

**DSP应用实验报告**

实验九：DSP开发基础

院 系：电子工程与光电技术学院

专 业：电子信息工程

姓 名：郭志航

学 号：9181040G0422

指导老师：李彧晟

2021年4月12日

目录

[9.1 实验目的 3](#_Toc69713960)

[9.2 实验仪器 3](#_Toc69713961)

[9.3 实验内容 3](#_Toc69713962)

[9.4 实验步骤 3](#_Toc69713963)

[9.5 实验总结 10](#_Toc69713964)

实验九：DSP开发基础

# 9.1 实验目的

1.了解 DSP 开发系统的基础配置

2.熟悉 DSP 集成开发环境（CCS）

3.掌握 C 语言开发的基本流程

4.熟悉代码调试的基本方法

# 9.2 实验仪器

计算机，TMS320F28335 DSP教学实验箱，XDS510 USB仿真器

# 9.3 实验内容

建立工程，对工程进行编译、链接，载入可执行程序，在 DSP 硬件平台上进行实时调试，利用代码调试工具，查看程序运行结果。

# 9.4 实验步骤

**1.将TMS320F28335教学实验箱连接至计算机。**

**2.点击桌面CCS5快捷方式，启动 CCS 集成开发环境。**

**3.根据实验讲义，熟悉 CCS 集成开发环境各项操作**，包括新建工程、添加工程文件、查阅代码、建立工程、调试程序、程序运行、程序调试等步骤。

**4.项目编译、链接、调试：**

1)将“LAB\_9”工程文件添加至目录。导入示范文件后，原工程调试后出现了报错现象。错误信息为：program will not fit into available memory，经查阅相关资料，原因认为是超过了cmd中MEMORY 里定义的地址长度从而导致的内存分配不足，并进行了修改：RAML1的length大小改为 3000。这样修改后不再报错，但是仍有警告出现。资料上说警告的原因是因为在 SECTIONS中缺少定义，在其中加上相应定义即可。但由于对本次实验无影响，所以就没有进行此处修改。具体 cmd 文件修改操作如图1所示：

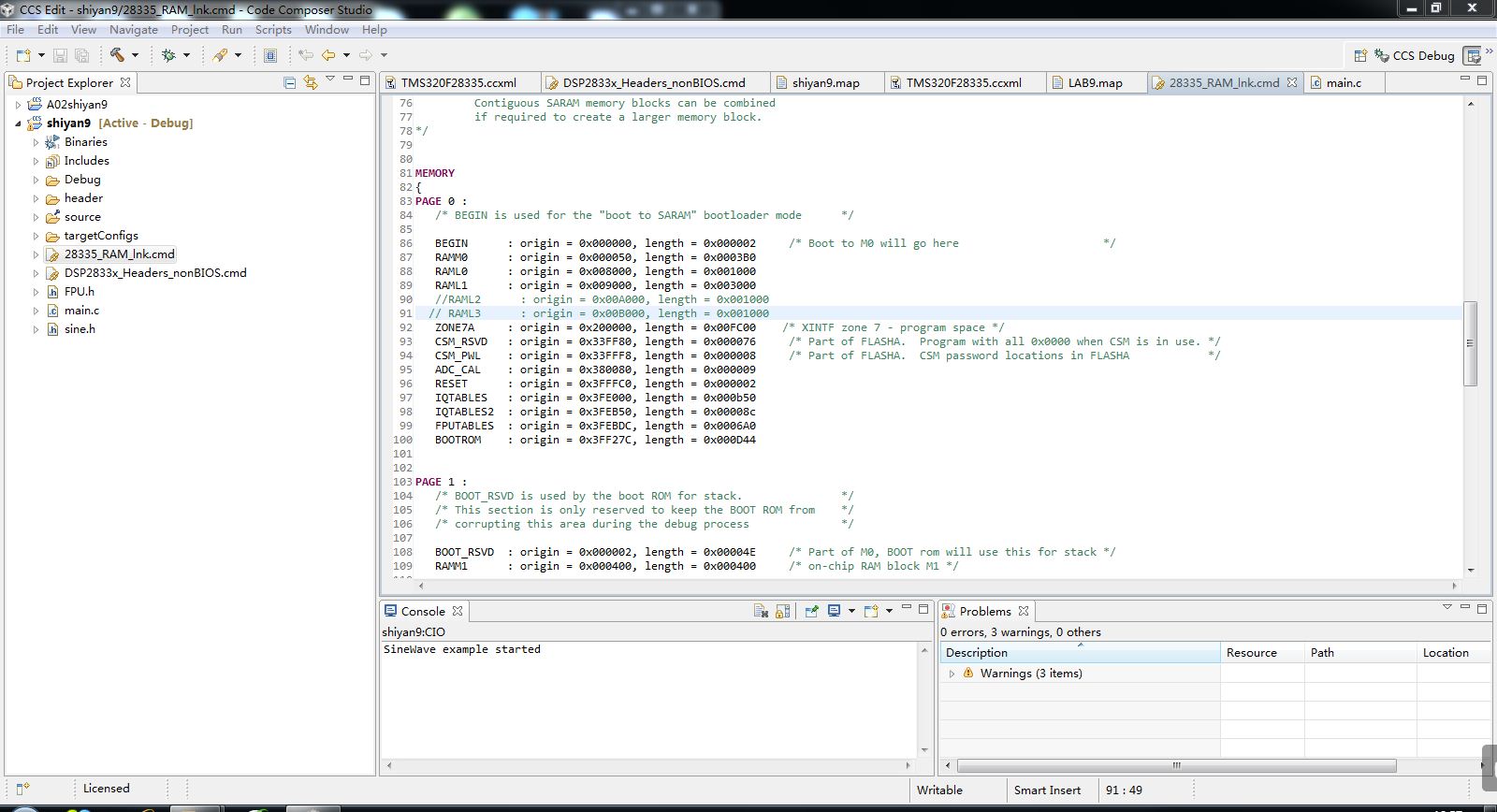


图 1 cmd文件修改

2）在“LAB\_9”工程中双击“TMS320F28335.ccxml”，在弹出的“Basic”界面中“connection” 选项中选择“SEED XDS510PLUS Emulator”，在“Board or Device”选项选择“TMS320F28335”后，点击右侧“Save Configuration”下的“Save”保存设置。

3)打开实验箱电源，在主菜单下选择“Run→Debug”，若仿真器正确连接后，进入“CCS Debug”界面。在CCS Debug环境界面的主菜单中选择“Run→Resume”运行程序。程序的执行结果依赖外部硬件或查看寄存器存储器的数值加以验证。

**5.添加结构体变量至观察窗口：**

1)点击Add new expression，添加变量currentBuffer到变量观察窗口

2)选中变量currentBuffer，点击左侧箭头，即看到input与output相关内容，可观察变量的类型、内容和地址。同样的操作，将dataIO( )添加到变量窗口，查看该子程序的入口地址。观察窗口截图如图：

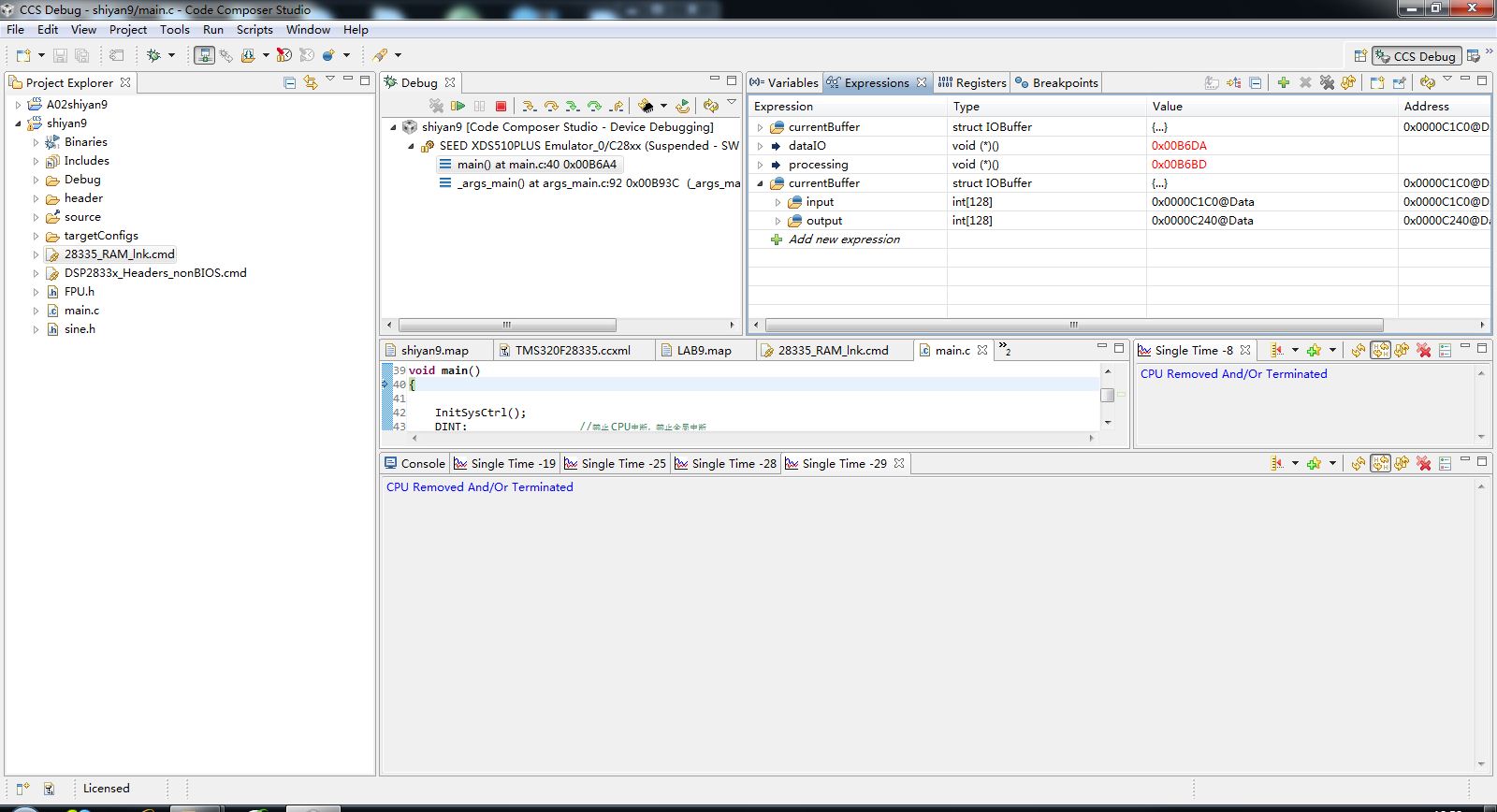


图 2 变量观察窗口

3)通过此操作可以观察到，变量currentBuffer.input所在存储器地址为 0x0000C1C0@Data，变量currentBuffer.output 所在存储器地址为0x0000C240@Data，子程序dataIO()的地址为 0x00B6DA。将两个变量和子程序的地址记录下来。

**6.在dataIO()处设置断点，关联输入文件sine.dat**

鼠标移动到断点所在行，右键选择“Breakpoint Properties”，在“Action”选项中选择“Read Data from file”,在“File”选项中选择工程文件夹中的“sine.dat”文件,勾选“Wrap Around”选项为“true”，起始地址“Start Address”为currentBuffer.input的起始地址（0x0000C1C0@Data），数据长度为 128，点击“OK”。各项设置如图3所示：

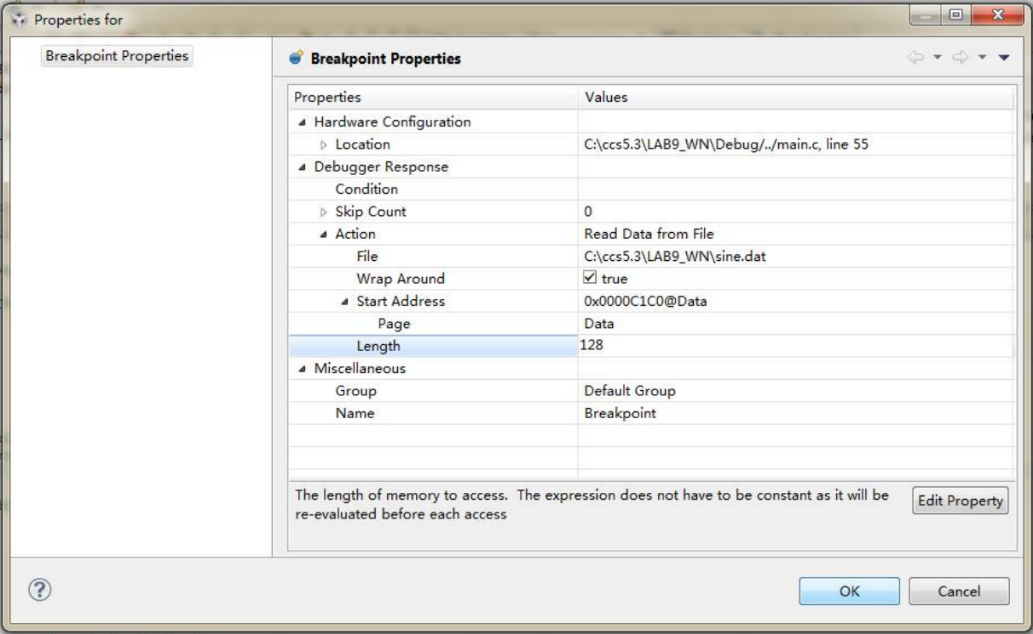


图 3 断点及关联数据设置DSP

**7.图形显示数据空间,打开图形显示功能**

在主菜单的点击“Tools→Graph→single time”，按照实验讲义所示设置各项参数。也可直接在观察窗口中选择要绘图的变量，右键选择“Graph”“sign Graph”即可获得想看的图形。设置起始地址为变量currentBuffer.input的起始地址0x0000C1C0@Data，得到存储空间的时域波形如图4所示

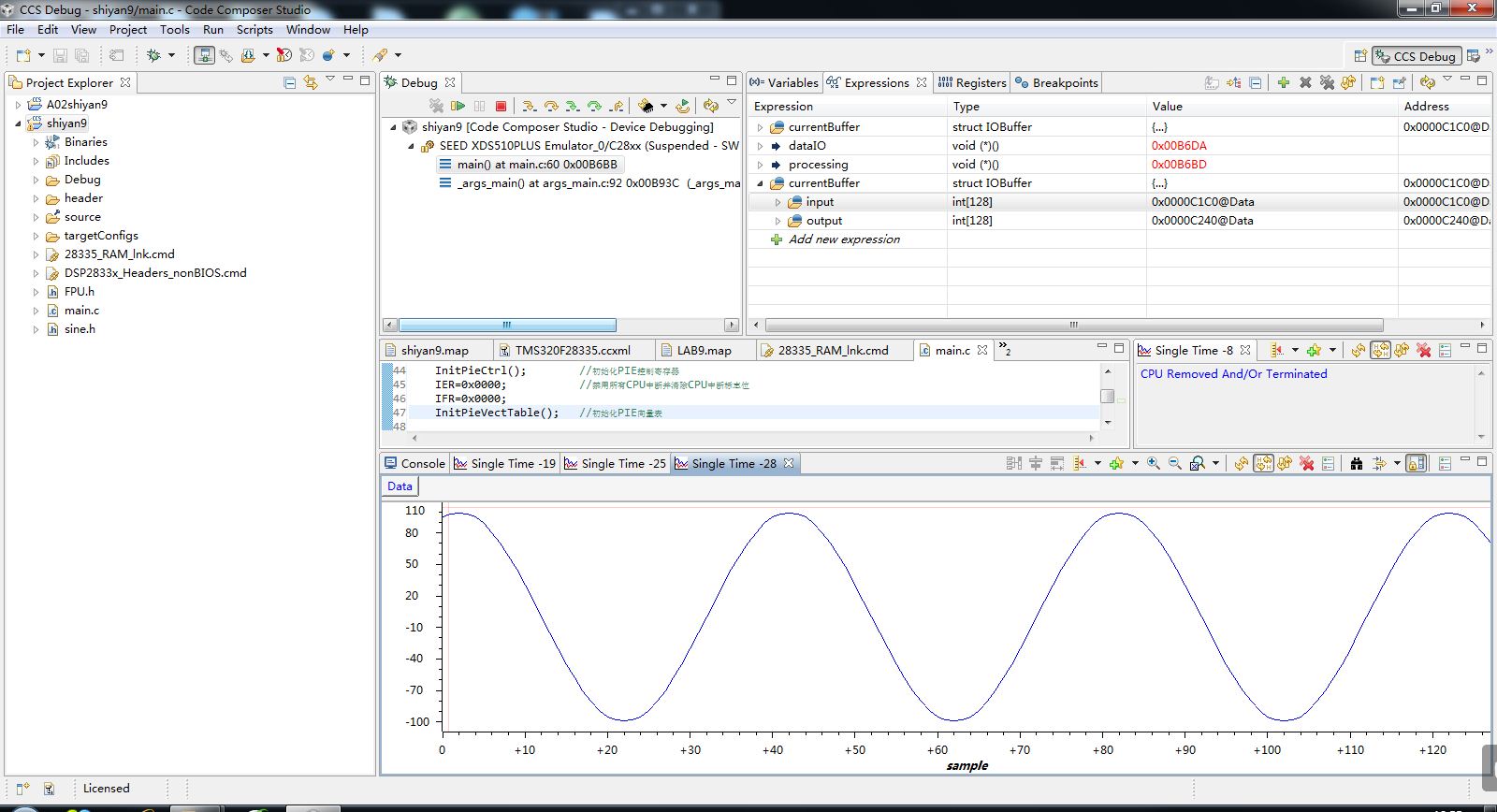


图 4 数据空间currentBuffer.input 缓冲存储器中的波形

设置起始地址为变量currentBuffer.output的起始地址 0x0000C240@Data，得到存储空间的时域波形如图 5 所示

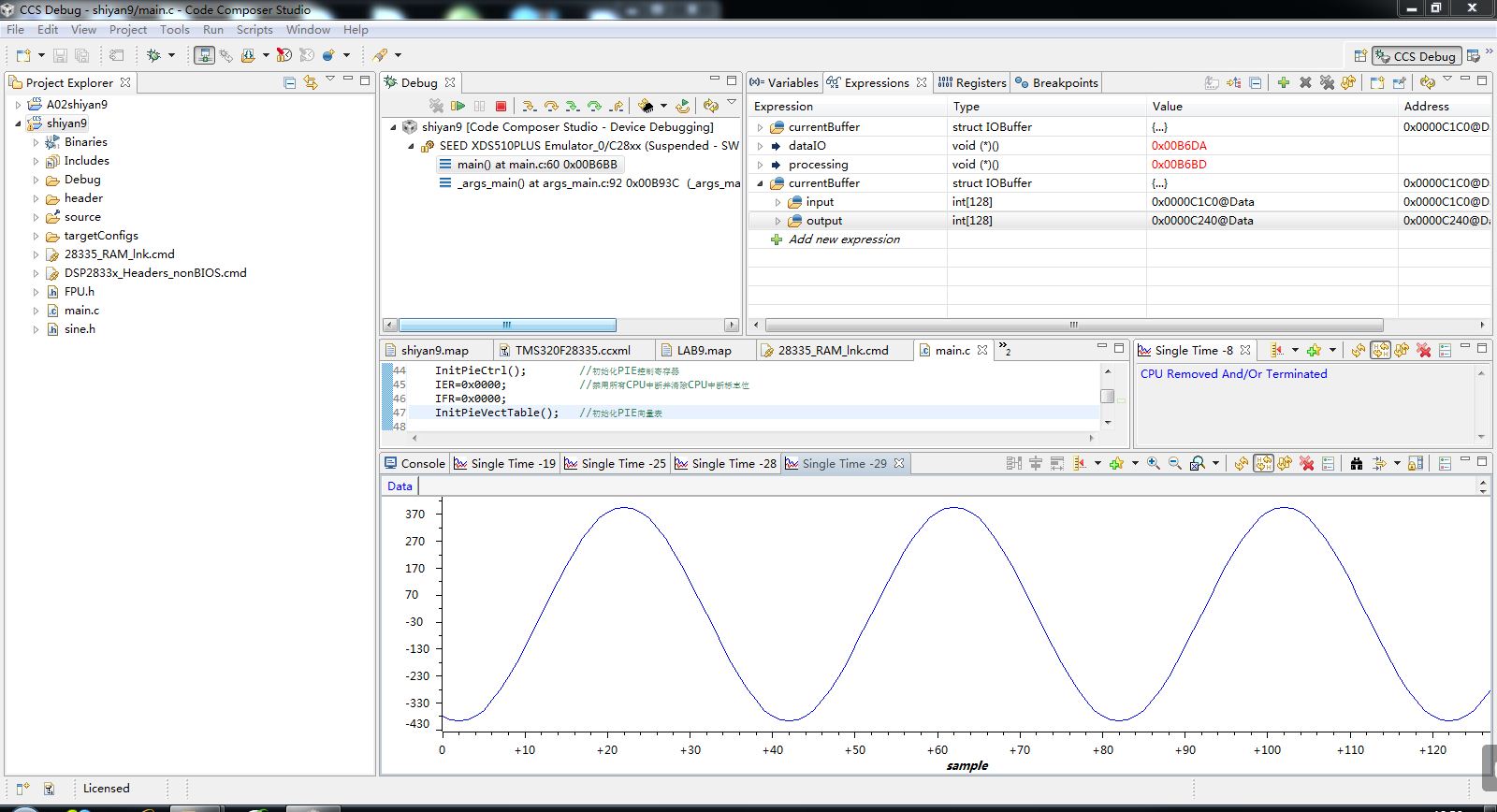
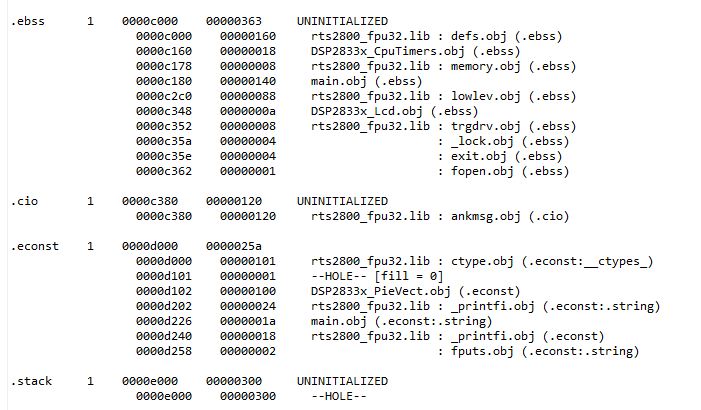


图 5 数据空间currentBuffer.output 缓冲存储器中的波形

**8.打开工程的.map文件，查看所有的段在存储空间的地址、长度和含义，指出分别位于TMS320F28335 的什么存储空间以及物理存储块名称，主程序中所用的变量分别属于什么段**

MAP文件大概分为文件头、内存配置、段映射、全局符号四部分。打开 MAP文件至 section location map 部分，即可知道该部分程序中所有的段实际映射的起始地址与实际长度。如图 6所示。



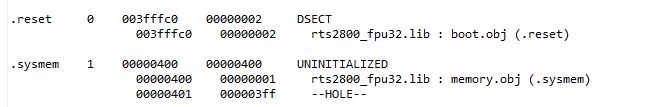
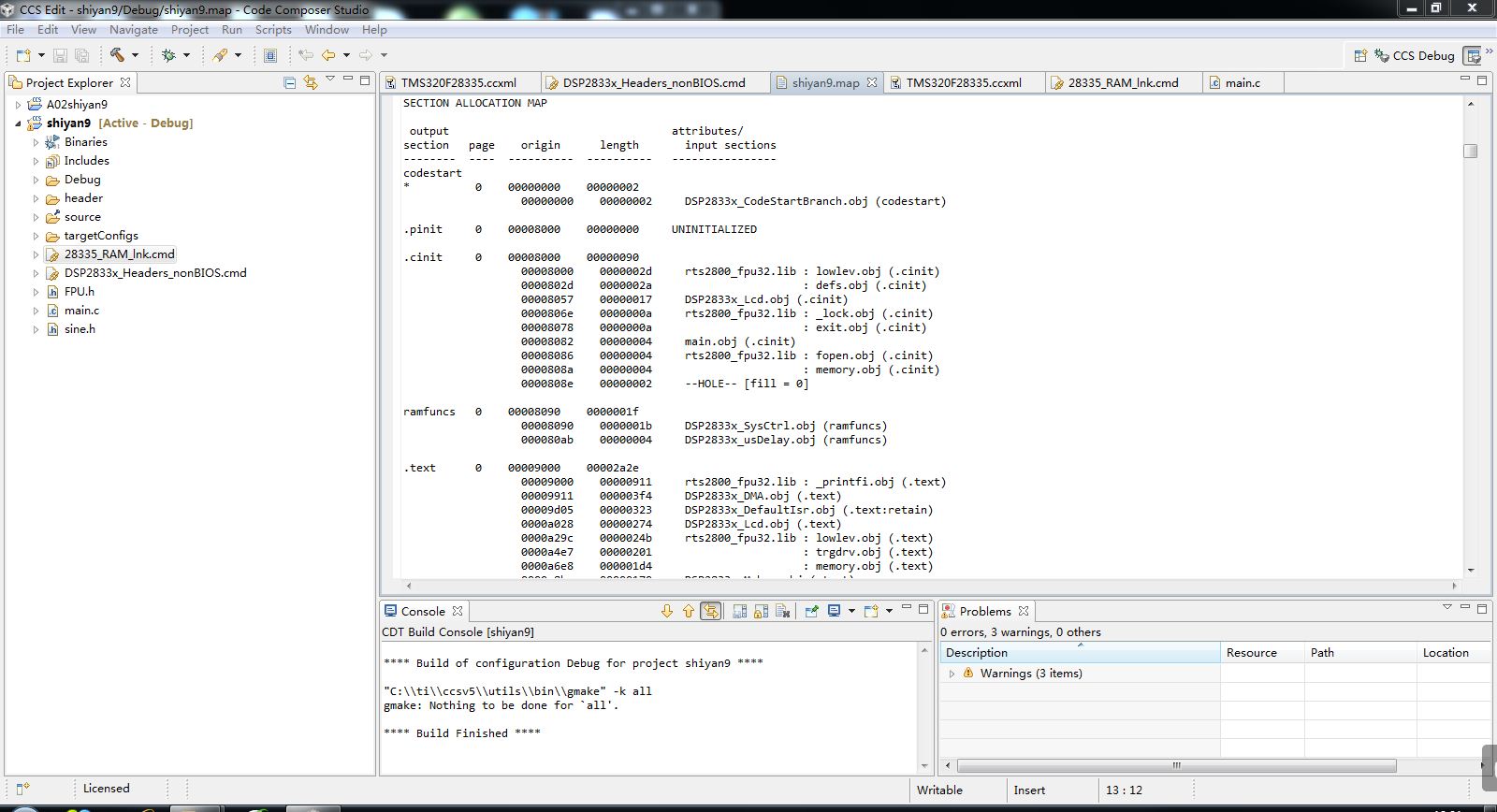


图 6 section location map

联系 cmd 文件中 section 指令即可了解该段所在地址。

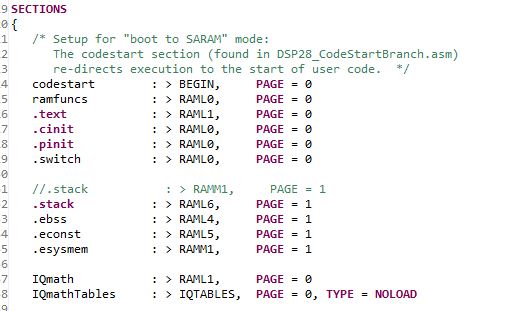


图 7 cmd文件section指令

下面以currentBuffer结构体变量为例，指出该变量所在段和物理存储空间。

法一： 由上文知currentBuffer 变量地址为0x0000C1C0,由MAP文件的 MEMORY CONFIGURATION 段即可得出currentBuffer 变量在RAML4中。如图8所示：

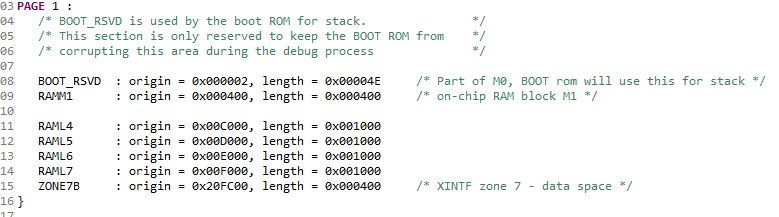


图 8 变量 currentBuffer 所在段和物理储存空间

法二： 阅读 main 函数，易知 currentBuffer 变量为全局变量，即为.ebss 段，查阅cmd文件section（图7），即可得出 currentBuffer 变量在 RAML4 中。

主程序中的变量有全局变量 ebss，局部变量.stack，代码.text，初始值.cinint，所在物理存储空间和存储块如上图所示。

**9.查看.cmd 命令文件，比较其与上述.map 中的映射关系。试图修改.cmd 文件，再次编译链接，查看配置命令与各段的映射关系。**

MAP文件大概分为文件头、内存配置、段映射、全局符号四部分。内存配置与cmd文件中的 MEMORY 指令关联，在cmd文件中定义的程序与数据区间定义，在该部分均可以找到对应，与cmd文件不同的时，在MAP文件中加入了一个实际使用的区间，即在程序中实际用到的空间长度。

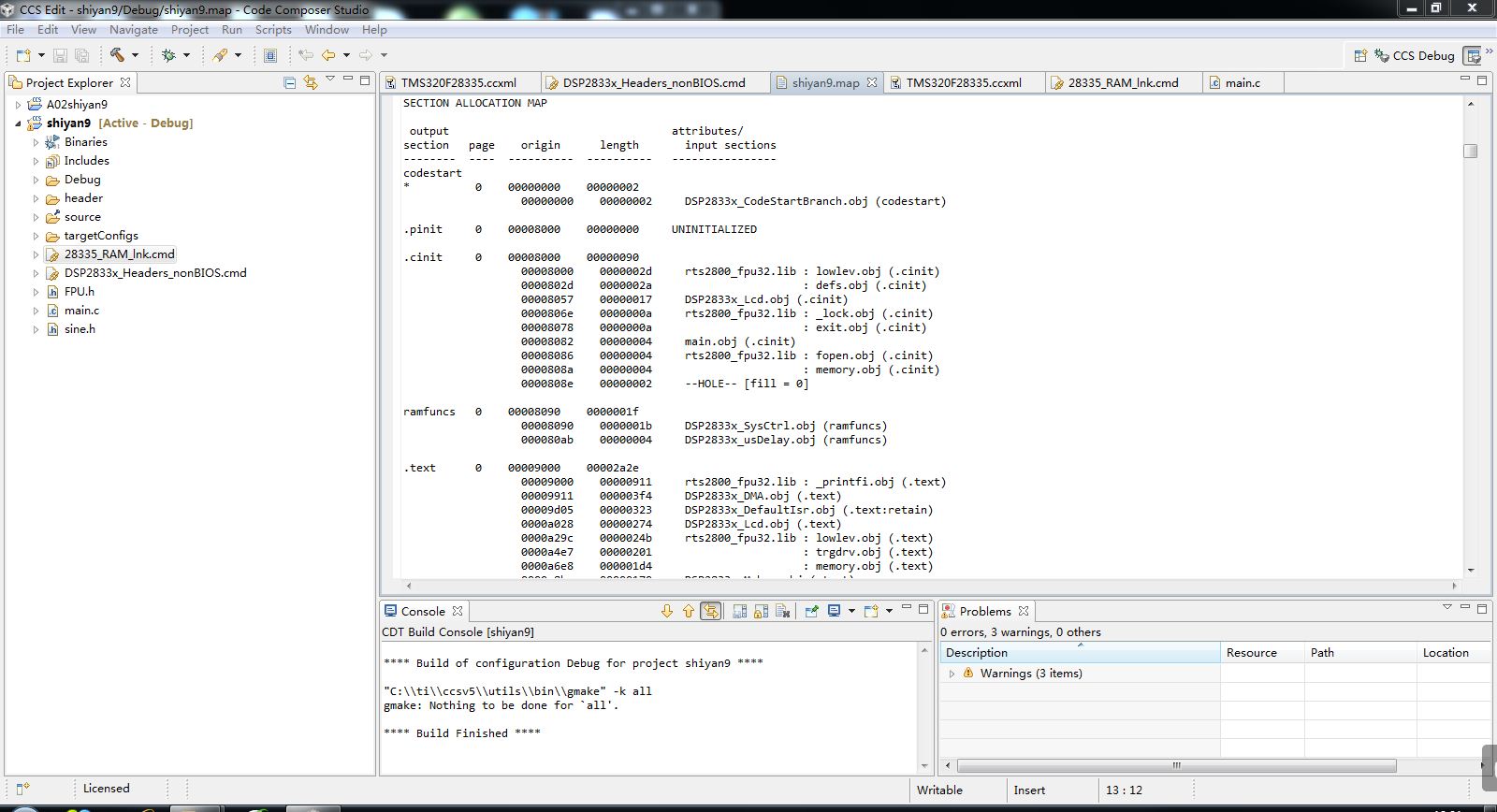


图 9 map文件

段映射部分与cmd文件中的 SECTION 指令关联，在该部分程序中所有的段实际映射的起始地址与实际长度均有详细说明。可以具体到程序中#pragma 指定的段和各个单独文件产生的OBJ文件。修改cmd配置文件，在进行断点求地址操作，发现地址产生变化,映射也发生了改变。例如将RAML1的起始地址由0x009000变化至0x009002，长度也进行相应改变，查阅map文件中.text起始地址也随之发生改变。改变如下表所示：

|  |  |  |
| --- | --- | --- |
|  | 修改前 | 修改后 |
| CMD |  |  |
| MAP |  | mapXIUGAI |

表1 cmd与map对应关系

# 9.5 实验总结

通过本次实验，首次真正接触到了DSP开发板和CCS开发软件。在刚开始进行调试的时候遇到了许多问题，主要几个问题如下：

**1.在实验时添加工程文件，出现了无法导入的情况或导入成功后无法编译的情况。**分析原因，由于Lab9实验工程配置文件起初采用手动输入，经助教提议通过导入范例文件的方式添加工程文件，同时勾选“copy projects into workspace”，此问题便得到了解决。

**2. 在调试代码时使用printf函数以直接输出目标参数，但在调用编译过程中出现如下错误：**

*error #10099-D: program will not fit into available memory.*

经查阅资料，出现上述报错的原因是分配内存不足，超出cmd文件中memory里定义的地址长度。我们对其地址长度进行修改，第一次仅增加1000大小占用了RAML2的内存，但是问题并未解决。考虑到RAML2、RAML3的内存在案例中并没有完全利用，故我们去除了两者，将2000大小均让给RAML1。经过调试发现该方法可行。

**3.在完成添加变量至观察窗口、添加断点和数据关联等步骤后，在绘制数据图形时发现绘制出来的图形是杂乱无序的**，确认 sine.data文件其中数据确是正弦波样点的数据。仔细检查操作步骤后发现，是在debug之后忘记进行Resume 操作，程序没有完全执行；同时我们断点的位置也出现错误，放在了dataIO（）的后面几行代码处。改正之后便得到了正确波形。

**4.DSP文件中源文件的作用**

|  |  |
| --- | --- |
| 文件名 | 作用 |
| include头文件(.h) | 定义程序中的函数、参数、变量和宏单元，配合库函数使用； |
| DSP/BIOS CONFIG FILES | 开发基于DSP/BIOS的程序保存BIOS配置之后自动生成的文件 |
| Source Files | 源程序，实现DSP系统指定功能的主要代码部分 |
| Program.cmd | 链接文件，在源文件通过汇编器（Assembier）输出的.obj files（目标文件）需要通过Linker（链接器）得到输出文件，在链接阶段Linker根据.cmd内部存储区、section分配以及lib来链接 |
| .cmd | 负责定义memory region，将用户的代码数据section映射到区域内 |
| .ccxml文件 | 配置仿真器 |
| .map | 执行文件的映像信息纪录文件 |
| .pdb（Program Database） | 记录程序有关的数据和调试信息 |

表 2 源文件在DSP工程中的应用

通过本次实验，我们熟悉了CCS的操作环境及调试方法，为接下来几次实验奠定了基础。本次实验不涉及程序编写，只是简单的操作过程，因此较为简单。希望在接下来几次实验中能够继续深入学习DSP芯片编程的其他方面。