使用 CCS 进行 DSP 编程（一）

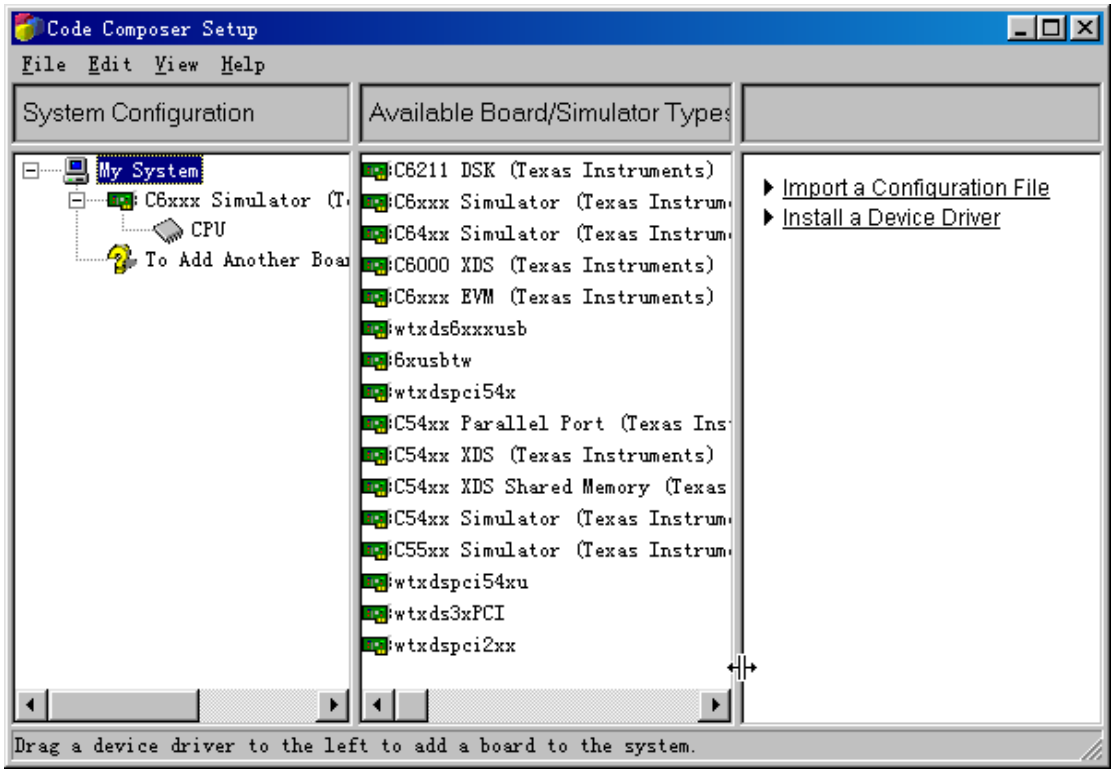
——CCS 编程入门

pacificxu

TI 公司提供了高效的 C 编译器和集成开发环境 Code Composer Studio，学习‘ C6X 的编程应该从学习 CCS 的使用开始。

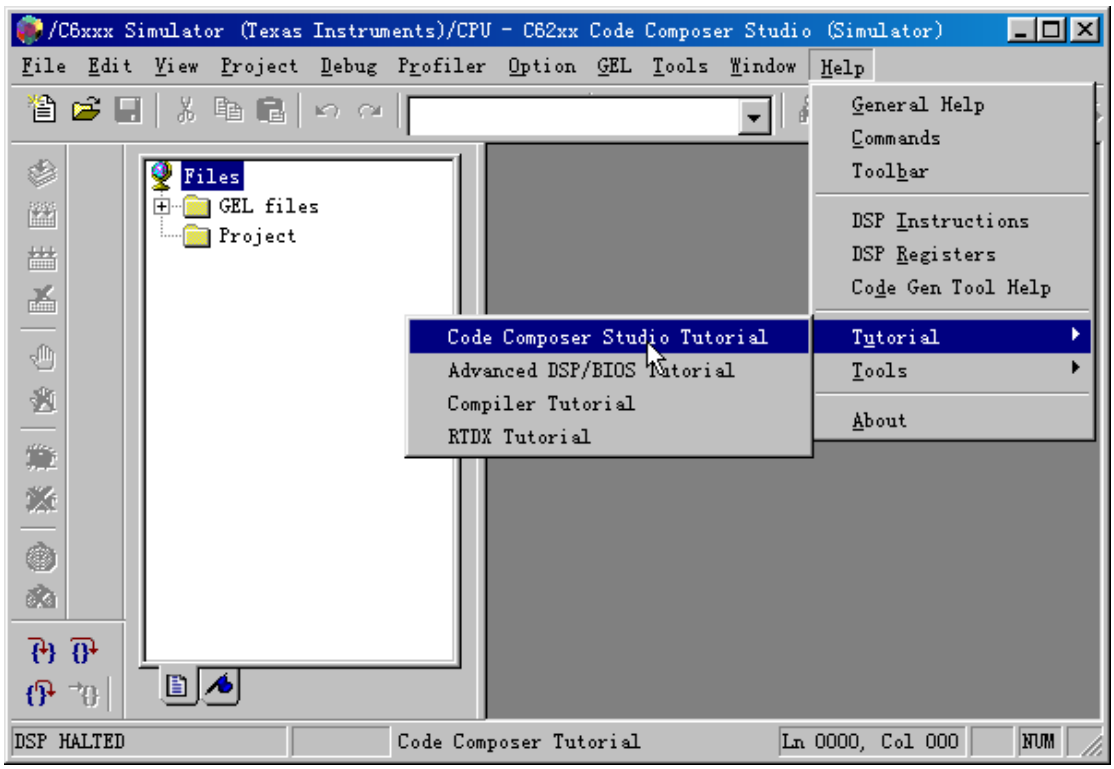
首先安装 CCS，CCS 的安装有详细的说明，并配有简短的 Quick Time 的多媒体介绍，对于没有购买 CCS 的用户，可以从 TI 处得到 30 天的试用版（没有硬件仿真功能）。

使用 CCS 前需要对 CCS 进行设置，以 Simulator 为例，运行 Setup CCS C6000 1.20，安装 Device Driver，对于有硬件支持的仿真器，可以选择配套的 CCS 驱动，设置完成的画面如下图所示：用户的界面大致相同。

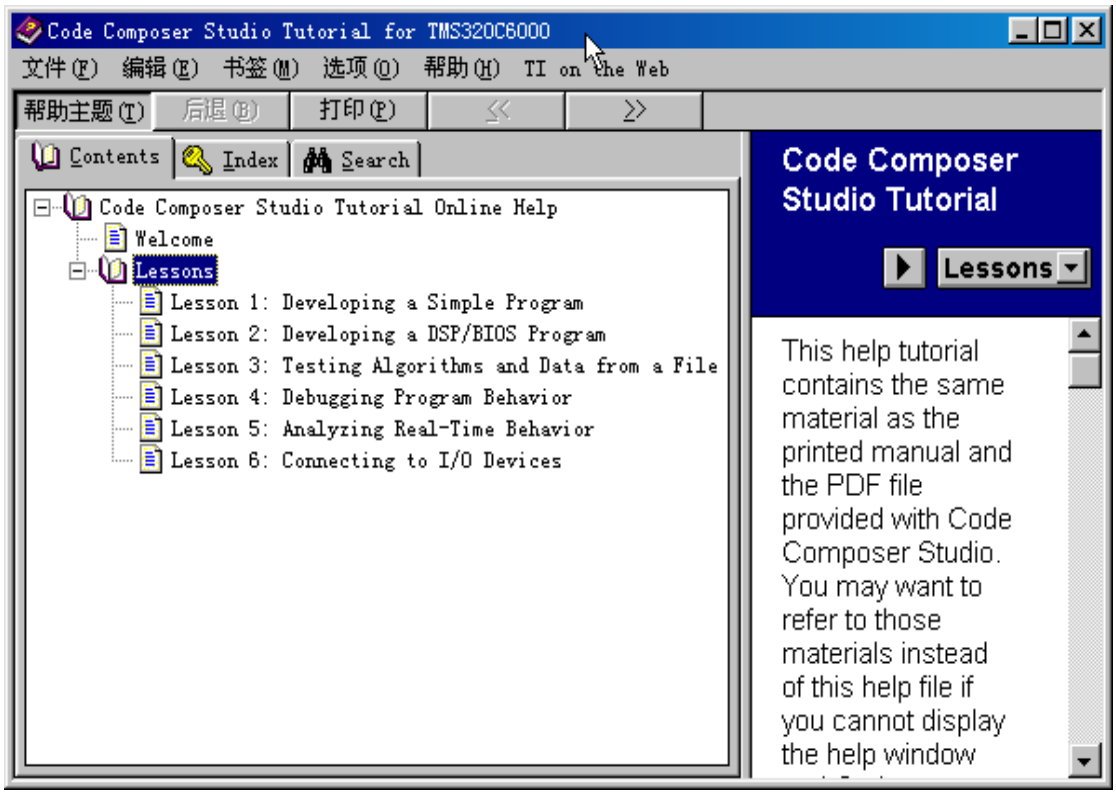


接下来就可以运行 CCS 了，**CCS 提供了比较好的例子，对于初学者，仔细学习这些例子，会起到事半功倍的效果**。在 CCS 的 Help 菜单的 Tutorial 子菜单下，给出了四个教程，分别是：Code Composer Studio Tutorial、Advanced DSP/BIOS Tutorial、Compiler Tutorial 和 RTDX Tutorial，用户可以从简单的 CCS 功能

开始，如创建一个工程文件 Project，到创建一个完善的用户程序一步一步的进行。



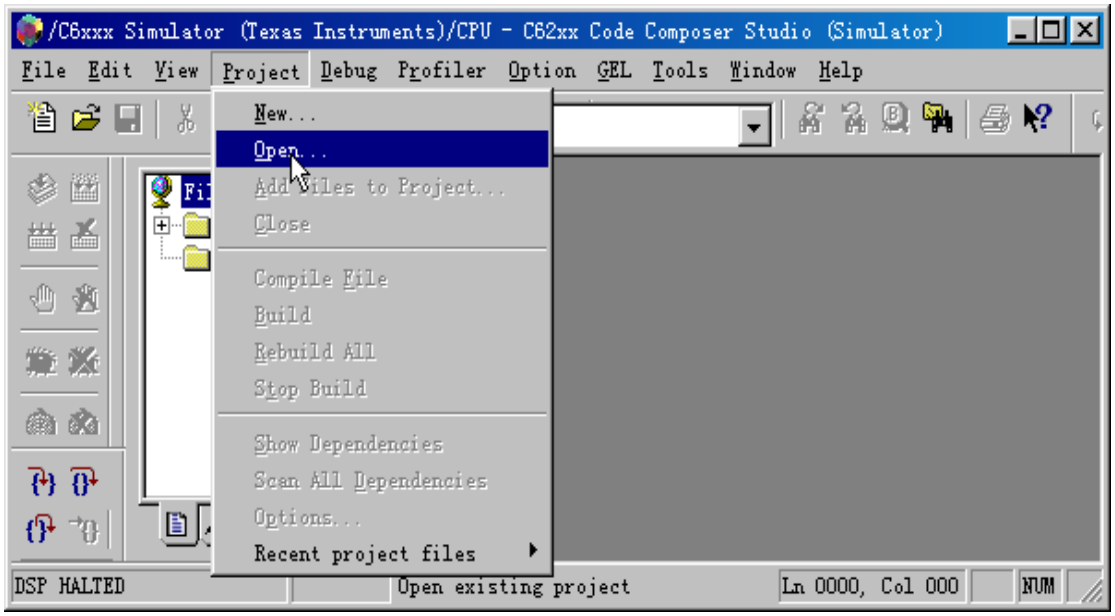
下面是 Code Composer Studio Tutorial 的例子：



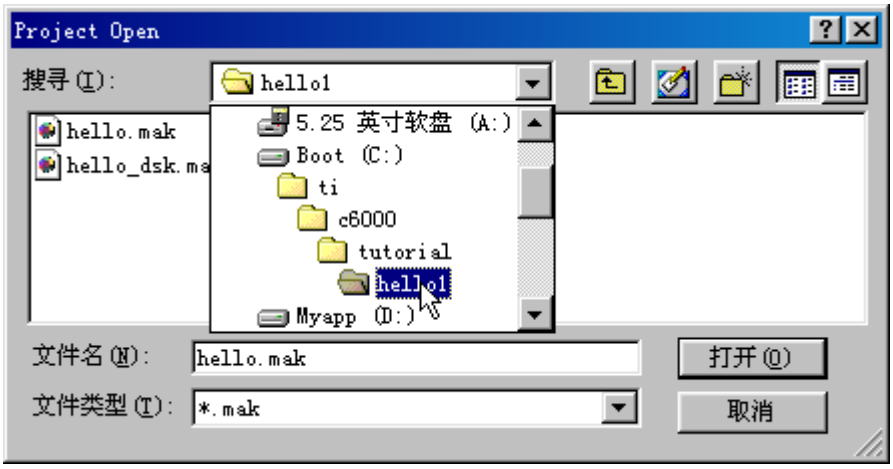
分别从生成一个简单的“ Hello World ”程序，到使用 DSP/BIOS 功能，到程序的调试，实时分析，I/O 操作等分 6 课来讲解，可以领略 TI 的 CCS 的强大功能。

下面以“ Hello World ”程序为例讲一下 CCS 的使用。

首先打开一个 Project 文件



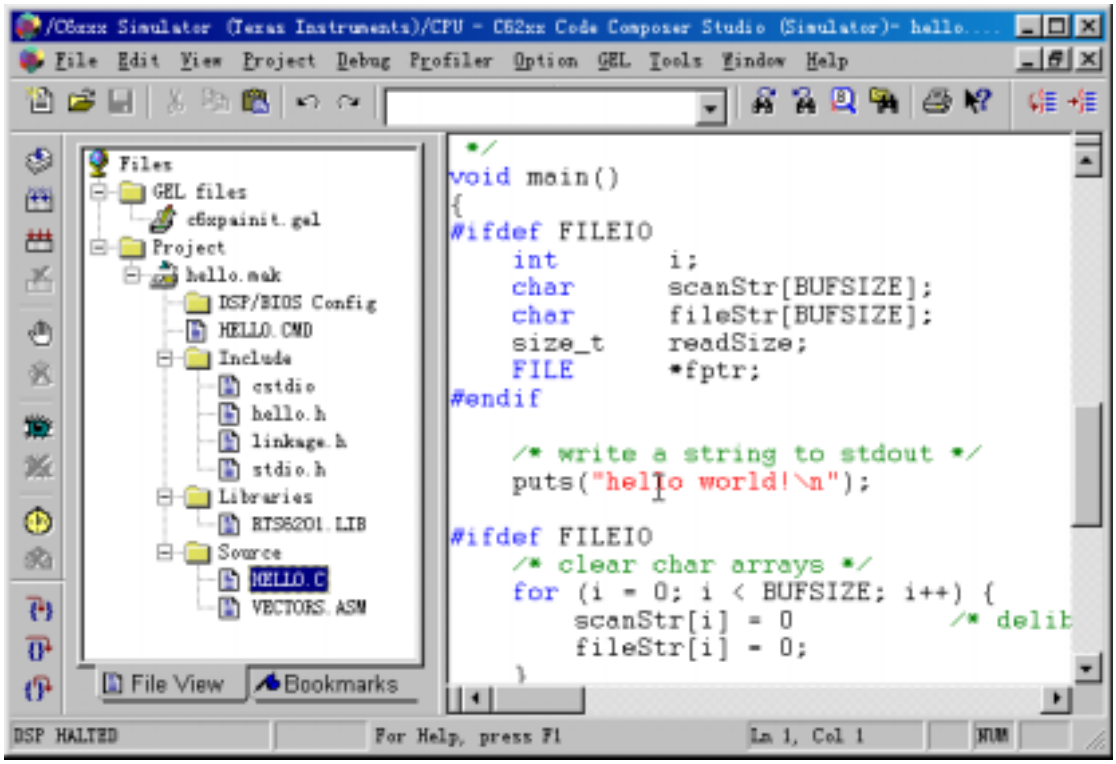
这些文件的路径如下图所示：



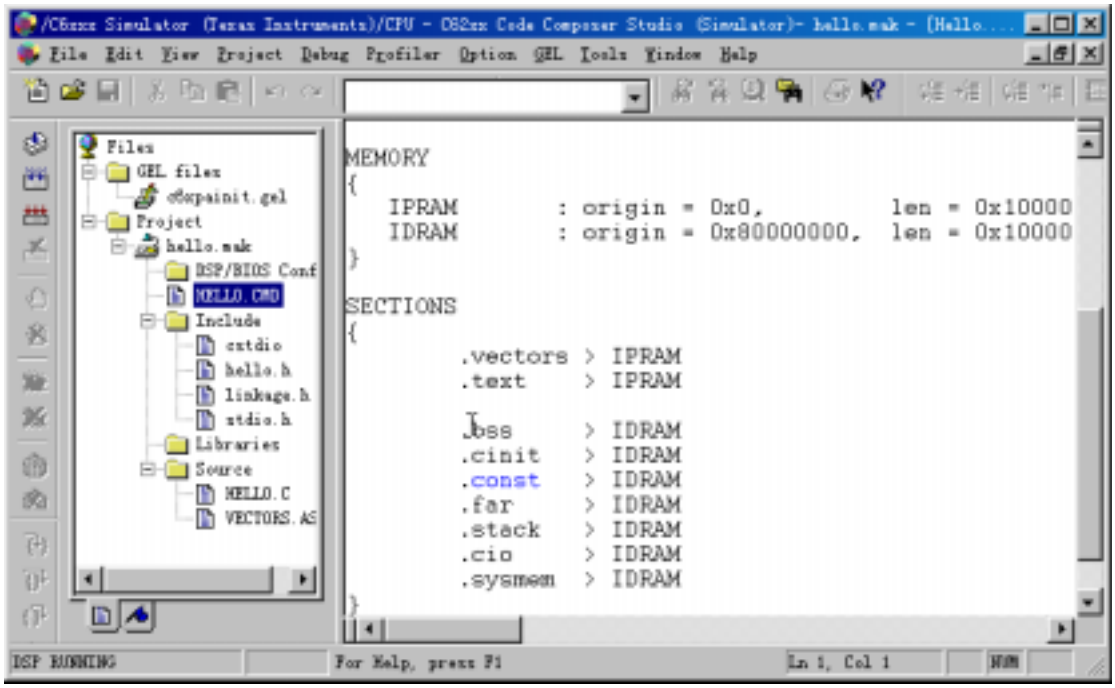
打开 hello.mak，会看到如下图所示的界面。将 File View 栏中的“ + ”号都打开，会看到整个项目工程中的所有资源。

其中*.c 文件和*.h 文件与普通的 C 语言编程中是一致的（TI 编译器支持 ANSI C 标准）。需要指出的是三个文件：HELLO.CMD、RTS6201.LIB、VECTORS.ASM。HELLO.CMD 文件给出了程序空间和数据空间的设置、及编译后各程序段在程序或数据空间的具体位置。RTS6201.LIB 文件为 DSP 运行时库，VECTORS.ASM 为中断

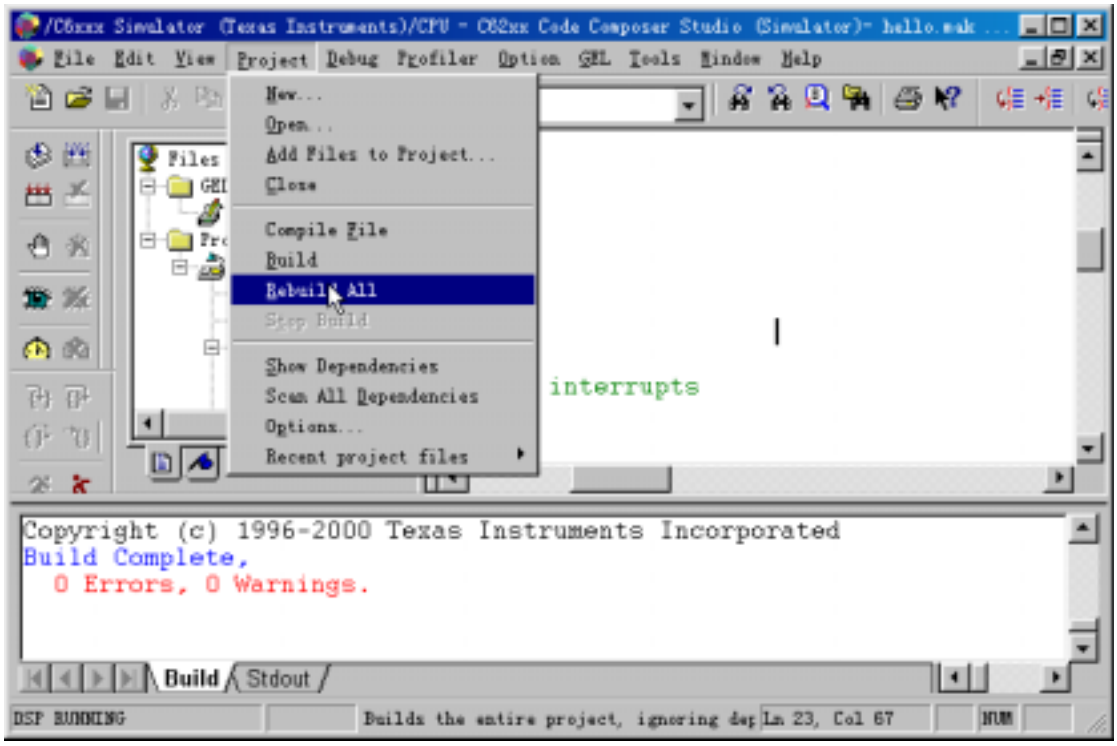
向量表，都是区别于纯软件编程的独到之处，熟悉以后会有更深的体会。



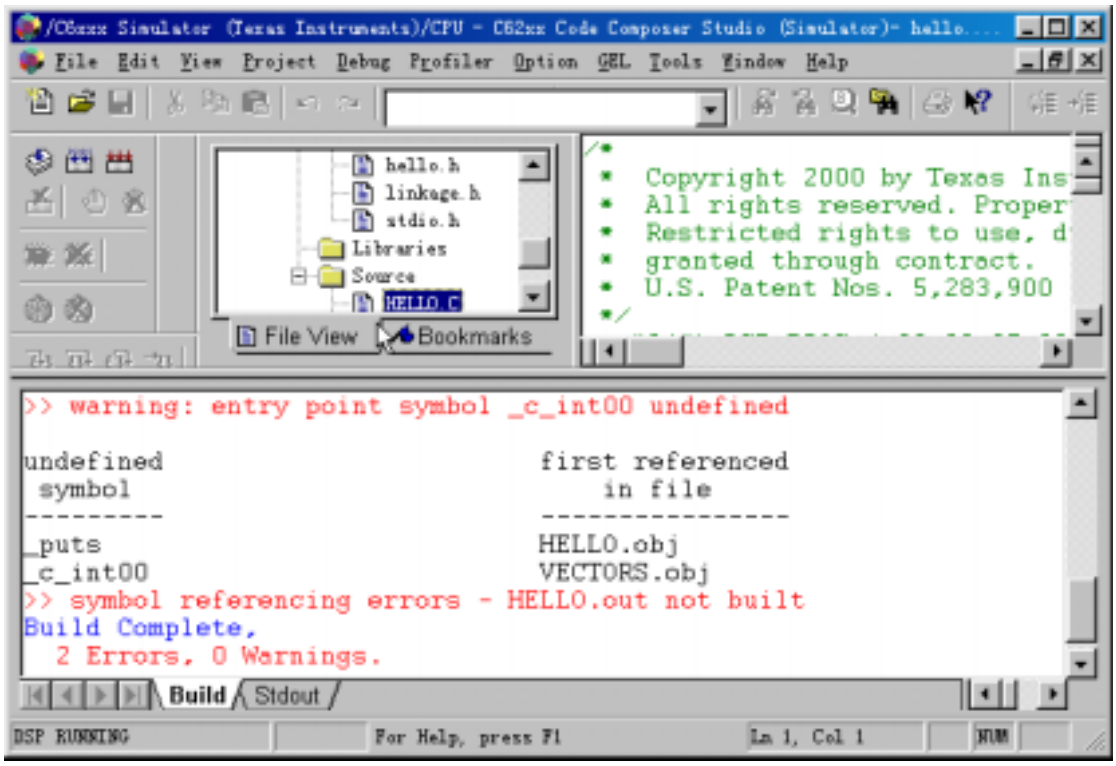
下图为 HELLO.CMD 文件的代码，MEMORY 分为程序空间 IPRAM 和数据空间 IDRAM，并分别给出了起始地址 origin 和长度 len，各段在 MEMORY 空间的分配也作了定义。对于实际的目标板硬件系统，由实际的存储器空间及 DSP 芯片上的存储空间决定。对于软件仿真，可以不考虑有没有 MEMORY 资源。



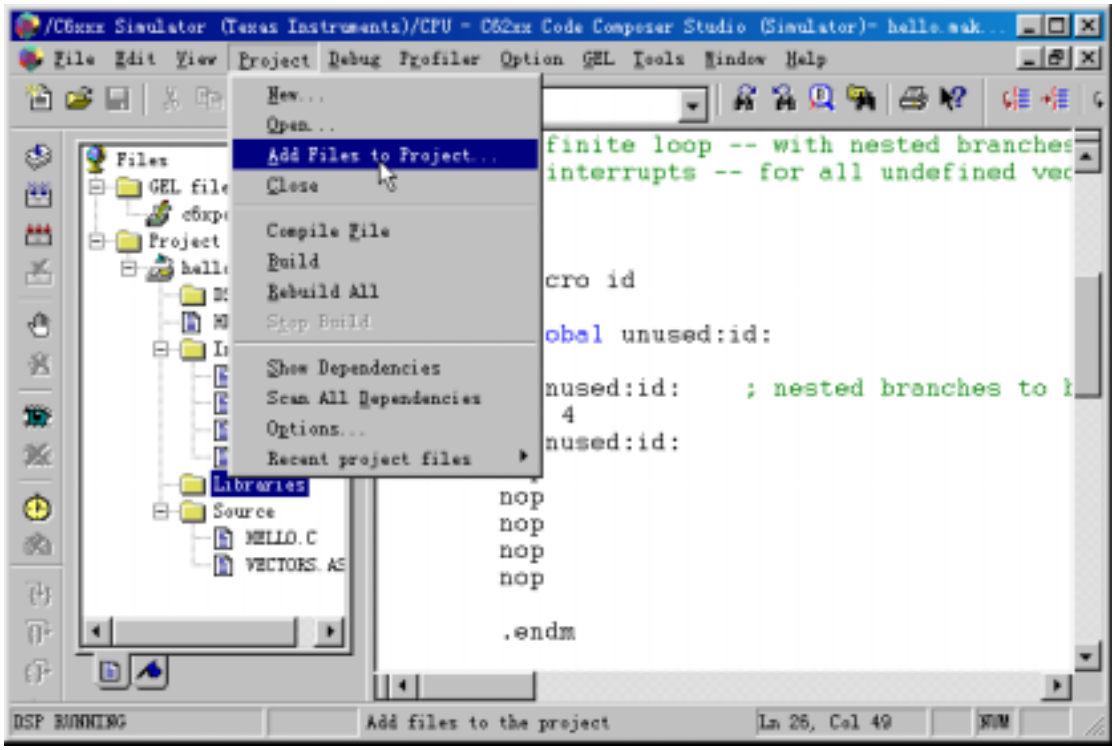
直接对该工程进行编译，会得到如下结果，试一下吧！也可以试一下快捷工具条上的按钮，随便点击鼠标右键，也会有意外的收获。怎么样？没有错误吧！



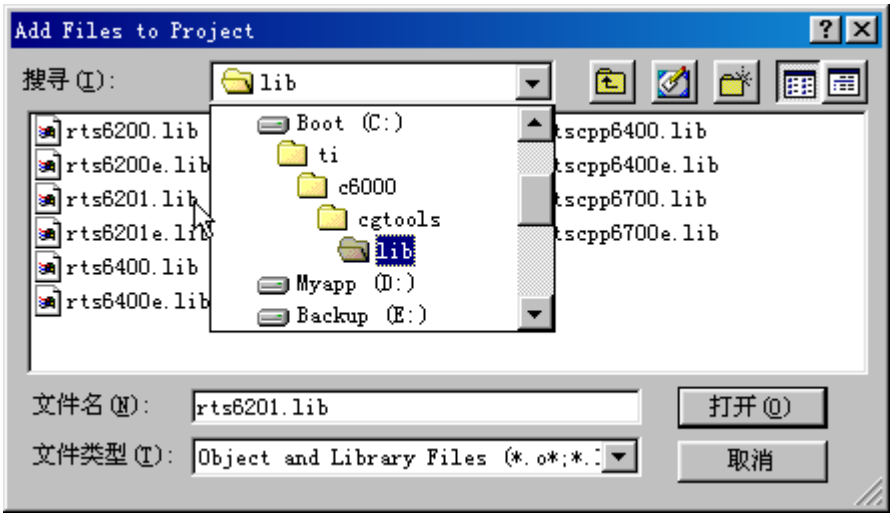
自己在编写工程项目文件时，经常会遇到下面的问题，没有 C 语言程序的入口函数，细心比较一下会发现工程文件中缺少了一个运行时支持库 RTS6201.LIB，不同的 DSP 芯片需要不同的运行时库来支持。



下面向项目工程中加上运行时库 RTS6201.LIB 来纠正刚才的编译错误，**同样的方法可以用来向工程中添加*.c、*.cmd、*.asm 文件**。*.h 文件在编译时会自己找到（当然需要在环境变量中设置好啦，一般不需要改动）。



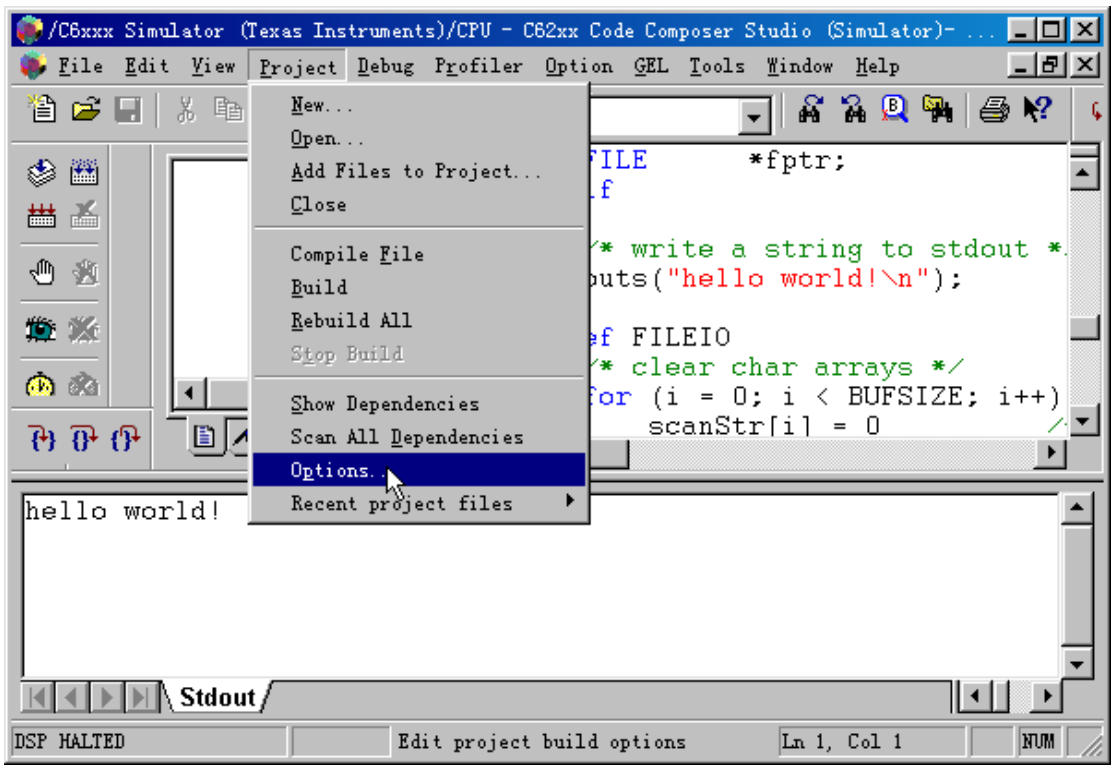
运行时库在 TI 的缺省路径下，注意将文件类型改为*.lib，

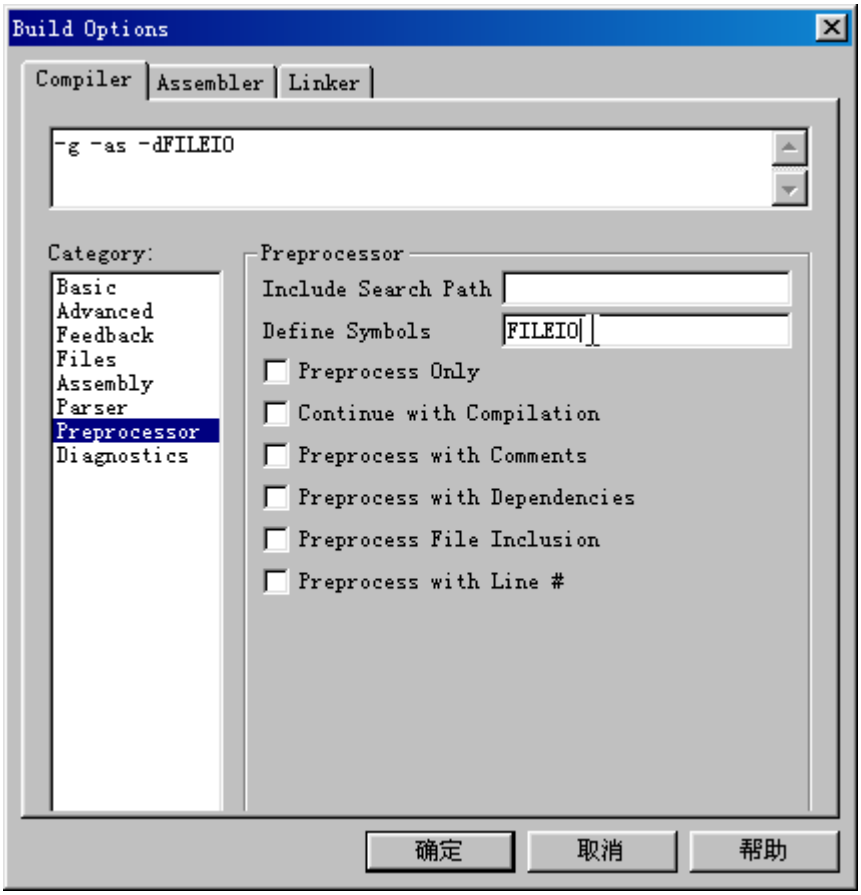


大家可能注意到，在 HELLO.C 文件中有这样的定义：

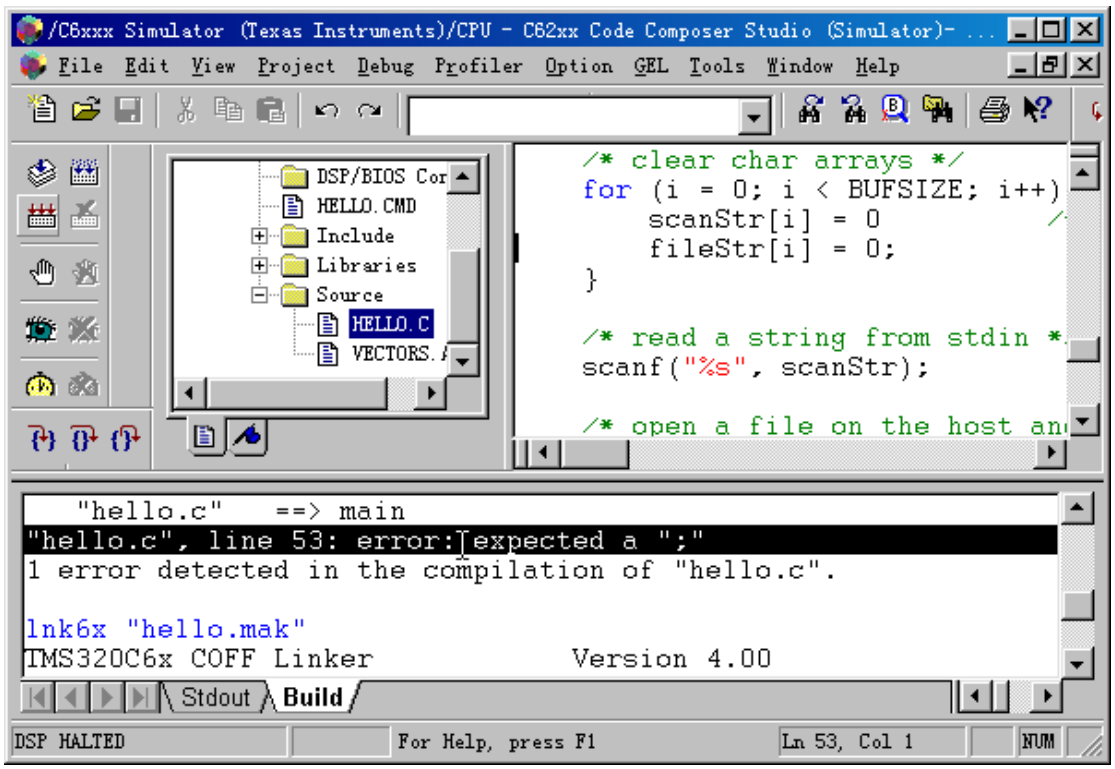
```
#ifdef FILEIO
    int       i;
    char      scanStr[BUFSIZE];
    char      fileStr[BUFSIZE];
    size_t    readSize;
    FILE      *fptr;
#endif
#ifdef FILEIO
    /* clear char arrays */
    for (i = 0; i < BUFSIZE; i++) {
        scanStr[i] = 0          /* deliberate syntax error */
        fileStr[i] = 0;
    }

    /* read a string from stdin */
    scanf("%s", scanStr);

    /* open a file on the host and write char array */
    fptr = fopen("file.txt", "w");
    fprintf(fptr, "%s", scanStr);
    fclose(fptr);

    /* open a file on the host and read char array */
    fptr = fopen("file.txt", "r");
    fseek(fptr, 0L, SEEK_SET);
    readSize = fread(fileStr, sizeof(char), BUFSIZE, fptr);
    printf("Read a %d byte char array: %s \n", readSize, fileStr);
    fclose(fptr);
#endif
```

其中还有一些变量的定义和对文件的操作，运行编译好的程序后好象这些语句都没有执行，因为在 CCS 的编译环境中这个参数还没有定义。按下图进行设置：





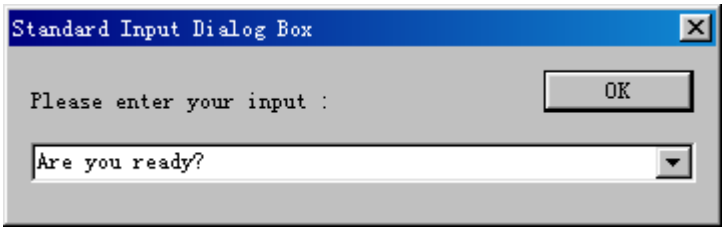
设置完成后可以进行重新编译，会发现新的错误（如果没有出现这个错误，说明设置的不对）。双击这个错误，在 HELLO.C 文件中，光标会出现在出错的地方。



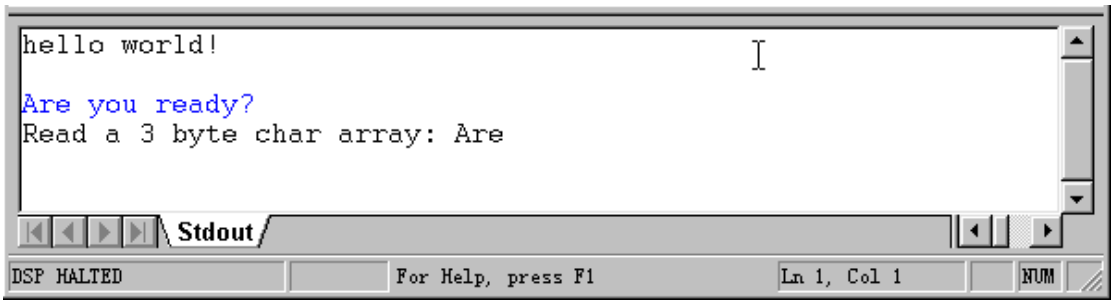
在第 52 行的这一句，可以看到语句的后面没有加“ 分号 ”，

```
scanStr[i] = 0
```

加上“ 分号 ”后重新编译，ok？！加载 hello.out 运行，会出现下面的输入界面，



输入一串文字并确定，在“ Stdout ”窗口会有下面的显示，



小结：在这里简单介绍了 CCS 的使用，包括 CCS 的设置、帮助文件的使用，（**TI 的帮助文件系统、详细地介绍了 CCS 的使用，强烈建议用户认真学习。**）

并以“ Hello World ”程序为例对 CCS 的使用中容易出现问题的一些地方作了一般的介绍，包括运行时库的添加、预编译定义设置等，用户在使用过程中会不断发现问题，通过使用 TI 的帮助文件及配套的资料会不断提高，不可急于求成，如果用户对 Visual C++比较熟悉，学起来会快很多；相反，那肯定要多花一些时间来学习了，学习 CCS 跟学习 Visual C++一样（简单/复杂？），但需要对硬件有一定的了解。