第 3 章　TMS320C54x 指令系统

TMS320C54x 的汇编指令系统有两种形式：助记符形式和代数式形式，两种指令形式具有相同功能，本章介绍助记符指令系统。关于助记符指令，学习本章时，不必完全去记这些指令，只是大概了解有哪几类指令，常用指令的使用详见第 6 章软件开发调试实例。

当硬件执行指令时，寻找指令所指定的参与运算的操作数的方法就是寻址方式，解决参与运算的操作数从哪来？运算的结果放到哪里去？TMS320C54x DSP 提供了 7 种基本数据寻址方式，可以根据程序要求采用不同的寻址方式，以提高程序执行速度和代码效率。

- 寻址方式
- 指令系统

3.1　数据寻址方式

TMS320C54x DSP 提供以下 7 种基本数据寻址方式：

（1）立即数寻址：指令中有一个固定的立即数。

（2）绝对地址寻址：指令中有一个固定的地址（16 位）。

（3）累加器寻址：按累加器的内容作为地址去访问程序存储器中的一个单元。

（4）直接寻址：指令编码中含有的 7 位地址与 DP 或 SP 一起合成数据存储器中操作数的实际地址。

（5）间接寻址：通过辅助寄存器寻址。

（6）存储器映射寄存器寻址：修改存储器映射寄存器中的值，而不影响当前数据页面指针 DP 和当前堆栈指针 SP 的值。

（7）堆栈寻址：把数据压入或弹出系统堆栈。

表 3-1 列出寻址指令中用到的缩写符号及其含义。

表 3-1　寻址指令中用到的缩写符号及其含义

缩写符号	含义
Smem	16 位单数据存储器操作数
Xmem	在双操作数指令及某些单操作数指令中所用的 16 位双数据存储器操作数，从 DB 总线上读出

续表

缩写符号	含义
Ymem	在双操作数指令中所用的 16 位双数据存储器操作数，从 CB 总线上读出；在读同时并行写的指令中表示写操作数
dmad	16 位立即数——数据存储器地址（0～65535）
pmad	16 位立即数——程序存储器地址（0～65535）
PA	16 位立即数——I/O 口地址（0～65535）
src	源累加器（A 或 B）
dst	目的累加器（A 或 B）
lk	16 位长立即数

3.1.1 立即寻址

在立即寻址方式中，指令中包括了立即操作数。一条指令中可对两种立即数编码，一种是短立即数（3、5、8 或 9 位），另一种是 16 位的长立即数。短立即数指令编码为一个字长，16 位立即数的指令编码为两个字长。

立即数寻址指令中在数字或符号常数前面加一个“#”号来表示立即数，如：

```
LD   #0,ARP              ;ARP=0（#k3）
LD   #3,ASM              ;ASM=3（#k5）
LD   #50,DP              ;DP=50（#k9）
LD   #1234,A             ;A=1234（#lk）
```

3.1.2 绝对寻址

在绝对寻址方式中，指令中包含要寻址的存储单元的 16 位地址。有 4 种绝对寻址的指令：①数据存储器（dmad）寻址；②程序存储器（pmad）寻址；③端口地址（PA）寻址；④*(lk)寻址。

在绝对寻址方式中，指令中包含一个固定的 16 位地址，能寻址所有 64K 存储空间，但包含有绝对寻址的指令编码至少为两个字，因此运行速度慢，需要较大的存储空间。

1. 数据存储器（dmad）寻址

使用数据存储器寻址的指令有：

```
MVDK Smem, dmad          MVDM dmad, MMR
MVKD dmad, Smem          MVMD MMR, dmad
```

数据存储器寻址使用符号（符号地址）或一个表示 16 位地址的立即数来指明寻址的数据存储单元的 16 位绝对地址。例如：

```
MVKD   SAMPLE,*AR5;
```

将数据存储器 SAMPLE 地址单元的数据复制到由 AR5 所指的数据存储单元中。这里符号 SAMPLE 是程序中的标号或已经定义好的符号常数，代表数据存储单元的地址。又如：

```
MVKD 1000h, *AR5;
```

将数据存储器 1000h 单元的数据复制到由 AR5 所指的数据存储单元中。

2. 程序存储器（pmad）寻址

使用程序存储器寻址的指令有：

```
FIRS Xmem, Ymem, pmad          MACD Smem, pmad, src
```

MACP Smem, pmad, src　　MVDP Smem, pmad

MVPD pmad, Smem

程序存储器（pmad）寻址使用符号（符号地址）或一个表示 16 位地址的立即数来给出程序空间的地址。例如，把程序存储器中标号为 TABLE 单元中的值复制到 AR7 所指定的数据存储器中去，指令可写为：

MVPD　TABLE,*AR7;

可以将经常用的系数驻留在程序 ROM 存储器中，复位后利用该指令将系数传送到数据存储区，这样不用配置数据 ROM。

3．端口地址（PA）寻址

使用端口地址的指令有：

PORTR　PA, Smem

PORTW　Smem, PA

端口地址（PA）寻址使用一个符号（符号地址）或一个表示 16 位地址的立即数来给出外部 I/O 口地址。例如：

PORTR　FIFO,*AR5;

表示从 FIFO 单元端口读入一个数据，传送到由 AR5 所指的数据存储单元中。这里 FIFO 是一个 I/O 端口地址标号。

4．长立即数*(lk)寻址

长立即数*(lk)寻址用于所有支持单数据存储器操作数（Smem）的指令。

长立即数*(lk)寻址使用一个符号（符号地址）或一个表示 16 位地址的立即数来指定数据存储空间的一个地址。例如，把数据空间中地址为 BUFFER 单元中的数据传送到累加器 A，指令可写为：

LD　*(BUFFER),A

(lk)寻址不需要改变数据页指针 DP 的值和初始化任何一个 ARx 就可以对整个数据空间寻址。当采用绝对寻址时，指令编码要增加一个字，原来一个字的指令要变成两个字，而原来两个字的指令要变成三个字。注意：绝对寻址中采用(lk)形式的指令不能与循环指令（RPT 和 RPTZ）配合使用。

3.1.3　累加器寻址

累加器寻址是将累加器中的内容作为地址，用来对存放数据的程序存储器寻址。用于完成程序存储空间与数据存储空间之间的数据传输。

共有两条指令可以采用累加器寻址：

READA　Smem

WRITA　Smem

READA 是把累加器 A（低 16 位）所确定的程序存储器单元中的一个字传送到单数据存储（Smem）操作数所确定的数据存储器单元中。WRITA 是把 Smem 操作数所确定的数据单元中的一个字传送到累加器 A 确定的程序存储器单元中。在重复模式中，A 的内容自动增加。

3.1.4　直接寻址

在直接寻址方式中，指令中包含数据存储器地址（dma）的低 7 位，这 7 位 dma 作为地址偏移量，结合基地址（由数据页指针 DP 或堆栈指针 SP 给出）共同形成 16 位的数据存储器地

址。使用这种寻址方式，用户可在不改变 DP 或 SP 的情况下，对一页内的 128 个存储单元随机寻址。采用这种寻址方式的好处是指令为单字指令，数据存储器地址（dma）的低 7 位放在指令字中。

数据页指针 DP 和堆栈指针 SP 都可以用来与数据存储器地址低 7 位结合产生一个实际地址。ST1 中的直接寻址编辑方式位 CPL 用于选择 DP 或 SP 来产生地址。

CPL=0（通过 RSBX CPL 指令可以使 CPL=0），数据存储器地址（dma）的低 7 位与 DP 中的 9 位字段相连组成 16 位的数据存储器地址，如图 3-1 所示。

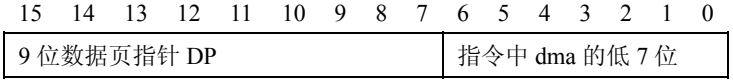


图 3-1　DP 作为基地址的直接寻址方式

CPL=1（通过 SSBX CPL 指令可以使 CPL=1），数据存储器地址（dma）的低 7 位与 SP 的 16 位地址相加形成 16 位的数据存储器地址，如图 3-2 所示。

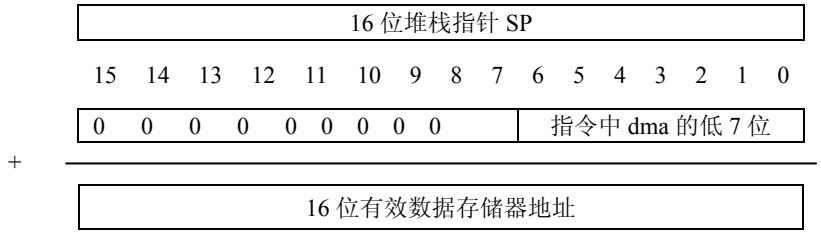


图 3-2　SP 作为基地址的直接寻址方式

将 64K 数据存储空间，分成 512 页，每页有 128 个单元，9 位的 DP 指向数据存储空间的 512 个数据页（0～511）中的一页，直接寻址使用一个符号或一个数给出 7 位地址偏移量，即指向该页中的某个单元。使用这种寻址方式时要注意对 DP 进行初始化，如下面一段程序所示：

```
.bss   x   128,1          ; 为 x 开辟 128 单元在同一页中
.text
LD   #0, A                ; 累加器 A 初始化为 0
LD   #x, DP               ; 对 DP 进行初始化，DP 指向 x 所在页
ADD   #1,A,A              ;
STL   A ,@ x              ; 把 A 中的值存放在 x 中
ADD   #1 ,A ,A            ;
STL   A, @x+128           ; 又回到页的开始（模 128）
```

直接寻址标识可以在变量前加@，如@x，或在偏移量前加@，如@5。其实@标识不是必须的，可有可无。

3.1.5　间接寻址

在间接寻址方式中，通过辅助寄存器中的 16 位地址可以访问 64K 数据空间的任何一个存储单元。间接寻址方式中使用两个辅助寄存器算术单元（ARAU0 和 ARAU1）和 8 个 16 位的辅助寄存器（AR0～AR7），进行无符号数算术运算，产生地址。

间接寻址有很大的灵活性，在单条指令中不仅可以读写数据存储器中的一个 16 位操作数，而且在单条指令中可以同时访问数据存储器空间的两个单元：读两个独立的存储器单元，读写

两个连续的存储器单元，读一个存储器单元和写另一个存储器单元。

1．单操作数寻址

表 3-2 列出了单数据存储器（Smem）操作数间接寻址类型。

表 3-2　单数据存储器操作数间接寻址类型

方式位（MOD）	操作数语法	功能	说明
0000 (0)	*ARx	地址=ARx	ARx 中的内容就是数据存储器的地址
0001 (1)	*ARx–	地址=ARx ARx= ARx-1	寻址结束后，ARx 中的地址减 1①
0010 (2)	*ARx+	地址=ARx ARx=Arx+1	寻址结束后，ARx 中的地址加 1①
0011 (3)	*+ARx	地址=ARx + 1 ARx=ARx + 1	ARx 中的地址加 1 后再寻址①②
0100 (4)	*ARx–0B	地址=ARx ARx=B(ARx–AR0)	寻址结束后，以位倒序借位的方式从 ARx 中减去 AR0 的值
0101 (5)	*ARx–0	地址=ARx ARx=ARx–AR0	寻址结束后，从 ARx 中减去 AR0 的值
0110 (6)	*ARx+0	地址=ARx ARx=ARx +AR0	寻址结束后，把 AR0 的值加到 ARx 中
0111 (7)	*ARx+0B	地址= ARx ARx=B(ARx + AR0)	寻址结束后，以位倒序进位的方式把 AR0 的值加到 ARx 中
1000 (8)	*ARx–%	地址=ARx ARx=circ(ARx – 1)	寻址结束后，ARx 中的地址按循环减的方法减 1①
1001 (9)	*ARx–0%	地址=ARx ARx=circ(ARx – AR0)	寻址结束后，按循环减的方法从 ARx 中减去 AR0 的值
1010 (10)	*ARx+%	地址=ARx ARx=circ(ARx + 1)	寻址结束后，ARx 中的地址按循环加的方法加 1①
1011 (11)	*ARx+0%	地址=ARx ARx= circ(ARx + AR0)	寻址结束后，按循环加的方法把 AR0 的值加到 ARx 中
1100 (12)	*ARx(lk)	地址=ARx + lk ARx=ARx	ARx 中的值加上 16 位长偏移（lk）的和作为地址，寻址结束后，ARx 中的值不变③
1101 (13)	*+ARx(lk)	地址=ARx + lk ARx=ARx + lk	将一个 16 位带符号数加至 ARx 后进行寻址③
1110 (14)	*+ARx(lk)%	地址=circ(ARx + lk) ARx=circ(ARx + lk)	将一个 16 位带符号数按循环加的方法加到 ARx，然后再寻址③
1111 (15)	*(lk)	地址= lk	利用 16 位无符号数作为地址寻址（相当于绝对寻址方式）③

注①：寻址 16 位字时增量/减量为 1，32 位字时增量/减量为 2。

注②：这种方式只能用于写操作指令。

注③：这种方式不允许对存储器映象寄存器寻址。

MOD = 0，1，2，3 为增 1、减 1 寻址方式，注意先增后寻址(*+ARx)只能用于写操作。

MOD = 12，13 为加偏移量寻址方式，前一种 ARx 的值不更新，对于后一种 ARx 的值更新。

MOD=5，6 为变址寻址方式，与加偏移量寻址方式相比，变址寻址方式不需要额外的偏移地址指令字，而且地址偏移量是在代码执行阶段确定的，因此可以调整变址步长。

特殊的间接寻址方式有循环寻址和位倒序寻址。下面分别加以介绍。

（1）循环寻址。在卷积、相关和 FIR 滤波等算法中，都要求在存储器中设置一个循环缓冲区。一个循环缓冲区就是一个包含了最新数据的滑动窗口，当新的数据来到时，新数据就会覆盖缓冲区中最早的数据。循环缓冲区实现的关键是循环寻址。C54x 间接寻址中以后缀“%”表示循环寻址的方式。使用循环寻址要遵循以下 3 条规则：

①大小为 R 的循环缓冲区必须从一个 N 位（N 是满足 2^N>R 条件的最小整数）边界开始。例如，R=31 转换为二进制为 011111，即 N=5，所以循环缓冲区必须从低 5 位为 0 的地址 XXXX XXXX XXX0 0000 开始，又如，R=32 转换为二进制为 100000，即 N=6，所以循环缓冲区必须从低 6 位为 0 的地址 XXXX XXXX XX00 0000 开始。

②循环缓冲区大小寄存器（BK）用于确定循环缓冲区的大小，用以下指令将 R=31 值加载到 BK 寄存器中：

```
STM   #31   BK
```

③要指定一个辅助寄存器 ARx 指向循环缓冲区的一个单元。

循环缓冲区的有效基地址（EFB）就是用户选定的辅助寄存器（ARx）的低 N 位置 0 后所得到的值。循环缓冲区的尾地址（EOB）是通过用 BK 的低 N 位代替 ARx 的低 N 位得到的。循环缓冲区的 INDEX 就是 ARx 的低 N 位，step 就是加到辅助寄存器或从辅助寄存器中减去的值，注意步长必须小于缓冲区的大小。循环寻址的算法如下：

```
if   0≤ index+step < BK
       index = index + step
else   if   index + step≥BK
       index = index + step - BK
else   if   index + step< 0
       index = index + step + BK
```

循环寻址可用于单数据存储器操作数寻址，也可用于双数据存储器操作数寻址。当 BK=0 时，ARx 的值不修正。表 3-2 中 MOD = 8，9，10，11，14 为循环寻址方式，不同的指令其步长和正负不同。

例如：若 BK=10，AR1=100H

```
LD   *+AR1(8)%, A          ;寻址 108H 单元
STL    A , *+AR1(8)%       ;寻址 106H 单元
```

（2）位倒序寻址。在 FFT 运算时，其输出、输入序列中必有其一要混序，所谓混序就是位倒序。对于 16 点 FFT 运算，其原序和位倒序寻址对应关系如表 3-3 所示。

表 3-3　位倒序寻址

原序		位倒序	
十进制数	二进制数	二进制数	十进制数
0	0000	0000	0
1	0001	1000	8
2	0010	0100	4
3	0011	1100	12

续表

原序		位倒序	
十进制数	二进制数	二进制数	十进制数
4	0100	0010	2
5	0101	1010	10
6	0110	0110	6
7	0111	1110	14
8	1000	0001	1
9	1001	1001	9
10	1010	0101	5
11	1011	1101	13
12	1100	0011	3
13	1101	1011	11
14	1110	0111	7
15	1111	1111	15

C54x 提供的位倒序寻址功能，提高了在 FFT 算法程序中使用存储器的效率及其执行速度。在这种寻址方式中，AR0 存放的整数 N 是 FFT 点数的一半。另外一个辅助寄存器指向数据存放的单元。当使用位倒序寻址把 AR0 加到辅助寄存器中时，地址以位倒序的方式产生，即进位是从左向右，而不是从右向左。间接寻址中*ARx+0B/-0B 表示位倒序寻址。

设 FFT 长度 N=16，则 AR0 赋值为 8，AR2 表示在存储区中数据的基地址$(01100000)_2$，位倒序方式读入数据情况如下：

*AR2+0B ; AR2 = 0110 0000　（第 0 个值）
*AR2+0B ; AR2 = 0110 1000　（第 1 个值）
*AR2+0B ; AR2 = 0110 0100　（第 2 个值）
*AR2+0B ; AR2 = 0110 1100　（第 3 个值）
*AR2+0B ; AR2 = 0110 0010　（第 4 个值）
*AR2+0B ; AR2 = 0110 1010　（第 5 个值）
*AR2+0B ; AR2 = 0110 0110　（第 6 个值）
*AR2+0B ; AR2 = 0110 1110　（第 7 个值）

2. 双操作数寻址

双操作数寻址用于执行两次读操作或一次读一次写并行存储指令中。在一个机器周期内通过两个 16 位数据总线（C 和 D）读两个操作数，Xmem 表示从 DB 总线上读出的 16 位操作数，Ymem 表示从 CB 总线上读出的 16 位操作数；或在一次读和一次写并行存储指令中，Xmem 表示读操作数，Ymem 表示写操作数。

双操作数间接寻址指令格式如图 3-3 所示。

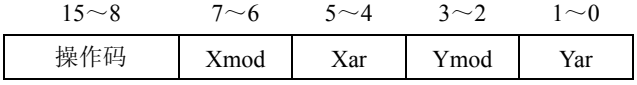


图 3-3　双操作数间接寻址指令格式

双操作数间接寻址指令代码都是 1 个字长，由于只有 2 位（Xar 或 Yar 的值）可以用于选择辅助寄存器，所以只能选择 4 个寄存器，所用辅助寄存器只能是 AR2、AR3、AR4、AR5。由于只有 2 位（Xmod 或 Ymod 的值）可以用于选则间接寻址类型，所以双数据存储器操作数间接寻址类型为*ARx、*ARx-、*ARx+、*ARx+0% 四种。

双操作数间接寻址特点是：占用程序空间小，运行速度快。

例如：

```
STM    #x , AR2
STM    #a , AR3
RPTZ    A , #3                  ;两个机器周期
MAC    *AR2+ , *AR3+,A          ;双操作数寻址，1 个机器周期
```

使用双操作数乘法指令和 RPT 指令，完成 N 项乘积求和的运算共需 N+2 个机器周期。

3.1.6 存储器映象寄存器寻址

存储器映象寄存器寻址用于修改存储器映象寄存器（MMR）中的内容，而不影响当前数据页指针 DP 和当前堆栈指针 SP。由于这种方式不需要修改 DP 和 SP，对寄存器的写操作开销最小。存储器映象寄存器寻址可用于直接寻址和间接寻址。

在直接寻址方式中，不管当前 DP 或 SP 的值如何，强制数据存储器地址的高 9 位为 0。利用指令中数据存储器地址（dma）的低 7 位访问 MMR，相当于基地址为 0 的直接寻址方式。

在间接寻址方式中，使用当前辅助寄存器的低 7 位作为地址访问 MMR。指令执行后，辅助寄存器中的高 9 位清为 0。例如，在存储器映象寄存器寻址方式中，用 AR1 指向存储器映象寄存器，它的值为 FF25h，由于 AR1 的低 7 位是 25h，因而它指向定时器周期寄存器 PRD。指令执行后 AR1 的值为 0025h。

只有 8 条指令能使用存储器映象寄存器寻址，即：

```
LDM    MMR, dst
MVDM   dmad, MMR
MVMD   MMR, dmad
MVMM   MMRx, MMRy
POPM   MMR
PSHM   MMR
STLM   src, MMR
STM    #lk, MMR
```

3.1.7 堆栈寻址

当发生中断或子程序调用时，系统堆栈自动保存 PC 值。堆栈也可以用于保存和传递其他数据。堆栈由高地址向低地址增长，处理器使用 16 位的存储器映象寄存器——堆栈指针（SP）对堆栈进行寻址，SP 总是指向压入堆栈的最后一个数据。

有 4 条使用堆栈寻址的指令：

- PSHD：把一个数据存储器数据压入堆栈；
- PSHM：把一个存储器映象寄存器中的值压入堆栈；
- POPD：从堆栈中弹出一个数据至数据存储器单元；
- POPM：从堆栈中弹出一个数据至存储器映象寄存器。

在压入堆栈操作时，SP 先减 1，然后将数据压入堆栈；在弹出堆栈操作时，数据从堆栈中弹出后，SP 再加 1。

3.2　指令系统中的符号和缩写

在介绍 C54x 指令系统之前，首先说明在指令系统的描述中使用的符号和缩写，如表 3-4 所示。

表 3-4　指令系统中的符号和缩写

符号	含义
A	累加器 A
ALU	算术逻辑单元
AR	辅助寄存器，泛指
ARx	指定某个特定的辅助寄存器（0≤x≤7）
ARP	ST0 中的 3 位辅助寄存器指针位，用 3 位表示当前的辅助寄存器
ASM	ST1 中的 5 位累加器移位方式位（-16≤ASM≤15）
B	累加器 B
BRAF	ST1 中的块重复指令有效标志
BRC	块重复计数器
BITC	4 位 BITC，决定位测试指令对指定的数据存储单元的哪一位（0≤BITC≤15）测试
C16	ST1 中的双 16 位/双精度运算方式位
C	ST0 中的进位位
CC	2 位条件码（0≤CC≤3）
CMPT	ST1 中的兼容方式位，决定 ARP 是否可以修正
CPL	ST1 中的编辑方式位
cond	表示一种条件的操作数，用于条件执行指令
[D]	延迟选项
DAB	D 地址总线
DAR	DAB 地址寄存器
dmad	16 位立即数数据存储器地址（0≤dmad≤65 535）
Dmem	数据存储器操作数
DP	ST0 中的 9 位数据存储器页指针（0≤DP≤511）
dst	目的累加器（A 或 B）
dst_	另一个目的累加器：如果 dst=A，则 dst_=B；如果 dst=B，则 dst_=A
EAB	E 地址总线
EAR	EAB 地址寄存器
extpmad	23 位立即数程序存储器地址
FRCT	ST1 中的小数方式位

续表

符号	含义
hi(A)	累加器 A 的高 16 位（位 31～16）
HM	ST1 中的保持方式位
IFR	中断标志寄存器
INTM	ST1 中的全局中断屏蔽位
K	少于 9 位的短立即数
k3	3 位立即数（0≤k3≤7）
k5	5 位立即数（-16≤k5≤15）
k9	9 位立即数（0≤k9≤511）
lk	16 位长立即数
Lmem	使用长字寻址的 32 位单数据存储器操作数
mmr,MMR	存储器映象寄存器
MMRx MMRy	存储器映象寄存器，AR0～AR7 或 SP
n	XC 指令后面的字数，n=1 或 n=2
N	RSBX 和 SSBX 指令中指定修改的状态寄存器：N=0，ST0；N=1，ST1
OVA	ST0 中累加器 A 的溢出标志
OVB	ST0 中累加器 B 的溢出标志
OVdst	目的累加器（A 或 B）的溢出标志
OVdst_	另一个目的累加器（A 或 B）的溢出标志
OVsrc	源累加器（A 或 B）的溢出标志
OVM	ST1 中的溢出方式位
PA	16 位立即数表示的端口地址（0≤PA≤65 535）
PAR	程序存储器地址寄存器
PC	程序计数器
pmad	16 位立即数程序存储器地址（0≤dmad≤65 535）
Pmem	程序存储器操作数
PMST	处理器方式状态寄存器
prog	程序存储器操作数
[R]	舍入选项
RC	重复计数器
REA	块重复结束地址寄存器
rnd	舍入
RSA	块重复起始地址寄存器
RTN	RETF[D]指令中用到的快速返回寄存器
SBIT	用 RSBX、SSBX 和 XC 指令所修改的指定状态寄存器的位号（0≤SBIT≤15）
SHFT	4 位移位数（0≤SHFT≤15）

续表

符号	含义
SHIFT	5 位移位数（-16≤SHIFT≤15）
Sind	间接寻址的单数据存储器操作数
Smem	16 位单数据存储器操作数
SP	堆栈指针
src	源累加器 A 或 B
ST0, ST1	状态寄存器 0，状态寄存器 1
SXM	ST1 中的符号扩展方式位
T	暂存器
TC	ST0 中的测试/控制标志位
TOS	堆栈顶部
TRN	状态转移寄存器
TS	由 T 寄存器的 5～0 位所规定的移位数
uns	无符号数
XF	ST1 中的 XF 引脚状态位
XPC	程序计数器扩展寄存器
Xmem	在双操作数指令以及某些单操作数指令中所用的 16 位双数据存储器操作数
Ymem	在双操作数指令中所用的 16 位双数据存储器操作数

3.3　指令系统

C54x 指令系统按功能可以分为 4 类：算术运算指令、逻辑运算指令、程序控制指令以及加载和存储指令。本节仅列出 4 类指令一览表供查阅，表中给出了助记符指令的语法、指令功能的表达式表示、指令功能的文字说明以及指令的字数和执行周期数，表中的指令字数和执行周期均为采用片内 DARAM 作为数据存储器。当使用长偏移间接寻址或以 Smem 绝对寻址时，应当增加 1 个字和 1 个机器周期。关于指令的使用及应用举例请参见第 6 章。

3.3.1　算术运算指令

C54x 的算术运算指令包括：加法指令、减法指令、乘法指令、乘累加指令与乘法减法指令、双字/双精度运算指令及专用指令。

1．加法指令

加法指令列在表 3-5 中。C54x 的加法指令共有 13 条，可完成两个操作数的加法运算、移位后的加法运算、带进位的加法运算和不带符号位扩展的加法运算。指令格式如下：

操作码　源操作数 [,移位数] , 目的操作数

① 操作码类型：ADD、ADDC、ADDM、ADDS。

ADD：不带进位；ADDC：带进位；ADDM：专用于立即数；ADDS：无符号数加法。

② 源操作数类型：Smem、Xmem、Ymem、#lk、src。

表 3-5　加法指令

语法	表达式	说明	字数	周期
ADD　Smem, src	src = src + Smem	操作数加至累加器	1	1
ADD　Smem, TS, src	src = src + Smem << TS	操作数移位后加至累加器	1	1
ADD　Smem, 16, src [, dst]	dst = src + Smem << 16	操作数左移 16 位加至累加器	1	1
ADD Smem [, SHIFT], src [, dst]	dst = src + Smem << SHIFT	操作数移位后加至累加器	2	2
ADD　Xmem, SHFT, src	src = src + Xmem <<SHFT	操作数移位后加至累加器	1	1
ADD　Xmem, Ymem, dst	dst = Xmem << 16 + Ymem << 16	两个操作数分别左移 16 位后加至累加器	1	1
ADD　# lk [, SHFT], src [, dst]	dst = src + #lk << SHFT	长立即数移位后加至累加器	2	2
ADD　# lk, 16, src [, dst]	dst = src + #lk << 16	长立即数左移 16 位后加至累加器	2	2
ADD　src [, SHIFT] [, dst]	dst = dst + src << SHIFT	累加器移位后相加	1	1
ADD　src, ASM [, dst]	dst = dst + src << ASM	累加器按 ASM 移位后相加	1	1
ADDC　Smem, src	src = src + Smem + C	操作数带进位加至累加器	1	1
ADDM　# lk, Smem	Smem = Smem + #lk	长立即数加至存储器	2	2
ADDS　Smem, src	src = src + uns(Smem)	无符号数加法，符号位不扩展	1	1

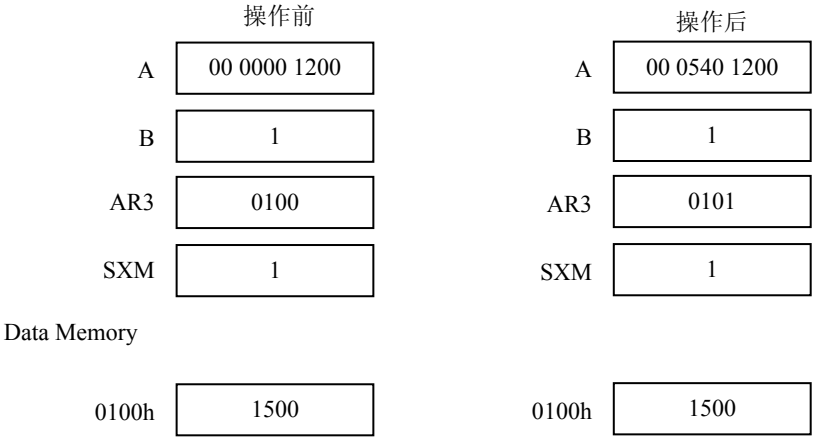
③移位数：TS、16、SHIFT、SHFT、ASM，正数左移，负数右移。

所移位数可由一个立即数（16、SHIFT、SHFT）、ST1 中的 ASM 位域、暂存器 T 中的（0～5）位（TS）决定。左移时低位添 0；右移时，如果 ST1 中的 SXM=1 高位进行符号扩展，如果 SXM=0 高位添 0。

④目的操作数：src、dst、Smem，指令中如果定义了 dst，结果放在 dst 中；否则，结果存在 src 中。

【例 3-1】ADD　*AR3+, 14, A

将 AR3 所指的数据存储单元内容，左移 14 位与 A 相加，结果放 A 中，AR3 加 1。



2．减法指令

加法指令列在表 3-6 中。加法指令共有 13 条。说明：

① SUBS　用于无符号数的减法运算；

SUBB　用于带借位的减法运算（如 32 位扩展精度的减法）；

SUBC　为条件减法。

② 使用 SUBC 重复 16 次减法，就可以完成除法功能，详见第 6 章。

表 3-6　减法指令

语法	表达式	说明	字数	周期
SUB Smem, src	src = src – Smem	从累加器中减去操作数	1	1
SUB Smem, TS, src	src = src – Smem << TS	从累加器中减去移位后的操作数	1	1
SUB Smem, 16, src [, dst]	dst = src – Smem << 16	从累加器中减去左移 16 位后的操作数	1	1
SUB Smem [,SHIFT],src [,dst]	dst = src – Smem << SHIFT	从累加器中减去移位后的操作数	2	2
SUB Xmem, SHFT, src	src = src – Xmem << SHFT	从累加器中减去移位后的操作数	1	1
SUB Xmem, Ymem, dst	dst = Xmem<<16–Ymem<<16	两个操作数分别左移 16 位后相减	1	1
SUB # lk [,SHFT], src[,dst]	dst = src – #lk << SHFT	长立即数移位后与累加器相减	2	2
SUB # lk, 16, src [, dst]	dst = src – #lk <<16	长立即数左移 16 位后与累加器相减	2	2
SUB src[, SHIFT] [, dst]	dst = dst – src << SHIFT	源累加器移位后和目的累加器相减	1	1
SUB src, ASM [, dst]	dst = dst – src << ASM	源累加器按 ASM 移位后与目的累加器相减	1	1
SUBB Smem, src	src = src – Smem – $\overline{C}$	从累加器中带借位减	1	1

SUBC Smem, src	If (src–Smem << 15)≥0
	src=(src–Smem<< 15)<<1+1
	Else
	src = src << 1
		条件减法指令	1	1

SUBS Smem, src	src = src – uns(Smem)	无符号数减法	1	1

3．乘法指令

乘法指令列在表 3-7 中。C54x 的指令系统提供了 10 条乘法运算指令，其运算结果都是 32 位的，存放在累加器 A 和 B 中。而参与运算的乘数可以是 T 寄存器、立即数、存储单元和累加器 A 或 B 的高 16 位。不同的乘法指令完成不同的功能：

MPY：普通乘指令；

MPYR：带四舍五入指令（加上 2^{15} 再对 15～0 位清零）；

MPYA：A 累加器高端参与乘法；
MPYU：无符号乘；
SQUR：平方。

表 3-7　乘法指令

语法	表达式	说明	字数	周期
MPY　Smem, dst	dst = T * Smem	T 寄存器值和操作数相乘	1	1
MPYR　Smem, dst	dst = rnd(T * Smem)	T 寄存器值和操作数相乘（带舍入）	1	1
MPY Xmem, Ymem, dst	dst = Xmem * Ymem, T = Xmem	两个操作数相乘，乘积存放在累加器中	1	1

MPY Smem, # lk, dst	dst = Smem * #lk
	T = Smem
		长立即数与操作数相乘	2	2

MPY # lk, dst	dst = T * #lk	长立即数与 T 寄存器值相乘	2	2
MPYA dst	dst = T * A(32– 16)	T 寄存器值与累加器 A 的高位相乘	1	1
MPYA Smem	B = Smem * A(32– 16), T = Smem	操作数与累加器 A 的高位相乘	1	1
MPYU Smem, dst	dst = uns(T) * uns(Smem)	无符号数乘法	1	1
SQUR Smem, dst	dst = Smem * Smem, T = Smem	操作数的平方	1	1
SQUR A, dst	dst = A(32– 16) * A(32– 16)	累加器 A 的高位平方	1	1

4．乘法－累加和乘法－减法指令

乘法－累加和乘法－减法指令列在表 3-8 中。这类指令共计 22 条，除了完成乘法运算外，还具有加法或减法运算。因此，在一些复杂的算法中，可以大大提高运算速度。

表 3-8　乘加和乘减指令

语法	表达式	说明	字数	周期
MAC Smem, src	src = src + T * Smem	操作数与 T 寄存器值相乘后加到累加器	1	1
MAC Xmem, Ymem, src [,dst]	dst= src + Xmem * Ymem T = Xmem	两个操作数相乘加到累加器中	1	1
MAC # lk, src [, dst]	dst = src + T * #lk	长立即数与 T 寄存器值相乘后加到累加器	2	2

MAC Smem, # lk, src [,dst]	dst = src + Smem * #lk
	T = Smem
		长立即数与操作数相乘后加到累加器	2	2

MACR Smem, src	src = rnd(src + T * Smem)	操作数与 T 寄存器值相乘后加到累加器（带舍入[①]）	1	1
MACR Xmem, Ymem, src [,dst]	dst= rnd(src + Xmem * Ymem) T = Xmem	两个操作数相乘加到累加器中（带舍入[①]）	1	1

续表

语法	表达式	说明	字数	周期
MACA Smem [, B]	B = B + Smem * A(32–16) T = Smem	操作数与累加器 A 高位相乘后加到累加器 B	1	1
MACA T, src [, dst]	dst = src + T * A(32–16)	T 寄存器值与累加器 A 高位相乘	1	1
MACAR Smem [, B]	B =rnd(B + Smem * A(32–16)) T = Smem	T 寄存器值与累加器 A 高位相乘后加到累加器中 B（带舍入①）	1	1
MACAR Smem [, B]	dst = rnd(src + T * A(32–16))	T 寄存器值与累加器 A 高位相乘后加到源累加器（舍入①）	1	1
MACD Smem, pmad, src	Src = src + Smem * pmad T =Smem, (Smem+ 1) = Smem	操作数与程序存储器值相乘后加到累加器并延迟	2	3
MACP Smem, pmad, src	src = src + Smem * pmad T = Smem	操作数与程序存储器值相乘后加到累加器	2	3
MACSU Xmem, Ymem, src	src =src+ uns(Xmem) * Ymem T = Xmem	无符号数和有符号数相乘后加到累加器	1	1
MAS Smem, src	src = src – T * Smem	从累加器中减去操作数与 T 寄存器值的乘积	1	1
MASR Smem, src	src = rnd(src – T * Smem)	从累加器中减去操作数与 T 寄存器乘积（带舍入①）	1	1
MAS Xmem, Ymem, src [, dst]	dst = src – Xmem * Ymem T = Xmem	从源累加器中减去两个操作数乘积	1	1
MASR Xmem, Ymem, src [, dst]	dst =rnd(src – Xmem * Ymem) T = Xmem	从源累加器中减去两个操作数乘积（带舍入①）	1	1
MASA Smem [, B]	B = B – Smem * A(32–16) T = Smem	从累加器 B 中减去操作数与累加器 A 高位乘积	1	1
MASA T, src [, dst]	dst = src – T * A(32–16)	从源累加器中减去 T 寄存器值与累加器 A 高位乘积	1	1
MASAR T, src [, dst]	dst = rnd(src–T * A(32–16))	从源累加器中减去 T 寄存器值与累加器 A 高位乘积（带舍入①）	1	1
SQURA Smem, src	src = src + Smem * Smem T = Smem	操作数平方并累加	1	1
SQURS Smem, src	src = src – Smem * Smem T = Smem	从累加器中减去操作数平方	1	1

注：①带有 R 后缀，对乘法结果进行舍入处理（加 2^{15}，低 16 位清 0）。

参与运算的乘数可以是 T 寄存器、立即数、存储单元和累加器 A 或 B 的高 16 位。乘法

运算结束后，再将乘积与目的操作数进行加法或减法运算。

5．双精度/双字算术运算指令

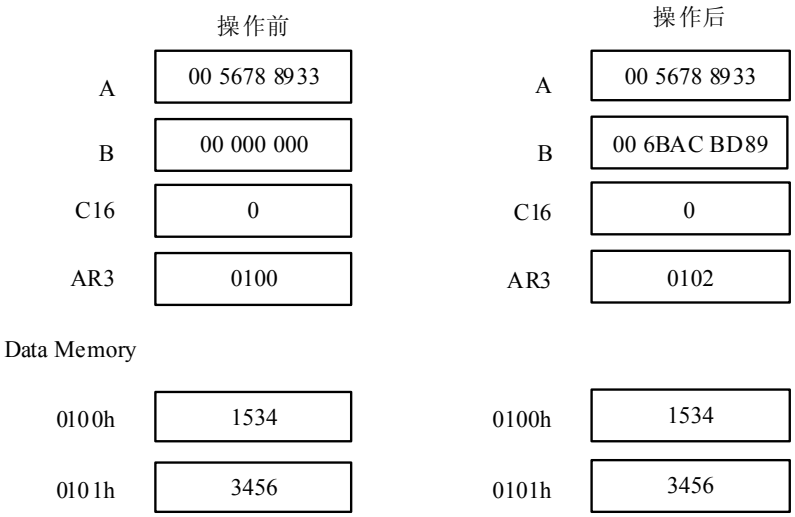
双精度/双字算术运算指令共计 6 条，列在表 3-9 中。C16=0，表中指令以双精度方式（32 位）执行；C16=1，表中指令以双 16 位方式（同时进行 2 次 16 位数的加或减）执行。

表 3-9　双精度（32 位操作数）指令

语法	表达式	说明	字数	周期
DADD Lmem,src[, dst]	If　C16 = 0 dst = Lmem + src If　C16 = 1 dst(39–16)=Lmem(31–16)+src(31–16) dst(15–0)= Lmem(15–0) +src(15–0)	双精度/双 16 位操作数加到累加器	1	1
DADST Lmem, dst	If　C16 = 0 dst = Lmem + (T << 16 + T) If　C16 = 1 dst(39–16) = Lmem(31–16) + T dst(15–0) = Lmem(15–0) – T	双精度/双 16 位操作数与 T 寄存器值相加/减	1	1
DRSUB Lmem, src	If　C16 = 0 src = Lmem – src If　C16 = 1 src(39–16)=Lmem(31–16)–src(31–16) src(15– 0) = Lmem(15–0)–src(15–0)	从双精度/双 16 位操作数中减去累加器值	1	1
DSADT Lmem, dst	If　C16 = 0 dst = Lmem – (T << 16 + T) If　C16 = 1 dst(39– 16) = Lmem(31– 16) – T dst(15– 0) = Lmem(15– 0) + T	长操作数与 T 寄存器值相加/减	1	1
DSUB Lmem, src	If　C16 = 0 src = src – Lmem If　C16 = 1 src(39–16)=src(31–16)–Lmem(31– 16) src(15– 0) = src(15– 0) – Lmem(15– 0)	从累加器中减去双精度/双 16 位操作数	1	1
DSUBT Lmem, dst	If　C16 = 0 dst = Lmem – (T << 16 + T) If　C16 = 1 dst(39– 16) = Lmem(31– 16) – T dst(15– 0) = Lmem(15– 0) – T	从长操作数中减去 T 寄存器值	1	1

指令中 Lmem 为 32 位长操作数，有两个连续地址的 16 位字构成，低地址为偶数，其内容为操作数的高 16 位，高地址的内容为操作数的低 16 位。

【例 3-2】DADD　*AR3+, A, B



6．专用指令

专用指令有 15 条，列在表 3-10 中。

表 3-10　专用指令

语法	表达式	说明	字数	周期
ABDST Xmem, Ymem	B = B + |A(32– 16)| A = (Xmem – Ymem) << 16	Xmem 和 Ymem 之差的绝对值	1	1
ABS src [, dst]	dst = |src|	累加器取绝对值	1	1
CMPL src [, dst]	dst = ~src	计算 src 的反码（逻辑反）	1	1
DELAY Smem	(Smem + 1) = Smem	把单数据存储单元 Smem 中的内容复制到紧接着的较高地址单元中	1	1
EXP src	T =源累加器符号位数–8	计算累加器指数值①	1	1
FIRS Xmem, Ymem, pmad	B = B + A * pmad A = (Xmem + Ymem) << 16	系数对称 FIR 滤波器	2	3
LMS Xmem, Ymem	B = B + Xmem * Ymem A = A + Xmem << 16 + 2^{15}	求最小均方值	1	1
MAX dst	dst = max(A, B)	比较累加器值，把较大值放到 dst 中	1	1
MIN dst	dst = min(A, B)	比较累加器值，把小值放到 dst 中	1	1
NEG src [, dst]	dst = –src	计算累加器值反值	1	1
NORM src [, dst]	dst = src << TS dst = norm(src, TS)	规一化。按 T 寄存器中的内容对累加器进行规格化处理（左移或右移）	1	1
POLY Smem	B = Smem << 16 A = rnd(A(32– 16) * T + B)	用于多项式计算	1	1
RND src [, dst]	dst = src + 2^{15}	累加器舍入计算	1	1
SAT src	Src 的饱和计算	累加器饱和计算	1	1
SQDST Xmem, Ymem	B = B + A(32– 16) * A(32– 16) A = (Xmem – Ymem) << 16	计算两点之间距离的平方	1	1

3.3.2 逻辑运算指令

逻辑指令包括与、或、异或（按位）、移位和测试指令，分别列在表 3-11 至表 3-15 中。

表 3-11 与逻辑运算指令

语法	表达式	说明	字数	周期
AND Smem, src	src = src & Smem	操作数和累加器相与	1	1
AND # lk [, SHFT], src [, dst]	dst = src & #lk <<SHFT	长立即数移位后和累加器相与	2	2
AND # lk, 16, src [, dst]	dst = src & #lk <<16	长立即数左移 16 位后和累加器相与	2	2
AND src [, SHIFT] [, dst]	dst = dst & src<<SHIFT	源累加器移位后和目的累加器相与	1	1
ANDM # lk, Smem	Smem = Smem & #lk	操作数和长立即数相与	2	2

注：如果有移位，操作数在移位后再进行与操作。左移时低位添 0；右移时高位添 0。不受 SXM 影响。

表 3-12 或逻辑运算指令

语法	表达式	说 明	字数	周期
OR Smem, src	src = src | Smem	操作数和累加器相或	1	1
OR # lk [, SHFT], src[,dst]	dst = src | #lk << SHFT	长立即数移位后和累加器相或	2	2
OR # lk, 16, src [, dst]	dst = src | #lk << 16	长立即数左移 16 位后和累加器相或	2	2
OR src [, SHIFT] [, dst]	dst = dst | src << SHIFT	源累加器移位后和目的累加器相或	1	1
ORM # lk, Smem	Smem = Smem | #lk	操作数和长立即数相或	2	2

表 3-13 异或逻辑运算指令

语法	表达式	说明	字数	周期
XOR Smem, src	src = src∧Smem	操作数和累加器相异或	1	1
XOR #lk [,SHFT,],src[,dst]	dst = src∧#lk << SHFT	长立即数移位后和累加器相异或	2	2
XOR # lk, 16, src [, dst]	dst = src∧#lk << 16	长立即数左移 16 位后和累加器异或	2	2
XOR src [, SHIFT] [, dst]	dst = ds∧src <<SHIFT	源累加器移位后和目的累加器异或	1	1
XORM # lk, Smem	Smem = Smem∧#lk	操作数和长立即数相异或	2	2

表 3-14 移位逻辑运算指令

语法	表达式	说明	字数	周期
ROL src	Rotate left with carry in	累加器循环左移一位。进位位 C 的值移入 src 的最低位，src 的最高位移入 C 中，保护位清 0	1	1
ROLTC src	Rotate left with TC in	累加器带 TC 位循环左移。TC 的值移入 src 的最低位，src 的最高位移入 C 中，保护位清 0	1	1

续表

语法	表达式	说明	字数	周期
ROR src	Rotate right with carry in	累加器循环右移一位。进位位 C 的值移入 src 的最高位，src 的最低位移入 C 中，保护位清 0	1	1
SFTA src, SHIFT[, dst]	dst=src<< SHIFT {arithmetic shift}	累加器算术移位①	1	1
SFTC src	if src(31) = src(30) then src = src << 1	累加器条件移位。当累加器的第 31 位、30 位都为 1 或为 0 时(两个符号位)，累加器左移一位，TC=0；否则（一个符号位），TC=1	1	1
SFTL src, SHIFT [, dst]	dst = src << SHIFT {logical shift}	累加器逻辑移位①	1	1

注：①在执行 SFTA 和 SFTL 指令时，移位数定义为-16≤SHFIT≤15。SFTA 指令受 SXM 位（符号位扩展方式位）影响。当 SHFIT 为负值时，如果 SXM=1，SFTA 进行算术右移（累加器的 39～0 位进行移位），并保持累加器的符号位（把 src(39)位移到 dst(39-(39+(SHIFT + 1)))；当 SXM=0 时，累加器的最高位添 0。当 SHFIT 大于 0 时，SFTA 进行算术左移，把 0 写进 dst((SHIFT–1)–0)。SFTL 指令不受 SXM 位影响，它对累加器的 31～0 位进行移位操作，移位时将 0 移到最高有效位 MSB 或最低有效位 LSB（取决于移位的方向）。

表 3-15　测试指令

语法	表达式	说明	字数	周期
BIT Xmem, BITC	TC = Xmem(15–BITC)	测试指定位。把 Xmem 存储单元值的第 15-BITC 位复制到 ST0 的 TC 位。 例如 BIT *AR5+, 15-12 测试 AR5 所指单元的第 12 位	1	1
BITF Smem, # lk	TC = (Smem && #lk)	测试由立即数规定的位域。lk 常数在测试 bit 或 bits 是起屏蔽作用。如果所测试的 bit 或 bits 为 0，TC 位清 0；否则，TC 置 1	2	2
BITF Smem	TC = Smem(15–T(3– 0))	测试由 T 寄存器规定的位。把 Smem 存储单元值的第 15- T(3–0)位复制到 ST0 的 TC 位	1	1
CMPM Smem, # lk	TC = (Smem == #lk)	存储单元和长立即数比较，如果相等 TC 置 1，否则清 0	2	2
CMPR CC, ARx	Compare ARx with AR0	辅助寄存器 ARx 内容与 AR0 内容比较。比较由条件 CC（条件代码）值决定。如果满足条件，TC 置 1，否则清 0。CC 值及其表示的条件和说明如下 条件 / CC 值 / 说明 EQ / 00 / 测试 ARx 是否等于 AR0 LT / 01 / 测试 ARx 是否小于 AR0 GT / 10 / 测试 ARx 是否大于 AR0 NEQ / 11 / 测试 ARx 是否不等于 AR0	1	1

3.3.3 程序控制指令

程序控制指令包括分支转移指令、子程序调用指令、中断指令、返回指令、重复指令、堆栈操作指令及混合程序控制指令，分别列在表 3-16 至表 3-22 中。

表 3-16 分支指令

语法	表达式	说明	字数	周期（非延迟/延迟）
B[D] pmad	PC = pmad(15–0)	无条件分支转移	2	4/2
BACC[D] src	PC = src(15–0)	用指定的累加器（A 或 B）的低 16 位作为地址转移	1	6/4
BANZ[D] pmad, Sind	if (Sind≠0) then PC = pmad(15–0)	辅助寄存器内容不为 0，转移到指定程序地址	2	4/2 条件满足 2/2 不满足
BC[D] pmad,cond [, cond [, cond]]	if (cond(s)) then PC = pmad(15–0)	条件分支转移	2	5/3 条件满足 3/3 不满足
FB[D] extpmad	PC = pmad(15–0) XPC = pmad(22–16)	无条件远程分支转移	2	4/2
FBACC[D] src	PC = src(15–0) XPC = src(22–16)	按累加器规定的地址远程分支转移	1	6/4

注：后缀 D 表示延迟分支转移，可以使一条指令的执行时间减少 2 个周期，先执行分支指令后续的 2 条指令，然后再分支转移。

表 3-17 调用指令

语法	表达式	说明	字数	周期（非延迟/延迟）
CALA[D] src	--SP, PC + 1[3] = TOS PC = src(15– 0)	按累加器规定的地址调用子程序	1	6/4
CALL[D] pmad	--SP, PC + 2[4] = TOS PC = pmad(15– 0)	无条件调用子程序	2	4/2
CC[D] pmad, cond [, cond [, cond]]	if (cond(s)) then —SP PC + 2[4] = TOS PC = pmad(15– 0)	条件调用子程序	2	5/3 条件满足 3/3 不满足
FCALA[D] src	--SP, PC + 1[3] = TOS PC = src(15–0) XPC = src(22–16)	按累加器规定的地址远程调用子程序	1	6/4
FCALL[D] extpmad	--SP, PC + 2[4] = TOS PC = pmad(15–0) XPC = pmad(22–16)	无条件远程调用子程序	2	4/2

表 3-18　中断指令

语法	表达式	说明	字数	周期
INTR K	--SP , ++PC = TOS PC = IPTR(15–7) + K << 2 INTM = 1	不可屏蔽的软件中断 关闭其它可屏蔽中断	1	3
TRAP K	--SP , ++PC = TOS PC = IPTR(15–7) + K << 2	不可屏蔽的软件中断 不影响 INTM 位	1	3

表 3-19　返回指令

语法	表达式	说明	字数	周期 （非延迟/延迟）
FRET[D]	XPC = TOS, ++ SP, PC = TOS,++SP	远程返回	1	6/4
FRETE[D]	XPC = TOS, ++ SP, PC = TOS, ++SP, INTM = 0	开中断，从远程返回	1	6/4
RC[D] cond [, cond [, cond]]	if (cond(s)) then PC = TOS, ++SP	条件返回	1	5/3 条件满足 3/3 不满足
RET[D]	PC = TOS, ++SP	返回	1	5/3
RETE[D]	PC = TOS, ++SP, INTM = 0	开中断，从中断返回	1	5/3
RETF[D]	PC = RTN, ++SP, INTM = 0	开中断，从中断快速返回	1	3/1

表 3-20　重复指令

语法	表达式	说明	字数	周期
RPT Smem	Repeat single, RC = Smem	重复执行下一条指令（Smem）+1 次	1	1
RPT # K	Repeat single, RC = #K	重复执行下一条指令 k+1 次	1	1
RPT # lk	Repeat single, RC = #lk	重复执行下一条指令 lk+1 次	2	2
RPTB[D]　pmad	Repeat block, RSA=PC+ 2[4], REA = pmad, BRAF = 1	块重复指令	2	4/2
RPTZ　dst,　# lk	Repeat single, RC = #lk, dst = 0	重复执行下一条指令 lk+1 次，目的累加器清 0	2	2

表 3-21　堆栈操作指令

语法	表达式	说明	字数	周期
FRAME K	SP = SP + K	把短立即数 K 加到 SP 中	1	1
POPD Smem	Smem = TOS, ++SP	把栈顶数据弹出到 Smem 数据存储单元中，然后 SP 加 1	1	1
POPM MMR	MMR = TOS, ++SP	把栈顶数据弹出到 MMR，然后 SP 加 1	1	1
PSHD Smem	--SP, Smem = TOS	SP 减 1 后，将数据压入堆栈	1	1
PSHM MMR	--SP, MMR = TOS	SP 减 1 后，将 MMR 压入堆栈	1	1

表 3-22　混合程序控制指令

语法	表达式	说明	字数	周期
IDLE K	idle(K)	保持空转状态，直到非屏蔽中断和复位①	1	4
MAR Smem	If CMPT = 0, then modify ARx If CMPT = 1 and ARx ≠AR0, then modify ARx, ARP = x If CMPT = 1 and ARx = AR0, then modify AR(ARP)	修改辅助寄存器的值。CMPT=1，修改 ARx 的内容及修改 ARP 值为 x；CMPT=0，只修改 ARx 的内容，而不改变 ARP 值	1	1
NOP	no operation	空操作，除了执行 PC+1 外不执行任何操作	1	1
RESET	software reset	非屏蔽的软件复位	1	3
RSBX N, SBIT	STN (SBIT) = 0	对状态寄存器 ST0、ST1 的特定位清 0。N 指定修改的状态寄存器，SBIT 指定修改的位。状态寄存器中的域名能够用来代替 N 和 SBIT 作为操作数	1	1
SSBX N, SBIT	STN (SBIT) = 1	对状态寄存器 ST0、ST1 的特定位置 1	1	1
XC n, cond [,cond[,cond]]	If (cond(s)) then execute the next n instructions; n = 1 or 2	条件执行指令	1	1

注：①K=1，诸如定时器、串口等片内外设在空闲状态时仍有效，外围中断与复位操作及外部中断能使处理器从空闲状态中解放出来；K=2，诸如定时器、串口等片内外设在空闲状态时无效，复位操作及外部中断能使处理器从空闲状态中解放出来；K=3，诸如定时器、串口等片内外设在空闲状态时无效，锁相环 PLL 被停止，复位操作及外部中断能使处理器从空闲状态中解放出来。该指令不能重复执行。

3.3.4　加载和存储指令

加载和存储指令包括：加载指令、存储指令、条件存储指令、并行加载和存储指令、并行加载和乘法指令、并行存储和加/减法指令、混合加载和存储指令，分别列在表 3-23 至表 3-30 中。加载指令是将存储器内容或立即数赋给目的寄存器；存储指令是把源操作数或立即数存入存储器或寄存器。

表 3-23　载指令

语法	表达式	说明	字数	周期
DLD Lmem, dst	dst = Lmem	双精度/双 16-Bit 长字加载目的累加器	1	1
LD Smem, dst	dst = Smem	把数据存储器操作数加载到累加器	1	1
LD Smem, TS, dst	dst = Smem << TS	操作数按 TREG（5～0）移位后加载到累加器	1	1
LD Smem, 16 , dst	dst = Smem << 16	操作数左移 16 位后加载累加器	1	1
LD Smem[,SHIFT],dst	dst = Smem << SHIFT	操作数 Smem 移位后加载累加器	2	2
LD Xmem, SHFT, dst	dst = Xmem << SHFT	Xmem 移位后加载累加器	1	1
LD #K, dst	dst = #K	短立即数 K 加载累加器	1	1
LD #lk [,SHFT],dst	dst = #lk << SHFT	长立即数移位后加载累加器	2	2

续表

语法	表达式	说明	字数	周期
LD #lk, 16, dst	dst = #lk << 16	长立即数左移 16 位后加载累加器	2	2
LD src, ASM [,dst]	dst = src << ASM	源累加器按 ASM 移位后加载目的累加器	1	1
LD src [,SHIFT],dst	dst = src << SHIFT	源累加器移位后加载目的累加器	1	1
LD Smem, T	T = Smem	操作数加载 T 寄存器	1	1
LD Smem, DP	DP = Smem(8–0)	9 位操作数加载 DP	1	3
LD #k9, DP	DP = #k9	9 位立即数加载 DP	1	1
LD #k5, ASM	ASM = #k5	5 位立即数加载 ASM	1	1
LD # k3, ARP	ARP = #k3	3 位立即数加载 ARP	1	1
LD Smem, ASM	ASM = Smem(4–0)	操作数低 5 位加载 ASM	1	1
LDM MMR, dst	dst = MMR	将 MMR 加载到目的累加器	1	1
LDR Smem, dst	dst(31–16)=rnd(Smem)	操作数舍入加载累加器高位	1	1
LDU Smem, dst	dst = uns(Smem)	无符号操作数加载 dst 低端（15～0），dst 保护位和高端（39～16）清 0	1	1
LTD Smem	T = Smem, (Smem + 1) = Smem	操作数加载到 T 寄存器和紧跟着的较高地址的数据单元	1	1

表 3-24　存储指令

语法	表达式	说明	字数	周期
DST src, Lmem	Lmem = src	累加器值存入长字单元中	1	2
ST T, Smem	Smem = T	存储 T 寄存器值	1	1
ST TRN, Smem	Smem = TRN	存储 TRN 寄存器值	1	1
ST # lk, Smem	Smem = #lk	存储长立即数	2	2
STH src, Smem	Smem = src<< –16	存储累加器高位（31～16）	1	1
STH src, ASM, Smem	Smem = src<<(ASM–16)	累加器按 ASM 移位后存储累加器高位	1	1
STH src, SHFT, Xmem	Xmem = src<<(SHFT–16)	累加器按 SHFT 移位后存储累加器高位	1	1
STH src[,SHIFT],Smem	Smem = src <<(SHIFT–16)	累加器按 SHIFT 移位后存储高位	2	2
STL src, Smem	Smem = src	存储累加器低位（15～0）	1	1
STL src, ASM, Smem	Smem = src << ASM	累加器按 ASM 移位后存储累加器低位	1	1
STL src, SHFT, Xmem	Xmem = src << SHFT	累加器按 SHFT 移位后存储累加器低位	1	1
STL src[,SHIFT],Smem	Smem = src << SHIFT	累加器按 SHIFT 移位后存储累加器低位	2	2
STLM src, MMR	MMR = src	累加器低位存储到 MMR	1	1
STM # lk, MMR	MMR = #lk	长立即数 lk 存储到 MMR	2	2

表 3-25　条件存储指令

语法	表达式	说明	字数	周期
CMPS src, Smem	If src(31–16) > src(15–0) Then Smem = src(31–16) If src(31– 16)≤src(15–0) Then Smem = src(15–0)	比较源累加器的高 16 位和低 16 位，把较大值存入单数据存储单元	1	1
SACCD src, Xmem, cond	If (cond) Xmem = src << (ASM–16)	如果条件（cond）满足，累加器按 ASM 移位后的高位存储到 Xmem 单元	1	1
SRCCD Xmem, cond	If (cond) Xmem = BRC	如果条件（cond）满足，把块循环计数器 BRC 中的值存储到 Xmem 单元	1	1
STRCD Xmem, cond	If (cond) Xmem = T	如果条件（cond）满足，把 TREG 中的值存储到 Xmem 单元	1	1

表 3-26　并行加载和存储指令

语法	表达式	说明	字数	周期
ST src, Ymem || LD Xmem, dst	Ymem = src << (ASM-16) || dst = Xmem << 16	源累加器按 ASM 移位后高位存储到 Ymem 单元中，同时并行执行把 Xmem 单元中值加载到目的累加器高位	1	1
ST src, Ymem || LD Xmem, T	Ymem = src << (ASM–16) || T = Xmem	源累加器按 ASM 移位后高位存储到 Ymem 单元中，同时并行执行把 Xmem 单元中值加载到 T 寄存器	1	1

表 3-27　并行加载和乘法指令

语法	表达式	说明	字数	周期
LD Xmem, dst || MAC Ymem, dst_	dst = Xmem << 16 || dst_ = dst_ + T * Ymem	双数据存储器操作数左移 16 位加载累加器高位，并行乘累加运算	1	1
LD Xmem, dst || MACR Ymem, dst_	dst = Xmem << 16 || dst_=rnd(dst_+T* Ymem)	双数据存储器操作数左移 16 位加载累加器高位，并行乘累加运算（带舍入）	1	1
LD Xmem, dst || MAS Ymem, dst_	dst = Xmem << 16 || dst_= dst_–T * Ymem	双数据存储器操作数左移 16 位加载累加器高位，并行乘法减法运算	1	1
LD Xmem, dst || MASR Ymem, dst_	dst = Xmem << 16 ||dst_=rnd(dst_–T*Ymem)	双数据存储器操作数左移 16 位加载累加器高位，并行乘法减法运算（带舍入）	1	1

表 3-28　并行存储和加/减法指令

语法	表达式	说明	字数	周期
ST src, Ymem || ADD Xmem, dst	Ymem = src << (ASM-16) || dst = dst_ + Xmem<<16	按 ASM 移位后存储累加器高位并行加法运算	1	1
ST src, Ymem || SUB Xmem, dst	Ymem = src << (ASM–16) ||dst =(Xmem << 16)–dst_	按 ASM 移位后存储累加器高位并行减法运算	1	1

表 3-29　并行存储和乘法指令

语法	表达式	说明	字数	周期
ST src, Ymem || MAC Xmem, dst	Ymem = src << (ASM – 16) || dst = dst + T * Xmem	按 ASM 移位后存储累加器高位并行乘法累加运算	1	1
ST src, Ymem || MACR Xmem, dst	Ymem = src << (ASM – 16) || dst = rnd(dst + T * Xmem)	按 ASM 移位后存储累加器高位并行乘法累加运算（带舍入）	1	1
ST src, Ymem || MAS Xmem, dst	Ymem = src << (ASM – 16) || dst = dst – T * Xmem	按 ASM 移位后存储累加器高位并行乘法减法运算	1	1
ST src, Ymem || MASR Xmem, dst	Ymem = src << (ASM – 16) || dst = rnd(dst – T * Xmem)	按 ASM 移位后存储累加器高位并行乘法减法运算（带舍入）	1	1
ST src, Ymem || MPY Xmem, dst	Ymem = src << (ASM – 16) || dst = T * Xmem	按 ASM 移位后存储累加器高位并行乘法运算	1	1

表 3-30　混合加载和存储指令（数据块传送指令）

语法	表达式	说明	字数	周期
MVDD Xmem, Ymem	Ymem = Xmem	数据存储器内部传送数据。Xmem 存储单元值复制到 Ymem 存储单元中	1	1
MVDK Smem,dmad	dmad = Smem	数据存储器内部指定地址传送数据。把单数据存储器操作数寻址的 Smem 单元内容复制到由 16-Bit 立即数 dmad 寻址的数据存储单元中	2	2
MVDM dmad, MMR	MMR = dmad	把由 16-Bit 立即数 dmad 寻址的数据存储单元内容复制到 MMR	2	2
MVDP Smem, pmad	pmad = Smem	数据存储器向程序存储器传送数据。把单数据存储器操作数寻址的 Smem 单元内容复制到由 16-Bit 立即数 pmad 寻址的程序存储单元中	2	4
MVKD dmad, Smem	Smem = dmad	数据存储器内部指定地址传送数据。把 16-Bit 立即数 dmad 寻址的数据存储单元内容复制到单数据存储器操作数寻址的 Smem 单元中	2	2
MVMD MMR, dmad	dmad = MMR	MMR 向数据存储器指定地址传送数据	2	2
MVMM MMRx, MMRy	MMRy = MMRx	MMRx 向 MMRy 传送数据	1	1
MVPD pmad, Smem	Smem = pmad	程序存储器向数据存储器传送数据	2	3
PORTR PA, Smem	Smem = PA	从 PA 口读入数据。从外部 I/O 口（PA—16-bit 立即数地址）把数据读到 Smem 单元中	2	2
PORTW Smem, PA	PA = Smem	向 PA 口输出数据。把 Smem 单元中的 16-bit 数据写到外部 I/O 口 PA 中去	2	2
READA Smem	Smem = A	按累加器 A 寻址读程序存储器并存入数据存储器	1	5
WRITA Smem	A = Smem	把数据存储单元中的值写到由累加器 A 寻址的程序存储器中	1	5

一、填空题

1. 在 C54x DSP 寻址和指令系统中，Xmem 和 Ymem 表示_____，Dmad 为 16 位立即数，表示_____，Pmad 为 16 位立即数，表示_____。

2．C54x DSP 的指令系统有_____和_____形式。

3．在堆栈操作中，PC 当前地址为 4020h，SP 当前地址为 0033h，运行 PSHM AR2 后，PC=_____，SP=_____（假设 PSHM 为单字指令）。

4．立即数寻址指令中在数字或符号常数前面加一个_____号来表示立即数。

5．位倒序寻址方式中，AR0 中存放的是_____。

6．双数据存储器操作数间接寻址所用辅助寄存器只能是_____。

7．在 TMS320C54X 中没有提供专门的除法指令，一般是使用_____指令完成无符号数除法运算。

8．含有 29 个字的循环缓冲器必须从最低_____位为 0 的地址开始。

二、指令执行前有关寄存器及数据存储器单元情况如下图所示，请在下图分别填写指令执行后有关寄存器及数据存储器单元的内容

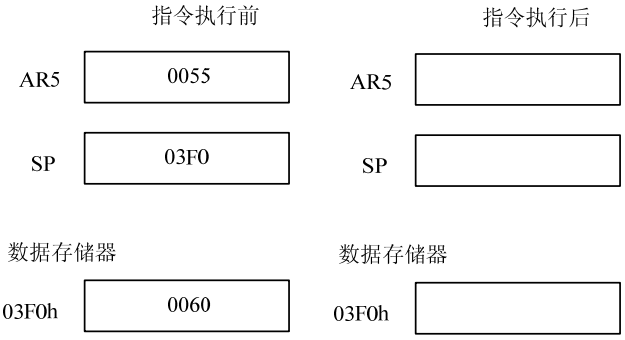
1．ADD　*AR3+ , 14, A

	指令执行前		指令执行后
A	00 0000 1200	A	
C	1	C	0
AR3	0100	AR3	
SXM	1	SXM	1
数据存储器		数据存储器	
0100h	1500	0100h	

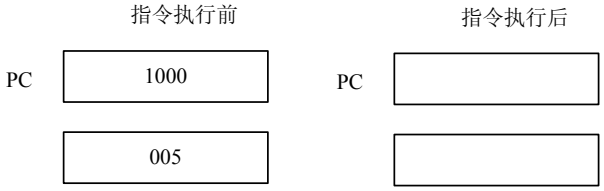
2．PUSH　*AR3+

	指令执行前		指令执行后
AR3	0200	AR3	
SP	8000	SP	
数据存储器		数据存储器	
0200h	07FF	0200h	
7FFFh	07FF	7FFFh	

3．POPM　AR5



4．BANZ 2000h, *AR3-



三、简答题

1．TMS320C54x 提供哪几种数据寻址方式？举例说明它们是如何寻址的？

2.在循环寻址方式中，如何确定循环缓冲的起始地址？如循环缓冲大小为 32，其起始地址必须从哪开始？

3．若辅助寄存器 AR0 的值为 0x0010H，AR3 的值为 0x0310H，循环缓冲起始地址为 0300H，BK=31，请分别给出下列寻址方式修改后的辅助寄存器的值。

*AR3+%

*AR3+0%

*AR3-%

*+AR3(-2)

*AR0(#0100)

4．请描述 TMS320C54x 的位倒序寻址方式。设 FFT 长度 N=16，AR0 应赋值为多少？若 AR2 中存放的数据存储器地址为 FF00H，则经过 8 次*AR2+0B 寻址，访问的内存单元地址依次为多少？

5．双数据存储器操作数间接寻址使用哪几种类型？所用辅助寄存器只能是哪几个？其特点是什么？

6．直接寻址方式有哪两种？其实际地址如何生成？当 SP=2000H，DP=2，偏移地址为 25H 时，分别寻址的是哪个存储空间的哪个单元？

7．TMS320C54x 指令系统包括哪几种基本类型的操作？