**实验一 基本指令实验**

**一、熟悉CCS集成开发环境**

1、建立或打开工程

2、编辑源文件或其它工程文件

主要了解两种文件：

①源文件：用汇编语言写的，后缀为.ASM；用C语言写的，后缀为.c。

②链接命令文件：后缀为.cmd，主要对存储空间进行分配（见教材P135~137）。其内容主要包含“MEMORY”和“SECTIONS”两节。

“MEMORY”将存储器划分为多个独立的存储区间，如：“PROG”、“DATA”等，其中“PAGE 0”用于程序存储器，“PAGE 1”用于数据存储器。

“SECTIONS”将程序中使用的各个段名映射到各个存储区间，这样链接程序就可以将目标代码装入存储器的对应位置或者将预定义的段名映射到存储器的具体位置。比如：§3.1程序中定义的“.bss”和“.data”在数据存储器中；而“.text”和“STACK”在程序存储器。

3、需要熟悉的菜单或工具栏命令（详情参考实验指导书“§2.3 CCS的使用”）

①“Project”菜单中的“New”、“Open”用于新建工程或打开已有的工程。

②“File”菜单中的“New”用于新建文件；“Project”菜单中的“Add File to Project”用于向工程中添加已有文件。

③打开和编辑文件。

④建造（Building）：一般选“Project”菜单中的“Rebuild All”，或者工具栏上的，完成后将生成一个“.out”文件，默认存放在当前工程目录下或“debug”目录中。

注意：当打开多个工程时，只有当前“活动”工程（资源管理器中加粗显示）才会被Build；可以通过右键菜单“Set as Active Project”将工程设置为“活动”工程；为避免出错，最好每次只打开一个工程。

⑤装载程序：“File”菜单中的“Load Program”。

⑥运行程序：“Debug”菜单中的“Step Into”（F8或工具栏按钮）：单步执行；“Run”（F5或工具栏按钮）：全速运行程序；“Halt”（Shift+F5或工具栏按钮）：停止运行。

⑦通过“View”菜单查看程序运行的结果和数据，结合具体的实例给大家讲解。

**二、熟悉汇编语言编写DSP程序**

1、汇编程序的设计见教材P94~P141第四章4.1~4.4的内容。

2、查询汇编指令可参考“[附录2] TMS320C54X指令速查表”。

3、通过“[实验3.1] 循环操作”，详细讲解每一条指令的作用和运行结果，其余例程，只关注相关的指令。

**三、例程详细讲解（§**3.1**）**

1、启动CCS，注意选择模拟的CPU（“C5416 Device Simulator”）。

2、打开Ex3\_1，展开资源管理器，打开Ex4\_1.asm和Ex4\_1.cmd文件。

3、Build工程，即选择“Rebuild All”命令。注意观察底部窗口中显示编译连接信息，有无错误信息。

4、装载程序：即选择 “Load Program”命令。

装载程序后，注意观察PC指针的位置，以及内存地址显示的变化。

5、“View”菜单“Registers”查看寄存器数据。

6、“View”菜单“memory……”查看存储器数据。

“Page”选项：Program为程序存储器；Data为数据存储器；I/O为I/O存储空间。一般选择Data。

“Address”：将要显示的起始地址，根据CMD文件中的设置，选择0x60。

“Format”：选择数据显示的格式。

7、结合CMD文件、汇编源文件、数据存储器、程序存储器，讲解内存分配及地址映射关系：

“MEMORY”一节定义了数据存储空间和程序存储空间的起始地址和长度，这样程序的起始地址为“1000H”，数据的地址为“60H”。

“SECTIONS”一节定义了“.text”、“.data”、“.bss”、“STACK”四个段名，分别存放到程序存储空间和数据存储空间。在汇编源文件中，使用这些段名定义的数据或程序代码在链接的时候即会装入或者对应到相应的存储空间。一般地：“.text”为程序代码，“.data”为已初始化的数据（常量），“.bss”为未初始化数据（变量），“STACK”为堆栈。

8、单步运行程序，观察寄存器和存储器的变化，逐条讲解指令。

注意：

①注意标号的使用以及汇编程序正确的书写格式。

②注意观察寄存器和存储器的变化，了解汇编指令的功能。

③循环或重复次数的正确设置。

④在循环或重复操作中，数据的寻址方式通常采用寄存器间接寻址。

⑤累加之前，注意先对累加器清零。

⑥CALL指令与B指令的区别。

9、找出构造循环的三条指令，说明其要点：

①循环控制变量。

②循环开始处设置标号以便于返回。

③循环结束处用BANZ条件转移指令测试循环控制变量（并增减其值），根据其值是否满足循环条件，跳回循环开始处继续下一次循环或者结束循环继续执行后续指令。

④循环体内，对变量的寻址方式，应该使用寄存器间接寻址，并自动增减地址值。

10、实验中的循环结构，是先执行循环再测试循环次数，因此无论如何设置循环次数，都至少循环一次。可以将循环变量的测试放在循环开始处，这样如果初始条件不满足时，则循环将执行0次。

11、改由重复指令实现循环操作。注意：RPT指令重复执行的只有一条指令。修改代码后，重新Build工程、装载程序、运行后得到的结果应与修改前相同。

12、需要记录的实验数据（用截图方式）：

①程序运行结束后，内存中的数据。

②构造循环的三条指令。

③修改为重复指令后的对应指令。

**四、其余例程讲解（§**3.2、**§**3.3、**§**3.5**）**

1、§3.2 双操作数乘法。

①双操作数乘法的执行的硬件条件是：多总线结构，可以在一个周期内同时寻址多个数据和系数。

②双操作数指令是用间接寻址方式获得操作数的，并且只能用AR2到AR5的辅助寄存器。

③实验中实现乘法累加使用的是MAC指令，与MPY格式相同，只是多了一个加法操作。

④因为乘法操作容易导致结果溢出，所以扩展到32位（2个存储字）。注意累加器中高低位与存储地址的对应关系。

⑤改用单操作数乘加指令来实现：

A、单操作数MAC指令执行时，另一个操作数默认在T寄存器中。

B、修改时，需要先将其中一个操作数装入T寄存器中。

C、去掉原双操作数MAC指令中装入T寄存器的那个操作数。

D、需要2条指令才能实现一条双操作数MAC指令的功能作。

E、原程序中采用重复指令（RPTZ）实现10次循环，需要类似**§**3.1一样采用BANZ指令重新构造循环。

2、§3.3并行运算。

①并行运算的执行的硬件条件是：多总线结构，可以在一个周期内同时读写多个存储单元。

②并行指令是1条指令，为便于理解，常写成两行，写成用“||”分隔的两个操作。

③程序实现的功能很简单，完成z = x + y和 f = e + d的计算，在存第一个结果z的同时读出第二个表达式的加数e。

④所有并行指令都是单字单周期指令，都工作在累加器的高位。

⑤改用非并行指令来实现：

A、去掉分隔符“||”，将一条并行指令变成2条指令。

B、并行指令中使用累加器的高位，但并未显式表示出来，需要修改为使用累加器的高位的形式。

3、§3.5长字运算。

①长字指令一次操作32位（2个存储字）。

②注意高低位与存储地址的对应关系。

③改用单字行指令来实现：

A、每1条长字指令需要2条单字指令实现。

B、应先装入或者加高位（因为最高位为符号为），可以通过左移16位操作实现。

C、低位应通过加法指令装入，因为低16位的最高位并非符号位，加法指令应使用ADDS。

**五、实验数据记录与提交**

1、将需要记录的数据截图，粘贴到文档“《实验一 基本指令实验》数据记录”中。

2、从数字化校园网登陆“开放实验选课系统”，在左栏点击“实验成绩查询”，右侧页面“作业”一列下面有“上传”选项，点击即可进入上传页面，将填好的word文档上传到系统中。

