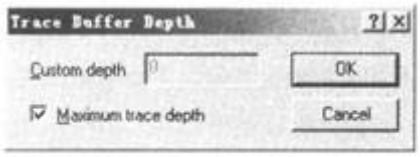

图 5-33　跟踪深度设置

- 3）单击 VisualDSP++主界面 View 下拉菜单中的 Debug Windows，选择 Trace 打开跟踪显示窗口；
- 4）运行程序，在通过对程序设置断点或者使用 Halt 命令来停止程序的执行后，通过跟踪窗口可以查看跟踪的执行结果，如图 5-34 所示。

图 5-34　跟踪执行结果

跟踪的结果含有如下内容：

访问类型（RD 或 WR）

内存类型（PM 或 DM）

方括号中的地址（[ ]）

读写的数据值

对于跟踪分析工具的使用需要注意以下几点：

1）对于 SHARC 及 TigerSHARC 处理器，系统的虚拟内存限制了深度；

2）对于 Blackfin 系列处理器，在模拟器中不支持跟踪，但是在仿真器中支持跟踪。

2、剖析（Profiling）

VisualDSP++ 的剖析工具 Linear Profile 是用来分析程序的运行时间特性，通过线性统

计剖析，可以分析出每段程序的耗时量和在整个程序运行中所占用的比例，为用户分析程序的性能、优化程序提供帮助。VisualDSP++4．5 提供了线性统计剖析工具，即对运行的程序做统计分析，计算出每条指令占用执行程序中的百分比和运行的周期数，并以统计表的形式给出。

完成一次剖析的基本步骤如下：

1）编译和链接工程完成；

2）单击 VisualDsP++主界面 Tool 下拉菜单中的 Linear Profiling，选择 New Profiling 建立和激活新的剖析；

3）在新建剖析窗 VI 中的空白处单击鼠标右键，选择 Properties…，如图 5-35 所示。

4）在选择 Properties…后，VisualDSP++将弹出剖析参数设置窗口，如图 5-36 所示。在该窗口中可以对全部程序进行分析，也可以对 C / C++子函数进行分析，还可以指定程序的址段进行分析。

5）在对所需分析的程序段设置后，运行程序，那么在剖析窗口中将显示线性统计剖析的结果，如图 5-37 所示。剖析窗口中的左半部分为用户所分析的程序段相关的结果，使用鼠标双击相关的函数，那么在窗口的右半部分将显示出对该函数中每条指令进行线性统计的结果。在该窗口中显示的结果默认为每个子程序或者每条程序占整个剖析运行程序段的百分比，通过修改剖析参数，也可以显示每条指令执行的周期数。

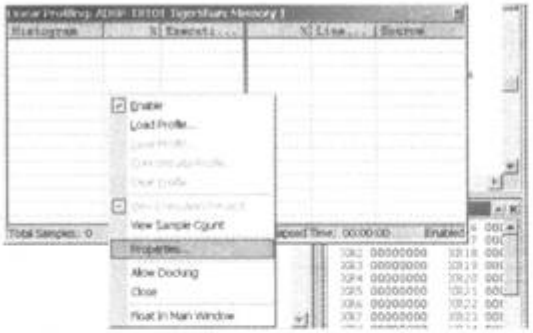

图 5-35　新建剖析窗口

图 5-36　剖析参数设置窗口

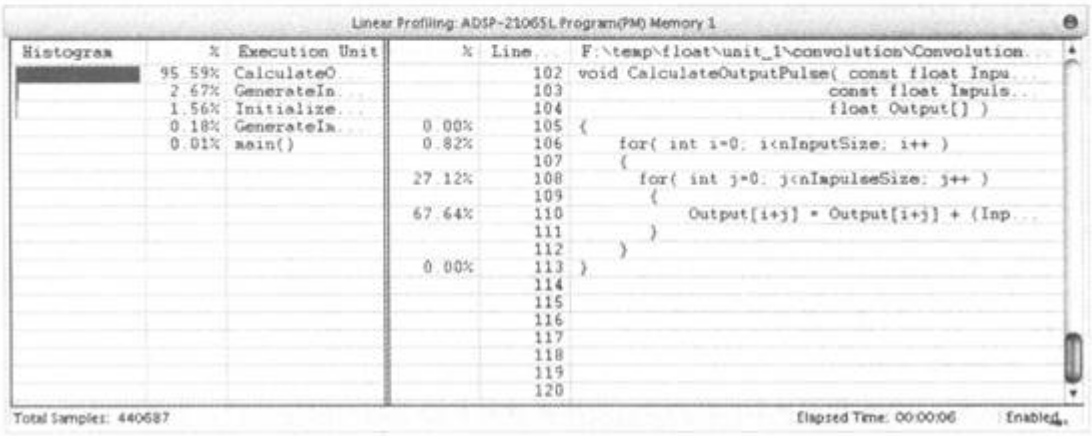

图 5-37　线性统计剖析的结果

值得注意的是，在每次重新编译的时候，由于 VisalDSP++会使用到预编译器，因此在每次编译后，剖析窗口中的数值都会发生变化。另外，如果对剖析窗 VI 中的数值不采用清除操作，那么剖析的结果将一直进行累计。所以在实际运用过程中，无论是程序重新编译还是重新执行程序，都应当先将剖析窗口中原有的数值清除后再运行程序进行分析，否则分析出来的数据有可能不准确。

清除剖析窗口中原有数值的方法非常简单，在剖析窗口中单击鼠标右键，选择菜单中的 Clear Profiling 即可。

4.4、设置观察点

观察点(Watch Point)与断点(Break Point)功能非常相似，断点可以在程序的任意位置上设置，使程序暂时停止执行。而观察点可以设置某种条件，当满足条件时才暂停程序的执行，如存储器读写、堆栈弹出等。

通过下列步骤来设置观察点。

1）选择 Visual DSP++主界面下拉菜单 Settings 中的 Watch points，会出现一个 Watch points 对话框，如图 5-38 所示。

图 5-38　观察点窗口

2）该窗口用于设置使程序停止的条件，使程序停止运行进行判断的方式有三种：寄存器(Registers)、硬件堆栈(Hardware Stacks)和存储器(Memory)。下面以寄存器页面为例进行说明，在寄存器类型中主要包括如下一些设置：

Register 一列出了所有寄存器，提供给用户用于选择需要的寄存器进行条件设置。

Watch For Read——当指定寄存器的读操作满足条件，就暂停程序的执行。读取到数值判断有四种：读到任意值(Any Read)、读入特定值(Read value)、读入值作为某种计算的操作数(Read in computation)和读入未定义的值(Read uninitial)。

Watch For WIite——当指定寄存器的写操作满足条件，就暂停程序的执行。条件判断有四种：写任意值(Any write)、或者写指定的值(Write vaIue)、写的值作为某种计算的操作数(Write in computation)、写未定义的值(Write uninitial)。

Value——读或者写操作的指定值。

Format——读或者写的指定值的格式，可以选择二进制、整数、浮点数等。

Add、Edit、Delete——对观察点列表进行添加、删除、编辑等管理操作。

3）对观察点判断条件设置完毕后，单击 Add 按钮，将所指定的观察点加入到观察点列表中。利用 Add 按钮可以加入多个观察点到列表中。

4）单击 OK 按钮，完成设置。

5）运行程序，当程序运行时，满足所设置的任意一个条件时，VisualDSP++将自动停止运行程序。

4.5、模拟硬件环境

为了方便用户在模拟器环境下更好的调试用户所编写的程序，VisualDSP++的调试器提供了 3 种硬件环境的方式模拟：

Interrupts——中断，模拟在程序的执行过程中产生外部随机中断。

Streams——数据流，模拟处理器通过外部端口进行数据传输。

Load Sim Loader——模拟处理器通过 EPROM 或主机等方式的加载过程。

上面三种硬件模拟均在 VisualDSP++主界面的 Settings 下拉菜单中，用户通过单击 Setting 按钮就可以对其进行设置，下面分别介绍。

1、中断（interrupts）模拟

中断模拟用于模拟程序在执行过程中处理器外部产生随机中断，这对调试中断服务程序是非常有用的，其设置窗口如图 5—39 所示。

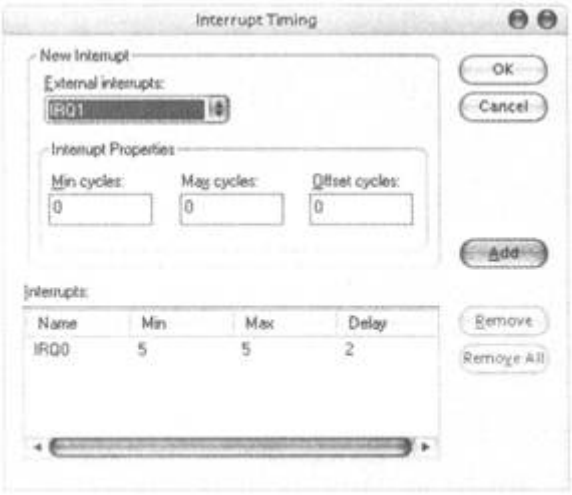

图 5-39　中断模拟设置窗口

该窗口中的主要选项意义如下：

External interrupts——外部中断类型，用于设置外部中断的类型，包括 FLAG 中断、IRQ 中断、定时器中断等，具体的中断类型与用户选择的处理器型号有关，不同的处理器所包含的外部中断类型也有所不同。

Min cycles——中断信号产生的最小指令周期间隔。

Max cycles——中断信号产生的最大指令周期间隔。

Offset cycles——在第一次中断发生之前的指令周期数。

Interrupts——显示设置完成的模拟中断及其参数。

Add、Remove、Remove All——用于对设置完成的模拟中断进行添加、删除的管理操作。

在对模拟断设置完毕后，直接运行程序即可，那么中断模拟器将会按照所设置的方式产生中断。

值得注意的是，无论是重新执行程序、重新编译链接工程还是重新加载已编译过的程序，模拟中断都并不会取消，只有在模拟中断设置窗口中，将其删除，那么才能停止产生模拟的中断。当然如果重新启动 VisualDSP++是可以关闭模拟中断的。

2、数据流（Streams）模拟和 DMA 模拟传输

1）SHARC 和 Blackfin 系列处理器的数据流传输

数据流模拟是用于模拟处理器通过外部端口进行数据传输过程的。数据流模拟可以模拟处理器的外部数据总线、链路口、串口等端口的数据传输。该功能对于在模拟器环境下调试处理器的 DMA 传输非常有效。

（1）单击 Visual DSP++主界面的下拉菜单中的 Stream，将弹出数据流管理窗口，单击数据流管理窗口中的添加(ADD)按钮，将显示新建的模拟数据流传输的设置窗口，如图 5-40 所示。数据流管理窗口可以对已经存在的数据流模拟进行修改或删除等操作。

图 5-40　数据流模拟窗口

（2）新建数据流设置窗口中的主要包含了数据流的源和目的、类型端口等。主要参数说明

如下：

Source／Destination——数据传送的源／目的，用以设置数据传输的来源和目的，只有数据传输的源和数据传输的目的均设置正确才能进行数据流传输。

Processor——给出针对数据流模拟的器件，该参数默认为会话选择的处理器型号。

Device——数据流使用的设备，也就是数据流传输使用的端口或者存储器。

Address——数据流使用的 I／O 地址。

File／Browse——用于采用文件方式实现数据流模拟中打开数据文件。

Format——数据传输使用的格式，数据流可以采用十六进制、十进制、二进制等整数类型和浮点格式进行数据传输。

Circular——设置数据文件读取过程中，读取到数据结束后是否采用循环方式再从头读取数据，如果该选项选中，则支持循环读取方式，否则不支持循环读取数据文件。

（3）对数据流设置完成后，单击 OK 按钮，退出窗口；

（4）运行程序，那么在程序运行过程中，处理器通过指令或者 DMA 均可以实现对数据的获取或者写出。

处理器读取数据流一般是从外部文件将数据读入，处理器写出的数据也将写出到文件中，用户通过相关的文件可以查看处理器传输的数据。

2）TigerSHARC 系列处理器的 DMA 模拟传输

值得注意的是，前面讲述的数据流模拟传输的设置只针对 SHARC 和 Blackfin 系列处理器有效，而对于 TigerSHARC 系列处理器，VisualDSP++只提供 DMA 模拟传输，且相关操作如下。

（1）如果用户选择了 TigerSHARC 系列处理器的会话环境，那么在 VisualDSP++主界面的下拉菜单 Settings 中的 Stream 选项将不可选择，用户应当选择 Simulator，然后单击 Config DMA File I／O…，将弹出 TigerSHARC 系列处理器的 DMA 传输模拟器设置窗口，如图 5-41 所示。

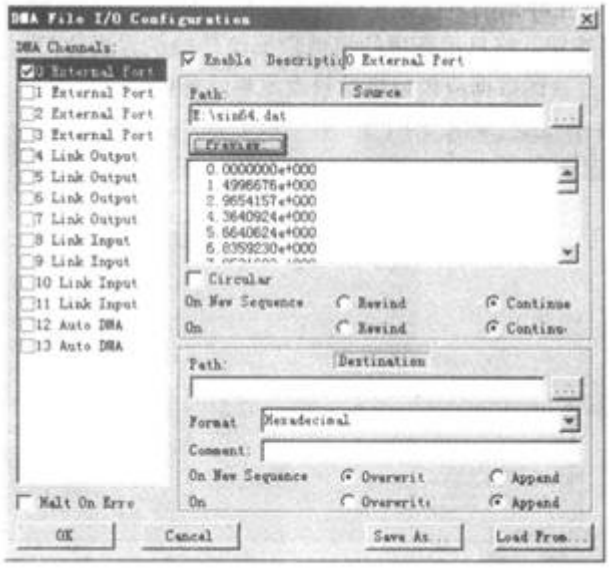

图 5-41　TigerSHARC 系列处理器的 DMA 传输模拟器设置窗口

（2）在该窗口中主要包含对 DMA 的数据源和数据目的进行设置的参数，主要选项如下：

DMA Channels——DMA 通道选择。TigerSHARC 系列处理器带有 14 个 DMA 通道，通道 0～3 分配给了处理器外部总线端口，通道 4～7 分别分配给了 4 个链路口的发送端口，通道 8～11 分别分配给了 4 个链路口的接收端口，通道 12～13 分配给了 AutoDMA 通道。用户根据模拟的需要，选中所需的 DMA 通道即可。该窗口支持对多个 DMA 同时模拟，但用户需要对每个 DMA 通道进行设置。

Enable Description——DMA 通道使能描述，用于显示选中的 DMA 是否被使能。如果相应的 DMA 通道使能，那么将使该选项使能。

Halt On Error——选中该选项，若 DMA 在传输过程中遇见任何错误，DMA 传输将停止。

Source——用于设置 DMA 传输的数据源，如果模拟 DMA 通道从处理器外部获取数据，设置该选项，如果模拟 DMA 通道从处理器内部向外部送出数据，那么该相关参数将不用设置。它包含以下参数：

Path——用于设置 DMA 数据传输的数据文件存放的路径。

Preview——对用户选中的数据文件进行预览，方便用户确认所需传输的数据是否正确。支持十六进制、十进制和浮点等处理器支持的所有格式。

Circular——设置数据文件读取过程中，读取到数据结束后是否采用循环方式再从头读取数据，如果该选项选中，则支持循环读取方式，否则不支持循环读取数据文件。

On／On New Sequence——进行新的 DMA 传输采取的方式，Rewind 为从数据文件的开

头进行数据读取；Continue 从上次 DMA 传输完毕的数据位置接着读取数据。

Destination——用于设置 DMA 传输的数据目的，如果模拟 DMA 通道从处理器内部向外部送出数据，设置该选项，如果模拟 DMA 通道从处理器外部获取数据，那么该相关参数将不用设置。它包含以下参数：

Path——用于设置 DMA 数据传输的数据文件存放的路径。

Fomat——用户通过 DMA 通道输出数据的格式，支持十六进制、十进制和浮点等处理器支持的所有格式。

Comment——写入到输出文件中的注释，在该窗口中的注释信息将写在输出数据文件中的开头位置。

On／On New Sequence——进行新的 DMA 传输的采取的方式，Rewind 为从数据文件的开头进行数据写操作，即覆盖上次 DMA 传输写出的数据；Append 为将 DMA 传输悬挂起来。

3）用户在设置该窗口中的参数完成后，单击 OK 按钮即可。数据的传输由用户的程序来控制开启，并且只能采用 DMA 方式进行数据传输，处理器通过指令访问将无效。

值得注意的是，由于对数据的传输需要通过用户的程序控制开启，因此在该窗口中的设置应当与用户程序中所设置的 DMA 通道及其方向一致，否则将不能模拟 DMA 的正常传输。

3、Load Sim badef 模拟

Load Sim Loader 是用来模拟 EPROM 或主机给处理器加载．1dr 文件的过程，为用户设计实现处理器加载提供帮助。

通过下列步骤可以建立一个处理器 EPROM 加载。

1）选择 VisualDSP++主界面的下拉菜单 Settings 中的 Load Sim Loader，如图 5-42 所示。

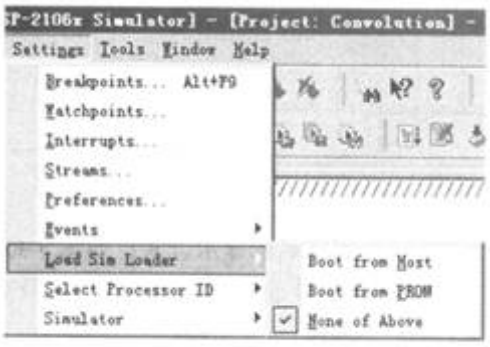

图 5-42　Load Sim Loader 菜单

2）在该菜单中有两个选项：从主机加载启动(Boot from Host)和从 PROM 加载启动(Boot from PROM)，分别用于模拟从主机对处理器进行加载启动和让处理器从 PROM 加载启动。由于不同处理器的加载方式不同，因此对于不同的处理器，该菜单中的内容也有所不同，具体的内容与处理器所具有的加载方式有关，例如对于 ADSP-21062 具有链路口加载方式，则该菜单中也就包含了连接口加载方式的模拟。

无论模拟主机加载启动还是 PROM 加载启动，选择以后，VisualDSP++都将提示选择所需要加载的文件，加载文件是后缀名为.ldr 的文件，选择加载文件完毕后，VisualDSP++将提示用户单击调试(Debug)菜单中的复位(Reset)按钮，然后弹出提示信息。如图 5-43 所示，在用户单击确认并复位完成后，VisualDSP++将进入模拟加载过程。

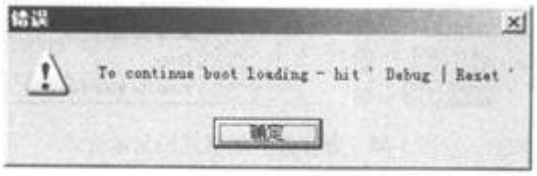

图 5-43　load Sim Loader 提示信息

3）进入模拟加载过程后，用户直接运行程序，VisualDSP++将自动完成加载过程。用户通过单步执行程序，可以观察到处理器的模拟加载过程。

4）如果要从模拟加载环境下退出，需要单击 VisualDSP++主界面的下拉菜单 Settings 中的 Load Sim Loader，并将其设置成无加载(None of Above)方式。

4.6、寄存器窗口操作

寄存器操作也是 DSP 调试过程中经常使用的。寄存器的显示通过在 VisualDSP++主界面的寄存器(Register)下拉菜单中进行选择即可，如图 5 坩所示。寄存器下拉菜单中一般包含了核寄存器、系统寄存器、IO 端口寄存器或者附属设备寄存器等，具体内容随处理器型号

而异。

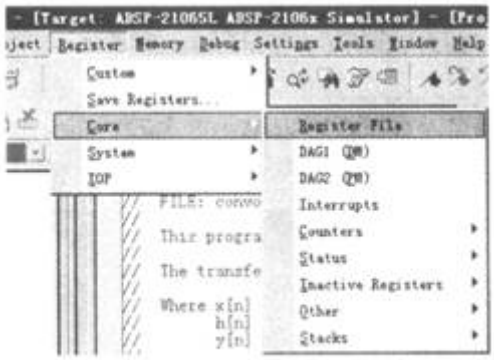

图 5-44　寄存器菜单

4.7、存储器窗口操作

存储器窗口不但像寄存器窗口那样，可以提供数据格式和编辑操作，还提供跳转(Goto)、查找(Search)、填充(Fill)、导出(Dump)等功能。

下面分别介绍存储器操作。

1、存储器查看

通过单击 VisualDSP++下来菜单存储器(Memory)，选择存储器查看方式，如图 5-46 所示．即可弹出存储器窗 VI，如图 5-47 所示。

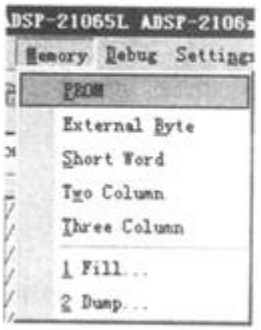

图 5-46　SHARC 系列存储器查看方式

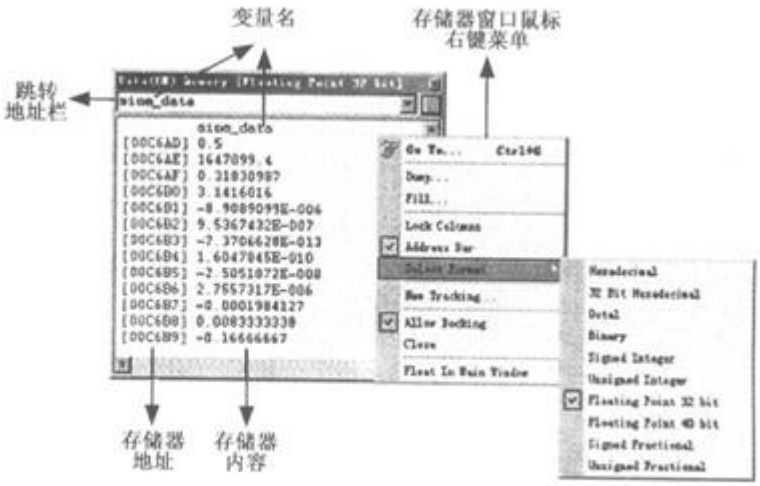

图 5-47　存储器窗口及其鼠标右键菜单

值得注意的是，不同的处理器片内存储器的物理结构不同，因此对于不同处理器，存

储器下拉菜单的内容也有所不同。例如，SHARC 系列处理器的片内存储器是按照 16 位设计的，因此 SHARC 系列处理器的存储器查看方式有短字(Short Word)、32 位字(Two Column)和 48 位字（Three Column)等查看方式；而 TigerSHARC 系列处理器的片内存储器是按照 32 位设计的，因此 TigerSHARC 系列处理器的存储器查看方式只有 32 位查看方式。

2、改变存储器数据格式

类似对寄存器窗口的操作，选中所需修改地址对应的数值，双击鼠标左键，高亮后填人修，然后修改即可。

3、跳到某一地址上查看

（1）直接在存储器窗口的跳转地址栏中敲人所需跳转的地址即可，该地址栏支持 16 进制地址输入和标号选择；

（2）在激活的存储器窗 VI 上单击鼠标右键，在菜单中选择 Go To 命令，会出现一个 GO To Address 对话框，如图 5-48 所示。在此对话框中敲人十六进制的地址或通过 Browse 从标号列表中选择一个标号，单击 OK 按钮即可。存储器窗口中的显示内容会变成指定地址的存储器的内容。

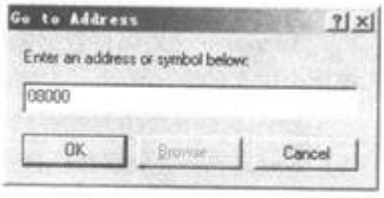

图 5-48 Go To Address 对话框

4、填充或者导出存储器数据

填充是把数据填充到存储器中，导出是把存储器内容写到数据文件(. dat)中。

（1）在激活的存储器窗口中单击鼠标右键，在菜单中选择 Fill，将出现存储器数据填充对话框，如图 5-49 所示。配置该窗口中的参数，完成后单击 OK 按钮即可。

填充存储器的数据来源有两个：固定值和用户数据文件。

该窗口中的主要选项含义如下：

Address——要填充的存储器首地址。该栏中可以填人 16 进制地址，也可以填入用户程

序中定义的变量名。

Memory——要填充的存储器类型，该选项使用默认即可。

Count——要填充的数据长度。

Stride——填充存储器的地址增量。

Value——填充的数值，该功能只在采用固定数值填充时有效，也就是在“Fill from a file”选项无效的时候才能使用。

Fill From a File——当选择此项时表示利用文件数据填充存储器，否则必须指定一个值，该选项有效后，文件设置(File settings)选项才有意义。

File Name——采用文件填充时，数据文件存放的路径和文件名。

（2）在激活的存储器窗口中单击鼠标右键，在菜单中选择 dump，将出现存储器数据导出设置对话框，如图 5-50 所示。配置该窗口中的参数，完成后单击 OK 按钮即可。

图 5-49　存储器数据填充对话框

该对话框中的主要选项含义如下：

Address——要保存数据的存储器首地址。该栏中可以填入十六进制地址，也可以填人用户程序中定义的变量名。

Memory——要填充的存储器类型，该选项选择默认即可。

Format——存储器数据格式，支持十六进制、十进制、八进制、二进制、整数、浮点数等各种类型。

Count——要保存的数据长度。

Stride——保存数据的存储器地址增量。

Dump to a file——将数据保存到文件选项，该选项必须有效，否则不能正常保存数据。

Show addresses——地址显示选项，如果该选项选中，那么在导出的数据文件中，除了会将数据保留，还会将数据多对应的存储器地址也保留：且一一对应。

File Name——保留数据的文件存放的路径和文件名。

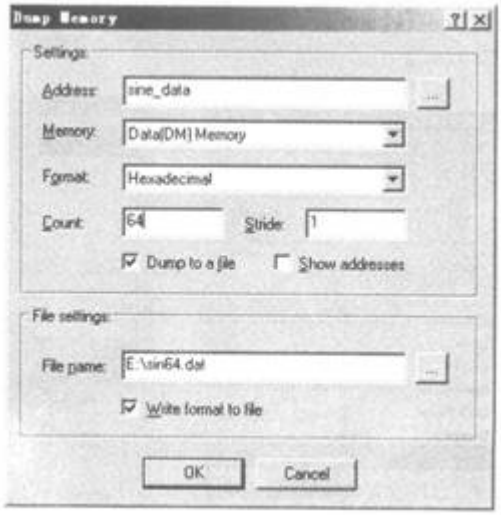

图 5-50　存储器数据导出设置对话框

5、新建跟踪（New Tracking）

可以在某一存储器窗口中输入一个表达式来进行跟踪，通过下列步骤来跟踪一个表达式。

（1）在激活的存储器窗口中单击鼠标右键，出现快捷菜单；

（2）在此菜单中选择 New-Tracking，会出现一个 Enter A New Tracking Expression 对话框；

（3）在此对话框中写入一个表达式；这个表达式可以是 C 表达式或寄存器表达式。如果是寄存器表达式，必须用$Xn 的形式；

（4）单击 OK 按钮。

6、将存储器内容画图

存储器中数据也可以采用图形的形式给出。Visual DSP++提供画图工具，将用户制定的数据进行画图，基本步骤如下。

1) 新建一个画图窗口，单毒 VisualDSP++主界面中的 View 下拉菜单，选择 Debug 选项中的 Plot，单击 New 按钮，如图 5-51 所示。

2) 新建画图窗 EI 后，VisualDSP++将弹出新建画图配置窗口，如图 5-52 所示。

该图中主要选项如下：

Data sets——数据集合，管理已经设置完毕的数据。

Add／Remove／New——绘图的新建、添加、删除等管理操作按钮。

Type——绘图的类型，即采用何种方式来绘图，常用的有：绘制线段图(Line P1ot)、X—Y 图、星座图(Constellation Plot)、眼图(Eye Diagram)、谱图(Spectrogram Plot)等。

Tide——绘图的标题。

Name——图形名称。

Memory——存储器类型，即所需绘图的数据存放的存储器类型。

Address——存储器地址，即所需绘图的数据存放的存储器起始地址。该窗口可以直接输入十六进制的地址，也可以使用程序中的变量名来替代地址。

Offsec——偏移地址，即所需绘图的数据存放的地址与 Address 中定义的地址之间的偏移量。

Count——绘图所用的数据长度。

Stride——地址增量。

DataL——数据类型。

Senings——绘图的高级设置。

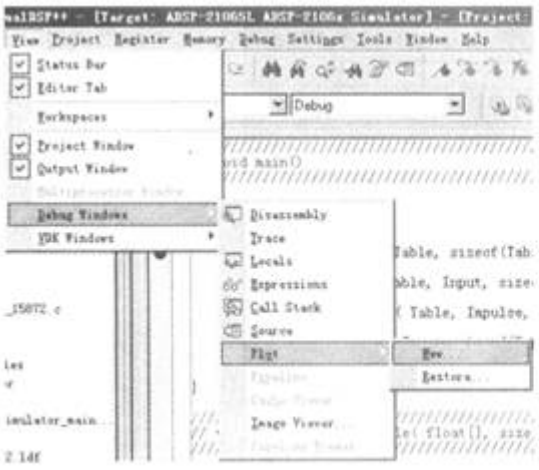

图 5-51　新建画图

图 5-52　新建画图配置窗口

3) 在对绘图进行配置完成后，单击 Add 按钮，将设置完成的绘图添加到数据集合中。如果不单击 Add 按钮将其添加，那么上一步的设置在该窗口关闭后都将无效。

4) 单击 OK 按钮，VisualDSP++将显示所绘制的图形。如图 5-53 所示。

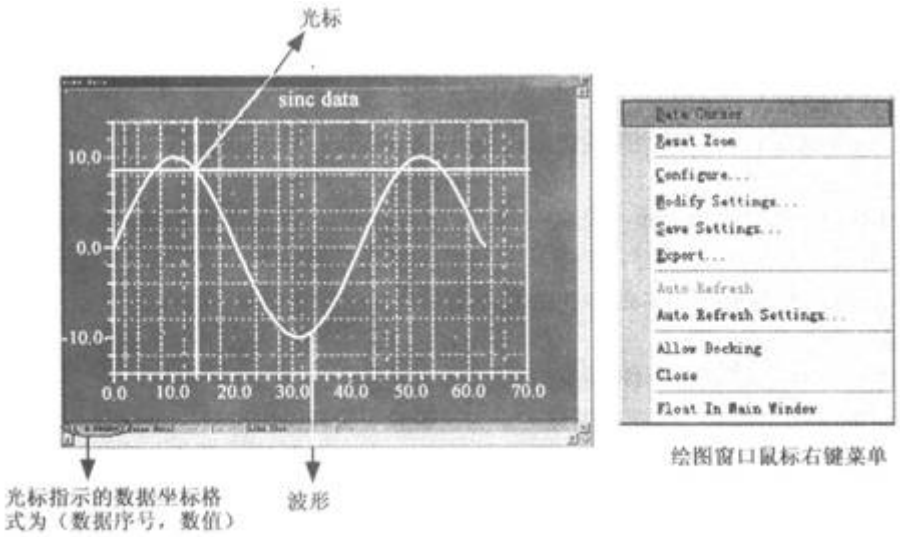

图 5-53　所绘制的图形窗口及其鼠标右键菜单

在图形窗口中，可以直接使用鼠标选择对区域图形进行放大，通过鼠标右键菜单中的 Reset Zoom 可以将图形恢复到满窗口模式。

在图形窗口的鼠标右键菜单中有光标(Data Cursor)，用户通过移动光标可以在图形窗口的左下角查看到光标位置所对应的数据序号和数值的大小。

图形窗口允许用户通过导出的方式将图形以图片或者数据的方式进行保存。单击鼠标右键菜单中的导出(Export)，将弹出导出图形设置窗口，如图 5-54 所示。

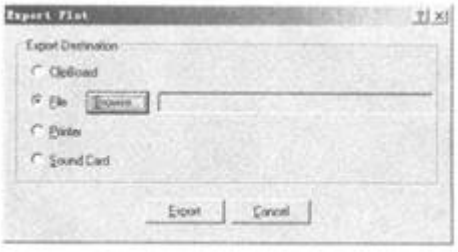

图 5-54　导出图形设置窗口

通过鼠标右键可以更改寄存器数据格式的显示方式，如图 5-45 所示。寄存器的数据格式包括：十六进制、八进制、二进制、有符号或无符号整数、32 / 40 位浮点、有符号或无符号小数等。

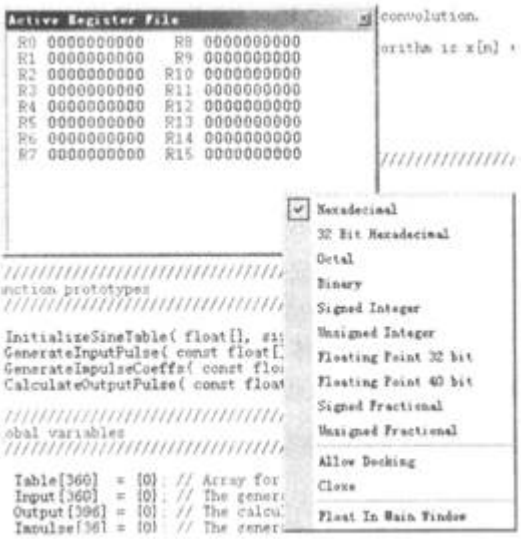

图 5-45　寄存器数据格式的显示方式

在寄存器窗口中可以改变寄存器内容显示的数据格式和修改寄存器内容。修改寄存器的值只需要选中需修改的寄存器，鼠标双击后即可对寄存器的值进行修改，当它高亮显示后，敲入新值并回车就可以了，也可以用文本编辑操作，如复制、剪切、粘贴等对寄存器的值进行修改。

（五）Visual DSP++操作使用举例

本节通过一个简单的例子让读者熟悉 VisualDSP 的操作过程，该例子使用的源文件在 VisualDSP++安装目录下的．＼Analog Devices＼VisualDSP 4．5＼21k＼Examples＼ADSP-21065L EZ—KIT Lite＼Dft_65L 目录中。

1) 在 WINDOWS 的开始菜单中选 Start＼Programs＼Aanlog Devices＼VisualDSP++4．5＼Visual DSP++Environment，打开 IDDE 主界面。在打开 IDDE 的同时也会把上次运行 IDDE 保存的工程内容打开了，因此应首先把它们关掉，并选择不要保存；

2) 选择 IDDE 主界面的下拉菜单 File 中的 New 按钮，选择 Project，启动新建工程向导。在新建工程向导中将工程名称设为 Dft，工程路径设置为 VisualDSP++安装目录下的．＼Analog Devices＼VisualDSP 4．5＼21 k＼Examples＼No Hardware Required 目录。单击下一步。

3) 在处理器选择窗口中选择使用的处理器为 ADSP-21065L，其他参数采用默认，单击下一步或者完成。将．＼Analog Devices＼VisualDSP 4．5＼2 1 k＼Examples＼No Hardware

Required＼DFT(ASM)目录下的 DFT．ASM 和 DFT．LDR 文件复制到．＼Analog Devices ＼VisualDSP 4．5＼21k＼Examples＼No Hardware Required＼DFl7 目录下。

4) 此时在工程管理窗口中包含了刚才建立的 DFT 工程，在工程管理窗口中将选择 source files，单击鼠标右键，选择 Add Files to Folder，然后将＼Analog Devices＼VisualDSP．4．5＼21k＼Examples＼No Hardware Required＼DFT 目录的源文件 DFT．ASM。并采用同样的方法把链接描述文件 DFT．LDR 添加到工程管理窗口的 ldf file 目录中。

5)设置工程选项如下：

Processor： ADSP-21062

Type：DSP executable file

Name：DFT

Build Type：Debug

6)选择 IDDE 主界面下拉菜单 Session 中的 New Session，建立新的会话。会话设置如下：

Processor：ADSP-21062

Connection Type：Simulator

Platform：ADSP-2106X simulator

7)选择下拉菜单 Project 中的 Build Project 编译链接此工程。编译链接过程中，输出窗中会显示编译链接信息。如果出错，通过双击输出窗口中的出错信息会自动打开出错的源文件。如果编译成功，在输出窗口中也将显示工程编译和加载成功的信息。

8)接下来可以设置断点，并调试程序。

9)利用绘图(Plot)来观察 DFT 输入信号波形，如图 5-55 所示。

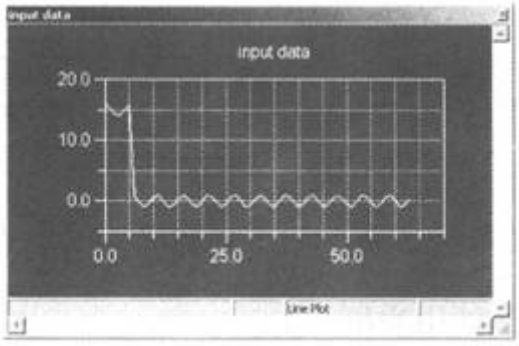

图 5-55 DFT 输入信号波形

输人数据画图参数：

Type：Line Plot

Title：Input data

Memory：Data(DM)Memory

Address：Input

Offset：0

Count：64

Stride：1

Data：Float

10)利用绘图(Plot)来观察 DFT 运算后的输出信号，如图 5-56 所示，使用 Plot 功能可以将 DFT 运算结果的实部和虚部同时在一个图形上显示。

数据画图参数：

Type：Line Plot

Title：Real and Imag Result

Memory：Data(DM)Memory

Address：分别输入 real 和 imag

Offset：0

Count：64

Stride：1

Data：Float

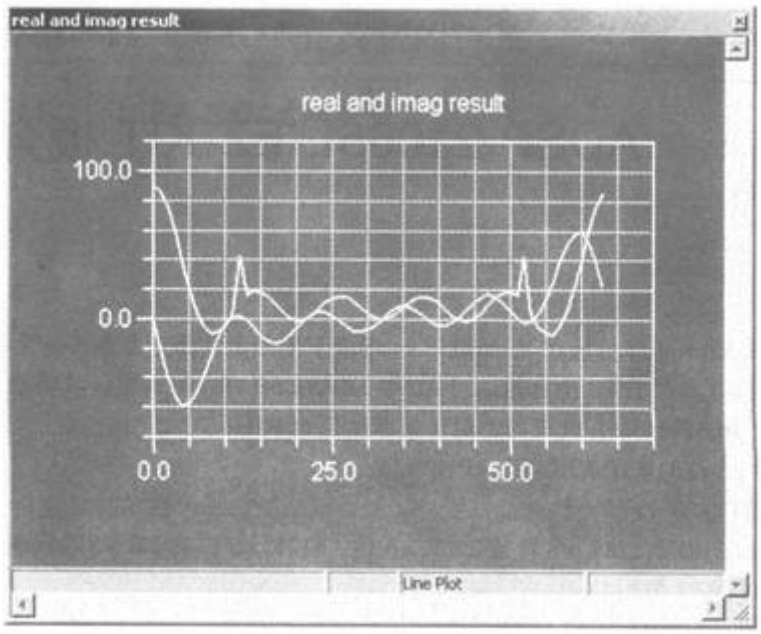

图 5-56　DFT 运算后的输出信号