第3章 CHAPTER 3

TMS320C55x DSP 汇编指令系统

本章主要讲述 TMS320C55x DSP 汇编指令的术语、符号，汇编指令的寻址方式，以及指令系统。

知识要点

- ◆ 熟练掌握 TMS320C55x DSP 汇编指令的术语、符号以及相关缩写及运算规则。
- ◆ 理解 TMS320C55x DSP 寻址方式以及物理实现过程。
- ◆ 理解汇编指令结构，熟练掌握常用汇编指令的使用方法。

3.1 术语、符号与缩写

本节列出 TMS320C55x DSP 助记符指令集和单独的指令说明中用到的术语、符号以及缩写，指令集的注释、规则以及不可重复指令的列表。

3.1.1 指令集术语、符号和缩写

表 3-1 列出了用到的术语、符号和缩写。表 3-2 列出了指令集以及单独的指令说明中用到的运算符。

表 3-1 指令集中的术语、符号和缩写

符 号	含 义
[]	可选的操作数
40	如果可选的关键字“40”被用到指令中，指令的执行使得 M40 被局部置“1”
ACB	把 D 单元的寄存器传给 A 单元和 P 单元运算单元的总线
ACOVx	累加器溢出状态位：ACOV0、ACOV1、ACOV2、ACOV3
ACw，ACx，ACy，ACz	累加器：AC0、AC1、AC2、AC3
ARn_mod	所选辅助寄存器(ARn)的内容在地址产生单元中被预修改或后修改
ARx，ARy	辅助寄存器：AR0、AR1、AR2、AR3、AR4、AR5、AR6、AR7
AU	A 单元
Baddr	寄存器位地址
BitIn	移位进：测试控制标志 2(TC2)或 CARRY 状态位
BitOut	移位出：测试控制标志 2(TC2)或 CARRY 状态位

续表

符 号	含 义
BORROW	CARRY 状态位的逻辑补
C,Cycles	执行的周期数。条件指令中,x/y 字段的意义是:x 个周期(若条件为真),y 个周期(若条件为假)
CA	系数地址产生单元
CARRY	进位状态位的值
Cmem	系数间接操作数引用数据空间中的 16 位或 32 位数值
cond	基于累加器(ACx)、辅助寄存器(ARx)、临时寄行器(Tx)、测试控制(TCx)标志或进位状态位的条件
CR	系数读总线
CSR	单重复计数寄存器
DA	数据地址产生单元
DR	数据读总线
dst	目标累加器(ACx)、辅助寄存器(ARx)的低 16 位或临时寄存器(Tx):AC0、AC1、AC2、AC3;AR0、AR1、AR2、AR3、AR4、AR5、AR6、AR7;T0、T1、T2、T3
DU	D 单元
DW	数据写总线
Dx	x 位长的数据地址标识(绝对地址)
E	表明指令是否包含一个并行使能位
kx	x 位长的无符号常数
Kx	x 位长的带符号常数
Lmem	长字单数据存储器访问(32 位数据访问);亦同于 Smem
lx	x 位长的程序地址标识(相对于寄存器 PC 的无符号偏移量)
Lx	x 位长的程序地址标识(相对于寄存器 PC 的有符号偏移量)
Operator	指令的运算符
Pipe,Pipeline	该指令执行的流水线阶段:AD 寻址,D 译码,R 读,x 执行
pmad	程序存储器地址
Px	x 位长的程序成数据地址标识(绝对地址)
RELOP	关系运算符:==(等于)、<(小于)、>=(大于等于)、!=(不等于)
R 或 rnd	若关键字 R 或 rnd 用于指令中,则该指令执行取整操作
RPTC	单重复计数寄存器
S,Size	指令长度(字节)
SA	堆栈地址产生单元
saturate	若可选的关键字 saturate 用于指令中,运算的 40 位输出被饱和处理
SHFT	4 位立即数移位值,0～15
SHIFTW	6 位立即数移位值,－32～＋31
Smem	单字数据存储器访问(16 位数据访问)
SP	数据堆栈指针

续表

符　号	含　义
src	源累加器(ACx)、辅助寄存器(ARx)的低16位或临时寄存器(Tx)：AC0、AC1、AC2、AC3；AR0、AR1、AR2、AR3、AR4、AR5、AR6、AR7；T0、T1、T2、T3
SSP	系统堆栈指针
STx	状态寄存器：ST0、ST1、ST2、ST3
TAx，TAy	辅助寄存器(ARx)或临时寄存器(Tx)：AR0、AR1、AR2、AR3、AR4、AR5、AR6、AR7或T0、T1、T2、T3
TCx，TCy	测试控制标志：TC1、TC2
TRNx	转换寄存器：TRN0、TRN2
Tx，Ty	临时寄存器：T0、T1、T2、T3
U或uns	若关键字U或uns用于输入操作数，则操作数被0扩展
XACdst	目标扩展寄存器：所有的23位系数数据指针(XCDP)和扩展辅助寄存器(XARx)：XAR0～XAR7
XACsrc	源扩展寄存器：所有的23位系数数据指针(XCDP)和扩展辅助寄存器(XARx)：XAR0～XAR7
XAdst	目标扩展寄存器：所有的23位数据堆栈指针(XSP)、系统堆栈指针(XSSP)、数据页指针(XDP)、系数数据指针(XCDP)和扩展辅助寄存器(XARx)：XAR0～XAR7
XARx	23位的扩展辅助寄存器(XARx)：XAR0～XAR7
XAsrc	源扩展寄存器：所有的23位数据堆栈指针(XSP)、系统堆栈指针(XSSP)、数据页指针(XDP)、系数数据指针(XCDP)和扩展辅助寄存器(XARx)：XAR0～XAR7
xdst	累加器：AC0、AC1、AC2、AC3 目标扩展寄存器：所有的23位数据堆栈指针(XSP)、系统堆栈指针(XSSP)、数据页指针(XDP)、系数数据指针(XCDP)和扩展辅助寄存器(XARx)：XAR0～XAR7
xsrc	累加器：AC0、AC1、AC2、AC3 扩展寄存器：所有的23位数据堆栈指针(XSP)、系统堆栈指针(XSSP)、数据页指针(XDP)、系数数据指针(XCDP)和扩展辅助寄存器(XARx)：XAR0～XAR7
Xmem，Ymem	间接双数据存储器访问(两个数据访问)

表3-2　指令集中用到的运算符

符　号	运算符	赋值	符　号	运算符	赋值
＋　－　～	一元的加、减、1位取反	从右到左	＞　＞＝	大于、大于等于	从左到右
＊　／　％	乘、除、取模	从左到右	＝＝　！＝	等于、不等于	从左到右
＋　－	加、减	从左到右	&	位与	从左到右
＜＜　＞＞	带符号的左移、右移	从左到右	|	位或	从左到右
＜＜＜　＞＞＞	逻辑左移、逻辑右移	从左到右	^	位异或	从左到右
＜　＜＝	小于、小于等于	从左到右			

注：一元＋、－、＊运算比二元运算有更高的优先级。

3.1.2　指令集条件字段

表3-3列出了条件指令中的cond字段可用的测试条件。

表 3-3　指令集条件(cond)字段

位或寄存器	条件(cond)字段	若条件为真…
累加器	测试累加器(ACx)内容对于 0。与 0 比较,取决于状态位 M40 若 M40=0,ACx(31～0)被比较与 0 若 M40=1,ACx(39～0)被比较与 0	
	ACx == #0	ACx 等于 0
	ACx < #0	ACx 小于 0
	ACx > #0	ACx 大于 0
	ACx != #0	ACx 不等于 0
	ACx <= #0	ACx 小于等于 0
	ACx >= #0	ACx 大于等于 0
累加器溢出状态位	测试累加器溢出状态位(ACOVx)对于 1。当可选符号!用在位名称之前时,测试该位对于 0。当此条件使用时,相应地,此 ACOVx 被清零	
	overflow(ACx)	ACOVx 位置"1"
	!overflow(ACx)	ACOVx 位清"0"
辅助寄存器	测试辅助寄存器(ARx)内容对于 0	
	ARx==#0	ARx 内容等于 1
	ARx<#0	ARx 内容小于 1
	ARx>#0	ARx 内容大于 1
	ARx != #0	ARx 内容不等于 1
	ARx <= #0	ARx 内容小于等于 1
	ARx >= #0	ARx 内容大于等于 1
进位状态位	测试进位(CARRY)状态位对于 1。当可选符号!用在位名之前,测试该位对于 0	
	CARRY	CARRY 位置"1"
	!CARRY	CARRY 位清"0"
临时寄存器	测试临时寄存器内容对于 0	
	Tx == #0	Tx 内容等于 0
	Tx < #0	Tx 内容小于 0
	Tx > #0	Tx 内容大于 0
	Tx != #0	Tx 内容不等于 0
	Tx <= #0	Tx 内容小于等于 0
	Tx >= #0	Tx 内容大于等于 0
测试控制标志位	测试控制标志(TC1 和 TC2)对于 1。当可选!符号使用在标志名称之前时,测试该标志位对于 0	
	TCx	TCx 标志置"1"
	!TCx	TCx 标志清"0"
	TC1 和 TC2 可以使用 AND(&)、OR(|)和 XOR(^)逻辑组合	
	TC1 & TC2	TC1 与 TC2 等于 1
	!TC1 & TC2	TC1 的反与 TC2 等于 1
	TC1 & !TC2	TC1 与 TC2 的反等于 1
	!TC1 & !TC2	TC1 的反与 TC2 的反等于 1
	TC1 | TC2	TC1 或 TC2 等于 1
	!TC1 | TC2	TC1 的反或 TC2 等于 1
	TC1 | !TC2	TC1 或 TC2 的反等于 1
	!TC1 | !TC2	TC1 的反或 TC2 的反等 1
	TC1 ^ TC2	TC1 异或 TC2 等于 1
	!TC1 ^ TC2	TC1 的反异或 TC2 等于 1
	TC1 ^ !TC2	TC1 异或 TC2 的反等于 1
	!TC1 ^ !TC2	TC1 的反异或 TC2 的反等于 1

3.1.3 状态位的影响

1. 累加器溢出状态位(ACOVx)

ACOV[0～3]取决于 M40。

(1) 当 M40＝0 时,在第 31 位处检测溢出。

(2) 当 M40＝1 时,在第 39 位处检测溢出。

如果检测到溢出,则目的累加器溢出状态位置“1”。

2. C54CM 状态位

(1) 当 C54CM＝0 时,为增强模式,CPU 支持为 TMS320C55x 系列 DSP 开发的代码。

(2) 当 C54CM＝1 时,为兼容模式,所有的 C55x CPU 资源仍然可利用。因此,当用户移植代码时,可以利用 C55x DSP 的额外功能来优化代码。当用户移植为 TMS320C54x 系列 DSP 开发的代码时,必须置此位。

3. 进位(CARRY)状态位

(1) 当 M40＝0 时,在第 31 位处检测进位/借位。

(2) 当 M40＝1 时,在第 39 位处检测进位/借位。

当执行影响进位状态位的逻辑移位或带符号移位的操作,并且移位数为 0 时,进位状态位被清零。

4. FRCT 状态位

(1) 当 FRCT＝0 时,不进入分数模式,乘操作的结果不执行移位。

(2) 当 FRCT＝1 时,进入分数模式,乘操作的结果左移 1 位,消除一个额外的符号位。

5. INTM 状态位

INTM 状态位全局使能/无效可屏蔽中断。此位对不可屏蔽中断不起作用。

(1) 当 INTM＝0 时,所有非屏蔽中断被使能。

(2) 当 INTM＝1 时,所有可屏蔽中断被禁止。

6. M40 状态位

(1) 当 M40＝0 时:

在第 31 位检测溢出;

在第 31 位检测进位/借位;

饱和值是 00 7FFF FFFFh(正溢出)或 FF 8000 0000h(负溢出);

TMS320C54x DSP 兼容模式;

对于条件指令,对于 0 的比较使用了 32 位,即 ACx(31～0)。

(2) 当 M40＝1 时:

在第 39 位检测溢出;

在第 39 位检测进位/借位;

饱和值是 7F FFFF FFFFh(正溢出)或 80 0000 0000h(负溢出);

对于条件指令,对于 0 的比较使用了 40 位,ACx(39～0)。

对其他状态位的影响，请参阅有关资料，这里不再赘述。

7. SXMD 状态位

此状态位控制 D 单元中的运算。

(1) 当 SXMD=0 时，输入操作数进行 0 扩展。

(2) 当 SXMD=1 时，输入操作数进行符号扩展。

3.1.4 指令集注释和规则

1. 注释

助记符语法关键字和操作数限定符不区分大小写，写作：

```
ABDST * AR0, * ar1,AC0,ac1
```

或

```
aBdST * ar0, * aR1,aC0,Ac1
```

可交换运算(+、*、&、|、^)的操作数可以以任意顺序排列。

2. 规则

简单指令不允许扩写到多行。一个例外是使用双冒号(::)的单指令，这是并行指令的标记。例如：

```
MPYR40 uns(Xmem),uns(Cmem),ACx
:: MPYR40 uns(Ymem),uns(Cmem),ACy
```

用户定义的并行指令(使用 ‖ 标记)允许扩写到多行。以下的例子是合法的：

```
MOV AC0,AC1||MOV AC2,AC3
MOV AC0,AC1||
MOV AC2,AC3
MOV AC0,AC1
||MOV AC2,AC3
MOV AC0,AC1
||
MOV AC2,AC3
```

(1) 保留字

寄存器名是被保留的，它们不能被用作标识符、标号等的名称。助记符语法名不被保留。

(2) 助记符语法字根

表 3-4 列出了助记符语法中使用的字根。

表 3-4 助记符语法字根

字 根	含 义	字 根	含 义
ABS	绝对值	B	转移
ADD	加	CALL	函数调用
AND	位与	CLR	清零

续表

字　根	含　义	字　根	含　义
CMP	比较	ROL	循环左移
CNT	计数	ROR	循环右移
EXP	指数	RPT	重复
MAC	乘加	SAT	饱和
MAR	修改辅助寄存器内容	SET	置“1”
MAS	乘减	SFT	移位(左/右取决于移位数的符号)
MAX	最大值	SQA	平方和
MIN	最小值	SQR	平方
MOV	移动数据	SQS	平方差
MPY	乘	SUB	减
NEG	取反(二元补码)	SWAP	交换寄存器内容
NOT	位补(一元补码)	TST	测试位
OR	位或	XOR	位异或
POP	从堆栈顶部取出	XPA	扩展
PSH	压入堆栈顶部	XTR	提取
RET	返回		

(3) 助记符语法前缀

助记符语法前缀如表 3-5 所示。

表 3-5　助记符语法前缀

前缀	含　义
A	指令作用于寻址阶段,受循环寻址影响。产生于 DAGEN 功能单元中,并不能和使用双寻址方式的指令并行放在一起
B	位指令。注意,B 也是一个字根(转移)、后缀(借位)和前缀(位)。用前后关系的差别来防止混淆

(4) 字母和地址操作数

字母在助记符序列中被记为 K 或 k 字段。在需要偏移量的 Smem 地址模式中,偏移量也是一个字母(K16 或 k3)。8 位和 16 位字母可以在链接时重分配。其他字母的值在汇编时必须确定。

地址是在助记符序列中被 P、L、I 标记的部分。16 位和 24 位绝对地址 Smem 模式是被@语法标记的。地址可以是汇编时间常数或链接时间已知的符号常数或表达式。

字母和地址都遵守语法规则 1。规则 2 和规则 3 只对地址成立。

① 规则 1：一个合法的地址或字母是跟在 # 后的,如下所示：一个数(#123)；一个标识符(#FOO)；一个括号表达式(#(FOO+2))。

注意：不用于表达式内部。

② 规则 2：当地址用于 DMA 中时,无论地址是数、标记还是表达式,地址无需跟在符号 # 之后。以下表示都正确。

```
@ #123
@ 123
@ #FOO
@ FOO
@ #(FOO+2)
@ (FOO+2)
```

③ 规则3：当地址用于除DMA之外的某些背景中(如分支目标或Smem绝对地址)时，地址通常需要加#。但为了方便，标识符前的#可以省略。以下都是正确的表示。

分支	绝对地址
B #123	*(#123)
B #FOO	*(#FOO)
B FOO	*(FOO)
B #(FOO+2)	*(#(FOO+2))
B 123	*(123)
B (FOO+2)	*((FOO+2))

(5) 存储器操作数

① Smem的语法与Lmem或Baddr相同。

② 在下列指令语法中，Smem不能引用一个存储器映射寄存器(MMR)。没有指令可以访问存储器映射寄存器中的字节。在下列语法中，如果Smem是一个MMR，DSP会向CPU发送一个硬件总线错误中断(BERRINT)请求：

```
MOV [uns()high_byte(Smem)()],dst
MOV [uns()low_byte(Smem)()],dst
MOV high_byte(Smem) <<#SHIFTW,ACx
MOV low_byte(Smem) <<#SHIFTW,ACx
MOV src,high_byte(Smem)
MOV src,low_byte(Smem)
```

③ Xmem语法和Ymem语法一致。

④ 系数操作数(Cmem)的语法如下所示。

```
*CDP
*CDP+
*CDP-
*(CDP+T0),when C54CM=0
*(CDP+AR0),when C54CM=1
```

当某一条指令与并行的指令共同使用系数操作数时，并行对的两条指令对Cmem的指针修改必须相同；否则，汇编器将产生错误。例如：

```
MAC *AR2+,*CDP+,AC0
::MAC *AR3+,*CDP+,AC1
```

⑤ 一个可选的mmr前缀用于指定间接存储器操作数，例如mmr(*AR0)。这是用户访问存储器映射寄存器的声明。汇编器会检查在某一环境下这样的访问是否合法。

(6) 操作数限定符

操作数限定符如表 3-6 所示,类似于对操作数的函数调用。需注意的是,uns 是无符号型操作数限定符,指令的后缀 U 意味着无符号型。在运算(MAC 的剩余部分中)限定操作数的时候,使用操作数限定符 uns。在整个运算都被影响时,使用指令后缀 U(如 MPYMU、CMPU、BCCU)。

表 3-6 有关的操作数限定符

限 定 符	含 义
dbl	访问一个真正的 32 位存储器操作数
dual	访问一个 32 位存储器操作数,它被用作某操作的两个独立的 16 位操作数
HI	访问累加器的高 16 位
high_byte	访问存储器位置的高字节
LO	访问累加器的低 16 位
low_byte	访问存储器位置的低字节
pair	双寄存器访问
rnd	舍入
saturate	饱和
uns	无符号操作数(不用于 MOV 指令)

当某一条指令和并行的指令一起使用某 Cmem 操作数,且此 Cmem 操作数被定义为无符号型(uns)时,并行对的两个 Cmem 操作数必须都定义为无符号型(反之亦然)。

当某一条指令和并行的指令一起使用 Xmem 和 Ymem 操作数,且 Xmem 操作数定义为无符号型(uns)时,Ymem 也必须定义为无符号型(反之亦然)。

3.1.5 并行特征和规则

1. 并行特征

C55x DSP 的结构允许用户在同一指令周期并行执行两条指令。并行类型如下所述。

(1) 单指令中的内嵌并行。

某些指令可以并行执行两个不同的操作。双冒号(::)用来分开这两个操作。这种并行也称为隐式并行。例如:

```
MPY *AR0,*CDP,AC0
::MPY *AR1,*CDP,AC1
```

这是一条单指令,AR0 和 AR1 引用的数据同时被 CDP 引用的系数乘。

(2) 用户定义的两条指令间的并行。

两条指令可以由用户或 C 编译器来并行。用平行竖线 ‖ 隔开两条并行执行的指令。例如:

```
MPYM *AR1-,*CDP,AC1
||XOR AR2,T1
```

第一条指令执行 D 单元中的乘运算，第二条指令执行 A 单元 ALU 中的逻辑运算。

(3) 内嵌并行可以与用户定义并行操作组合。例如：

```
MPYM T3= * AR3+,AC1,AC2
‖ MOV #5,AR1
```

第一条指令包含隐式并行，第二条指令是由用户定义的并行。

2. 并行基础

在并行指令中，遵循以下限制：两条指令的总长度不能超过 6 字节；不发生下一节中所述的资源冲突；一条指令必须有一个并行使能位或并行对进行软双重并行；没有存储器操作数可以使用 16 位或更大常数的寻址方式。

```
* abs16(#k16)
* (#k23)
port(#k16)
* ARn(k16)
* +ARn(k16)
* CDP(k16)
* +CDP(k16)
```

以下指令不能并行。

```
BCC P24,cond
CALLCC P24,cond
IDLE
INTR k5
RESET
TRAP k5
```

并行对中的两条指令都不可以使用如下指令或操作数限定符。

```
mmap()
port()
<instruction>.CR
<instruction>.LR
```

在流水线的每个阶段，某一特定的寄存器或存储器位置只能被写一次。违反此规则的形式有多种。一个简单的例子是装载同一个寄存器两次。其他例子包括：冲突的地址模式修改(如 * AR2+对 * AR2-)；与一条 SWAP 指令(修改该指令的所有寄存器)关联的，对相同寄存器有写操作的其他指令；修改数据堆栈指针(SP)或系统堆栈指针(SSP)不能与下列指令组合。

(1) 所有压入栈顶的指令(PUSH)。

(2) 所有弹出栈顶的指令(POP)。

(3) 所有条件调用(CALLCC)和无条件调用指令(CALL)。

(4) 所有条件返回(RETCC)、无条件返回(RET)和中断返回(RETI)指令。

(5) TRAP 和 INTR 指令。