TMS320C54XDSP
实验指导书

曾 浩　韩 亮　　黄扬帆　编

朱冰莲　审

重庆大学通信工程学院

目　　录

第一章　概　述

1.1　引言

数字信号处理（Digital Signal Processing，简称 DSP）是一门涉及许多学科而又广泛应用于许多领域的新兴学科。它利用计算机或专用处理设备，以数字形式对信号进行采集、变换、滤波、估值、增强、压缩、识别等处理，以得到符合人们需要的信号形式。

数字信号处理是围绕着数字信号处理的理论、实现和应用等几个方面发展起来的。而数字信号处理的实现则是理论和应用之间的桥梁。数字信号处理是以众多学科为理论基础的，它所涉及的范围极其广泛。例如，在数学领域，微积分、概率统计、随机过程、数值分析等都是数字信号处理的基本工具，与网络理论、信号与系统、控制论、通信理论、故障诊断等也密切相关。近来新兴的一些学科，如人工智能、模式识别、神经网络等，都与数字信号处理密不可分。可以说，数字信号处理是把许多经典的理论体系作为自己的理论基础，同时又使自己成为一系列新兴学科的理论基础。

数字信号处理的实现方法一般有以下几种：

(1) 在通用的计算机（如 PC 机）上用软件（如 Fortran、C 语言）实现；

(2) 在通用计算机系统中加上专用的加速处理机实现；

(3) 用通用的单片机（如 MCS-51、96 系列等）实现，这种方法可用于一些不太复杂的数字信号处理，如数字控制等；

(4) 用通用的可编程 DSP 芯片实现。与单片机相比，DSP 芯片具有更加适合于数字信号处理的软件和硬件资源，可用于复杂的数字信号处理算法；

(5) 用专用的 DSP 芯片实现。

虽然数字信号处理的理论发展迅速，但在 20 世纪 80 年代以前，由于实现方法的限制，数字信号处理的理论还得不到广泛的应用。直到 20 世纪 70 年代末 80 年代初世界上第一片单片可编程 DSP 芯片的诞生，才将理论研究结果广泛应用到低成本的实际系统中，并且推动了新的理论和应用领域的发展。

1.2　DSP系统

1.2.1　DSP系统构成

图 1.1 所示为一个典型的 DSP 系统。

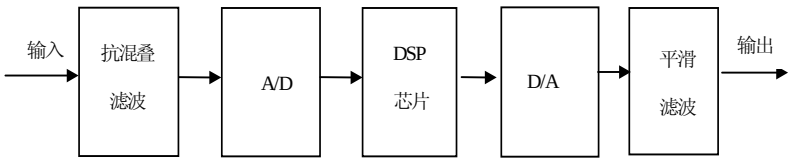

图1.1　典型的DSP系统

必须指出的是，上面给出的 DSP 系统模型是一个典型模型，但并不是所有的 DSP 系统都必须具有模型中的所有部件。

1.2.2　DSP系统的特点

数字信号处理系统是以数字信号处理为基础，因此具有数字处理的全部优点：

(1) **接口方便**。DSP 系统与其他以现代数字技术为基础的系统或设备都是相互兼容的，与这样的系统接口以实现某种功能要比模拟系统与这些系统接口要容易得多；

(2) **编程方便**。DSP 系统中的可编程 DSP 芯片可使设计人员在开发过程中灵活方便地对软件进行修改和升级；

(3) **稳定性好**。DSP 系统以数字处理为基础，受环境温度以及噪声的影响较小，可靠性高；

(4) **精度高**。16 位数字系统可以达到10^{-5}的精度；

(5) **可重复性好**。模拟系统的性能受元器件参数性能变化比较大，而数字系统基本不受影响，因此数字系统便于测试、调试和大规模生产；

(6) **集成方便**。DSP 系统中的数字部件有高度的规范性，便于大规模集成。

DSP 系统其突出的优点已经使之在通信、语音、图像、雷达、生物医学、工业控制、仪器仪表等许多领域得到越来越广泛的应用。

1.2.3　DSP系统的设计过程

总的来说，DSP 系统的设计还没有非常好的正规设计方法。图 1.2 所示是 DSP 系统设计的一般过程。

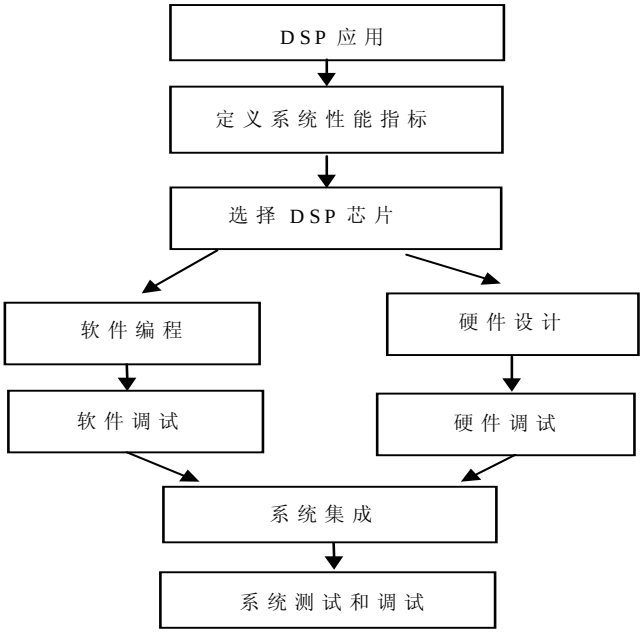

图1.2　DSP系统的设计流程

在设计 DSP 系统之前，首先必须根据应用系统的目标确定系统的性能指标、信号处理的要求，通常可用数据流程图、数学运算序列、正式的符号或自然语言来描述。第二步是根据系统的要求进行高级语言的模拟。在完成第二步之后，接下来就可以设计实时 DSP 系统，实时 DSP 系统的设计包括硬件设计和软件设计两个方面。硬件设计首先要选择合适的 DSP 芯片。然后设计 DSP 芯片的外围电路及其他电路。软件设计和编程主要根据系统要求和所选的 DSP 芯片编写相应的 DSP 汇编程序，若系统运算量不大且有高级语言编译器支持，也可用高级语言（如 C 语言）编程。DSP 硬件和软件设计完成后，就需要进行硬件和软件的调试。软件的调试一般借助于 DSP 开发工具。调试 DSP 算法时一般采用比较实时结果与模拟结果的方法，如果实时程序和模拟程序的输入相同，则两者的输出应该一致。硬件调试一般采用硬件仿真器进行调试。系统的软件和硬件分别调试完成后，就可以将软件脱离开发系统而直接在应用系统上运行。

DSP 系统的开发，特别是软件开发是一个需要反复进行的过程，必须进行反复的修改，以达到最佳性能。

1.3　可编程DSP芯片

1.3.1　什么是DSP芯片

DSP 芯片，也称数字信号处理器，是一种特别适合于进行数字信号处理运算的微处理器，其主要应用是实时快速地实现各种数字信号处理算法。根据数字信号处理的要求，DSP 芯片一般具有如下主要特点：

(1) 在一个指令周期内可完成一次乘法和一次加法；
(2) 程序和数据空间分开，可以同时访问指令和数据；
(3) 片内具有快速 RAM，通常可通过独立的数据总线在两块中同时访问；
(4) 具有低开销或无开销循环及跳转的硬件支持；
(5) 快速的中断处理和硬件 I/O 支持；
(6) 具有在单周期内操作的多个硬件地址产生器；
(7) 可以并行执行多个操作；
(8) 支持流水线操作，使取指、译码和执行等操作可以重叠执行。

当然，与通用微处理器相比，DSP 芯片的其他通用功能相对较弱些。

1.3.2 DSP芯片的发展

世界上第一个单片 DSP 芯片应当是 1978 年 AMI 公司发布的 S2811，1979 年美国 Intel 公司发布的商用可编程器件 2920 是 DSP 芯片的一个主要里程碑。这两种芯片内部都没有现代 DSP 芯片所必须有的单周期乘法器。1980 年，日本 NEC 公司推出的 μPD7720 是第一个具有乘法器的商用 DSP 芯片。

在这之后，最成功的 DSP 芯片当数美国德州仪器公司（Texas Instruments，简称 TI）的一系列产品。TI 公司在 1982 年成功推出其第一代 DSP 芯片 TMS32010 及其系列产品 TMS32011、TMS320C10/C14/C15/C16/C17 等，之后相继推出了第二代 DSP 芯片 TMS32020、TMS320C25/C26/C28，第三代 DSP 芯片 TMS320C30/C31/C32，第四代 DSP 芯片 TMS320C40/C44，第五代 DSP 芯片 TMS320C5X/C54X。

第一个采用 CMOS 工艺生产浮点 DSP 芯片的是日本的 Hitachi 公司，它于 1982 年推出了浮点 DSP 芯片。与其他公司相比，Motorola 公司在推出 DSP 芯片方面相对较晚。1986 年，该公司推出了定点处理器 MC56001。1990 年，推出了与 IEEE 浮点格式兼容的浮点 DSP 芯片 MC96002。美国模拟器件公司（Analog Devices，简称 AD）在 DSP 芯片市场上也占有一定的份额，相继推出了一系列具有自己特点的 DSP 芯片。

表 1.1 是 TI 公司 DSP 芯片 1982 年、1992 年、1999 年的比较表。

表1.1 TI DSP 芯片发展比较表（典型值）

年份	1982 年	1992 年	1999 年
制造工艺	4μm NMOS	0.8μm CMOS	0.3μm CMOS
MIPS	5MIPS	40MIPS	100MIPS
MHz	20MHz	80MHz	100MHz
内部 RAM	144 字	1K 字	32K 字
内部 ROM	1.5K 字	4K 字	16K 字
价格	$150.00	$ 15.00	$5.00~$25.00
功耗	250mW/MIPS	12.5mW/MIPS	0.45mW/MIPS
集成晶体管数	50K	500K	

1.3.3 DSP芯片的分类

DSP 芯片可以按照下列三种方式进行分类。

1．按基础特性分

这是根据 DSP 芯片的工作时钟和指令类型来分类的。如果在某时钟频率范围内的任何时钟频率上，DSP 芯片都能正常工作，除计算速度有变化外，没有性能的下降，这类 DSP 芯片一般称为静态 DSP 芯片。

如果有两种或两种以上的 DSP 芯片，它们的指令集和相应的机器代码机管脚结构相互兼容，则这类 DSP 芯片称为一致性 DSP 芯片。

2．按数据格式分

这是根据 DSP 芯片工作的数据格式来分类的。数据以定点格式工作的 DSP 芯片称为定点 DSP 芯片，以浮点格式工作的称为浮点 DSP 芯片。

3．按用途分

按照 DSP 的用途来分，可分为通用型 DSP 芯片和专用型 DSP 芯片。本书主要讨论通用型 DSP 芯片。

1.3.4 DSP芯片的选择

设计 DSP 应用系统，选择 DSP 芯片是非常重要的一个环节。只有选定了 DSP 芯片，才能进一步设计其外围电路及系统的其他电路。总的来说，DSP 芯片的选择应根据实际的应用系统需要而确定。不同的 DSP 应用系统由于应用场合、应用目的等不尽相同，对 DSP 芯片的选择也是不同的。一般来说，选择 DSP 芯片时应考虑到如下诸多因素。

1．DSP 芯片的运算速度。DSP 芯片的运算速度可以用以下几种性能指标来衡量：

(1) 指令周期；

(2) MAC 时间；

(3) FFT 执行时间；

(4) MIPS；

(5) MOPS；

(6) MFLOPS；

(7) BOPS；

2．DSP 芯片的价格。

3．DSP 芯片的硬件资源。

4．DSP 芯片的运算精度。

5．DSP 芯片的开发工具。

6．DSP 芯片的功耗。

7．其他。除了上述因素外，选择 DSP 芯片还应考虑到封装的形式、质量标准、供货情况、生命周期等。

1.3.5 DSP芯片的应用

自从 20 世纪 70 年代末 80 年代初 DSP 芯片诞生以来，DSP 芯片得到了飞速的发展。DSP 芯片的高速发展，一方面得益于集成电路技术的发展，另一方面也得益于巨大的市场。在近 20 年时间里，DSP 芯片已经在信号处理、通信、雷达等许多领域得到广泛的应用。目前，DSP 芯片的价格越来越低，性能价格比日益提高，具有巨大的应用潜力。DSP 芯片的应用主要有：

(1) 信号处理——如数字滤波、自适应滤波、快速傅立叶变换、相关运算、谱分析、卷积、模式匹配、加窗、波形产生等；

(2) 通信——如调制解调器、自适应均衡、数据加密、数据压缩、回波抵消、多路复用、传真、扩频通信、纠错编码、可视电话等；

(3) 语音——如语音编码、语音合成、语音识别、语音增强、说话人辨认、说话人确认、语音邮件、语音存储等；

(4) 图形/图像——如二维和三维图形处理、图像压缩与传输、图像增强、动画、机器人视觉等；

(5) 军事——如保密通信、雷达处理、声纳处理、导航、导弹制导等；

(6) 仪器仪表——如频谱分析、函数发生、锁相环、地震处理等；

(7) 自动控制——如引擎控制、声控、自动驾驶、机器人控制、磁盘控制等；

(8) 医疗——如助听、超声设备、诊断工具、病人监护等；

(9) 家用电器——如高保真音响、音乐合成、音调控制、玩具与游戏、数字电话/电视等。

随着 DSP 芯片性能价格比的不断提高，可以预见 DSP 芯片将会在更多的领域内得到更为广泛的应用。

第五章　DSP 实验

5.1　实验一　　CCS基本使用

一、预习内容

复习汇编语言指令内容，汇编语言使用中的基本概念，CCS 的基本概念。

二、实验目的

1. 掌握一个 DSP 软件开发流程。
2. 理解 DSP 的启动过程。
3. 掌握汇编的书写规范，学会汇编指令的运用。
4. 掌握编译器和连接器的使用，能够合理地分配存储空间。
5. 学习 CCS 的各种调试技巧，如：CUP 寄存器、数据和程序存储器的观察，断点的设置，反汇编窗口的使用。
6. 学习定点数的运算方法。

三、实验要求

1.用.set 定义四个立即数。

2.在.bss 段建立几个存储空间。

3.把立即数相加和相乘，结果放在.bss 分配的存储空间。

4.合理分配各个段的存储地址，并使用 CCS 观察。

四、实验原理

1. 软件开发流程

从大的步骤来讲，一个软件要能够在 DSP 上面正常运行，需要用户完成以图 5.1.1 所示的流程。

第一步：在用户的工程里面，必须包含至少两个文件。一个是程序指令的源文件，这个文件可以是汇编编写的，也可以是 C 语言编写的。当然，这样的文件在一个工程当中可以用多个，而且，编程的语言可以不统一。工程当中必须包含的第二个文件是连接文件。这个文件的作用是把源文件中定义的各个段放到用户指定的 DSP 的程序存储区，从而可以在该区域按一定顺序执行。

第二步：两个文件编辑完成以后，可以调用编译、汇编和连接命令。编译功能把 C 源文件转化为汇编文件，期间的转化有一定的规则，这些规则是在书写 C 语言语句是应该注意的东西。汇编的功能是把编译形成的汇编文件或者是用户自己编辑的汇编文件转化为 COFF 格式文件。而最后通过连接命令，结合连接文件，把 COFF 文件连接成为可执行的 COFF 文件，即后缀名为.OUT 的文件。

第三步：通过 CCS 的加载功能，把.OUT 文件加载到 DSP 内部，其地址由连接文件规定。如果成功，可以在 CCS 中运行该文件，并相应进行调试。

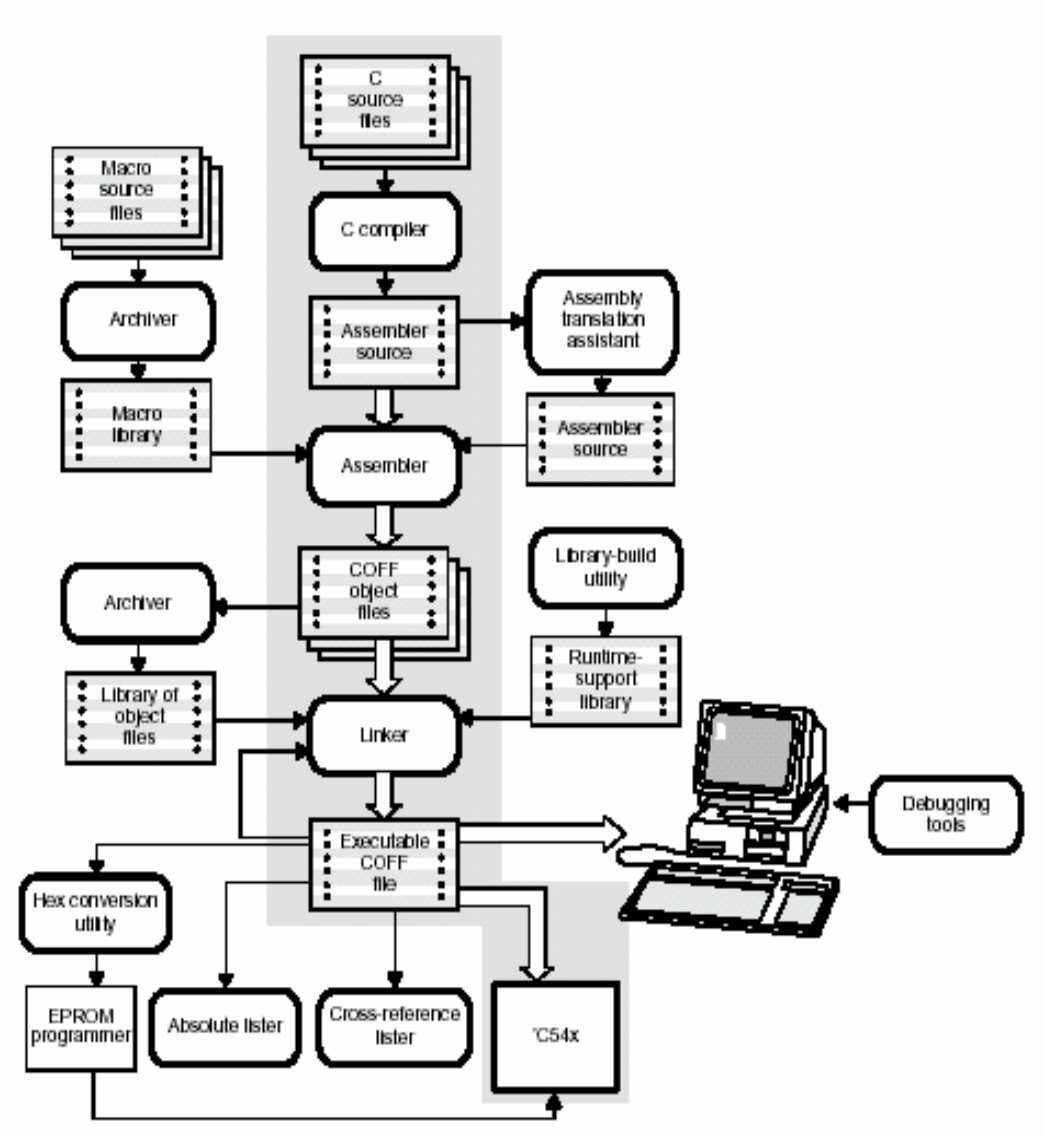

图 5.1.1　　开发流程图

2.汇编源文件书写

在 CCS 中，新建一个文件，在这个文件中，如果使用汇编语言，那就命名时使用 ASM 的后缀名，如果使用 C 语言，那命名时就使用 C 的后缀名。同时应该把文件加载到相应的工程当中去。

对于使用汇编的源文件，他有一定的语法规范。包括语法格式，标号的表示，立即数的表示，段的定义和宏定义。《TMS320C54x Assembly Language Tools User’s Guide》和《TMS320C54x DSP Reference Set---Volume 2: Mnemonic Instruction Set》是掌握汇编语言书写方法的两本手册。

3.启动方式

DSP 有多种启动方式。但是，由于我们采用通过 CCS 从 JTAG 直接加载的方式，对于其他的加载过程可以不必考虑。但是，MP/MC 方式，对于启动却有一定影响。这里，有两个问题要注意。一是系统复位后，MP/MC 寄存器的值究竟是多少。二是 MP、MC 方式对应的 DSP 片内存储器分配关系如何。

通过硬件跳线，可以设置启动方式，通过连接文件，可以把程序代码放到指定的位置。

4. CCS 的调试工具

在本次实验中，CCS 的调试工具将要使用到一些基本的操作。主要包括如何观察 CPU 的各个寄存器，如何观察数据区和程序区的数据；单步执行的方法和断点执行。

五、实验步骤

1. 双击桌面的 CCS5000 图标，进入 CCS。如果遇到问题，检查硬件线路是否连接正确，电源是否打开，CCS SETUP 是否合理。当然，也有可能是 PC 机内的 ISA 插卡接触不好。
2. 新建一个工程。执行 Project 菜单下的 New，选择一个合适的位置，存放这个新建的工程，最好是自己建立的目录，便于管理。工程的名称以字母开头，其他没有要求，不需要加后缀名，如：sy1。
3. 新建一个文件作为汇编源文件。执行 File 菜单下的 New/Sourse File 命令，接着就可以在编辑框里，按照汇编语言的规范，编辑相应的指令代码。指令完成以后，执行 File/Save as 命令，给汇编文件取一个合适的文件名，保存在同工程相同的目录下。保存时，不需要加后缀名，但是，要选择文件的类型为 Assembly Source File(.asm)。
4. 编辑汇编源文件，完成以后，需要把这个文件加入到相应的工程中去。在左边的工程管理窗口里，鼠标移到工程文件名上面，单击右键，选择 Add File。寻找到刚才编辑的汇编源文件，选择确认。之后可以在工程管理器中观察到，Source 文件夹中，出现了刚才的汇编文件。
5. 汇编文件的代码如下：

```
.global _c_int00

VAL1      .set      012h          ; 18
VAL2      .set      034h          ; 52
     .bss temp,1                  ; address of 0x080
     .bss temp1,1                 ; address of 0x081
     .bss temp2,1                 ; address of 0x082
;result register
     .bss add_result,1            ; address of 0x088
     .bss mpy_i_h,1               ; address of 0x08a
     .bss mpy_i_l,1               ; address of 0x08b
```

```
        .text
_c_int00:
        ld   #temp,DP        ; load DP of temp1

        st   #VAL1,temp1
        st   #VAL2,temp2     ; init temp1 & temp2 ,18+52=70(0x46)
;-------- test ADD ---------------------------
        ld   temp1,a         ; load temp1 -> a
        add  temp2,a         ; a+temp2 -> a
        stl  a,add_result    ; save a(low 16 bits) -> add_result
        nop                  ; set breakpoint

        st   #VAL1,temp1
        st   #VAL2,temp2     ; init temp1 & temp2,18*52=936(0x3a8)
;-------- test MPY (integer) ------------------
        rsbx FRCT            ; prepare for integer mpy
        ld   temp1,T         ; temp1   -> T
        mpy  temp2,a         ; temp1*temp2 -> A (result is 32 bit)
        sth  a,mpy_i_h       ; the high 16bit in mpy_i_h
        stl  a,mpy_i_l       ; the low 16bit in mpy_i_l
        nop                  ; set breakpoint

end:
        b    end
        .end
```

使用.set 定义立即数的值；使用.bss 分配一定存储空间；在.text 段编写指令，主要有：ld、st、add、mpy、stl、sth 等等。

6. 根据 MP/MC 的值，理解 DSP 存储器地址分配，编辑好连接文件。连接文件的编辑方法同汇编源文件是一样的，也是建立、编辑、存盘、命名、添加入工程。不同在于文件的内容和文件的后缀名不同。连接文件的后缀名文.cmd，选择存盘类型的时候要注意。文件的内容如下：

```
MEMORY
{
   PAGE 0:
            IPROG:      origin = 0x2000,        len = 0x1000
   PAGE 1:
           IDATA:      origin = 0x80,           len = 0x1000
}
```

```
SECTIONS
{
   .text:    {} > IPROG PAGE 0
   .bss:     {} > IDATA PAGE 1
  }
```

7. 连接文件本来可以设置许多的参数，但是，这些参数也可以在 Project/Options 中设置。在这个窗口中，可以对编译、汇编和连接的过程设置参数，各个参数的意思查阅相关的手册。通常，使用默认参数就可以了。参数设置完成后，就可以执行 Debug/build 命令。这个命令将依次执行编译、汇编、连接的三个过程，任何一个过程中出现错误，都会在荧幕下方的窗口中显示出来。对于错误，要认真阅读提示信息，从而知道错误出现的位置，并进行相应的改正。
8. 如果 Build 的过程没有错，将会在工程所在目下出现一个后缀名为.out 的文件，这个文件文件名是在 Project/Options 中设置产生的。目标板采用 MC 方式，load 刚生成的.out 文件，如果提示错误，检查跳线设置。
9. 加载成功，光标出现在程序起始位置，呈黄色。这时，在需要设置断点的地方设置断点。打开 View 里面的 CPU Register,在新出现的窗口里，可以观察 DSP CPU 寄存器的各个数据。同样，可以打开 View/Memery，选择.data 代码段所在的位置，开始地址为 0X80，位于 data 页，在新窗口中，可以观察该地址开始的数据存储区的各个数据。
10. 执行 Debug/Run 命令，程序开始运行。到断点处停止，加法的运算完成。这时，可以再次观察 CPU Register 窗口，红色数字代表数据有变换，注意 PC 指针的改变。数据窗口中，则注意用于存放运算结果的地址单元的值的变化。
11. 继续执行 Run 命令，观察乘法的运算结果。
12. 执行 Debug 中 Reset Dsp 和 Restart 命令，光标从新回到程序入口。执行 Debug/StepInto，单步运行程序，再次观察结果。

六、实验报告要求

给出数据空间的定义数据大小和存储位置，记录各个计算结果变化前后的内容，给出连接文件和汇编程序。