ARM指令集—转移指令

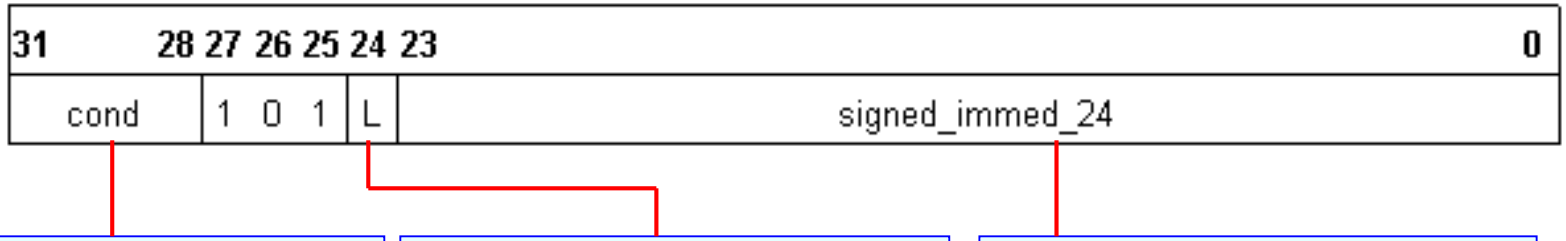
3.2.7 分支指令

- ARM指令集——分支指令

在ARM中有两种方式可以实现程序的跳转，一种是使用分支指令直接跳转，另一种则是直接向PC寄存器赋值实现跳转。 分支指令有以下三种：

- 分支指令B；——Branch
- 带链接的分支指令BL；——Branch with link
- 带状态切换的分支指令BX。——Branch and exchange

- ARM分支指令——指令编码
  - 分支指令B/BL指令编码格式

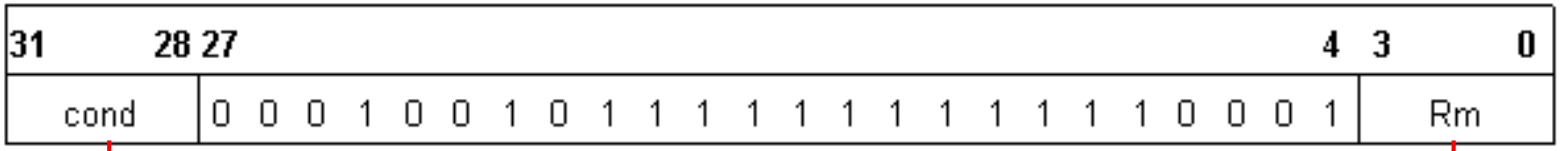


指令执行的条件码

L为0时为B指令
L为1 时为BL指令

24位有符号立即数（偏移量）

  - 分支指令BX指令编码格式



指令执行的条件码

Rm目标地址寄存器，该寄存器装载跳转地址

3.3 ARM指令的使用

- ARM指令——分支指令

助记符	说明	操作	条件码位置
B label	分支指令	PC←label	B{cond}
BL label	带链接的分支指令	LR←PC-4，PC←label	BL{cond}
BX Rm	带状态切换的分支指令	PC←label，切换处理器状态	BX{cond}

3.3 ARM指令的使用

- ARM指令——分支指令

助记符	说明	操作	条件码位置
B label	分支指令	PC←label	B{cond}
BL label	带链接的分支指令	LR←PC-4，PC←label	BL{cond}
BX Rm	带状态切换的分支指令	PC←label，切换处理器状态	BX{cond}

分支指令—B指令，该指令跳转范围限制在当前指令的±32M字节地址内(ARM指令为字对齐，最低2位地址固定为0)。指令格式如下：

```
B{cond}    Label
```

应用示例：

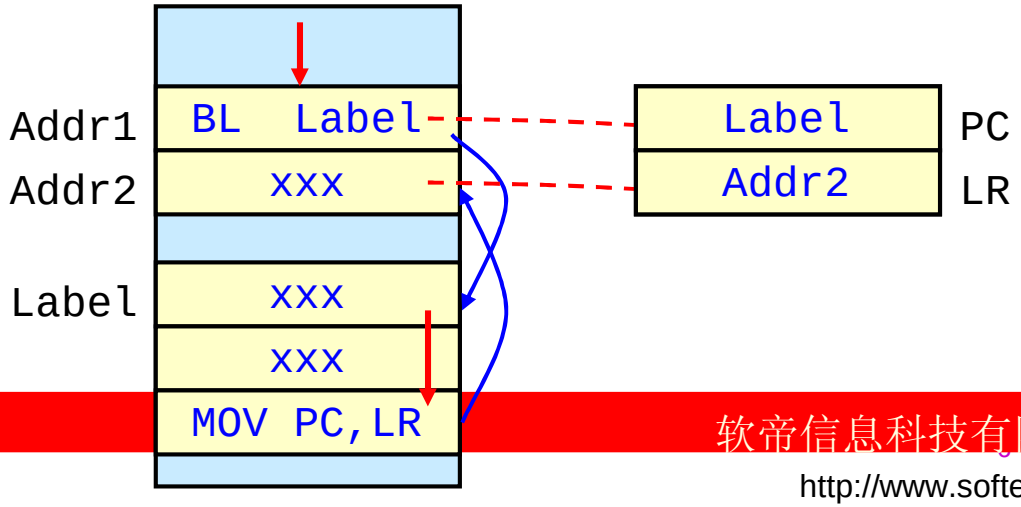
```
B       WAITA          ; 程序无条件跳转到WAITA标号处，
                       ; PC←WAITA
BEQ     0x1234         ; 程序当CPSR中Z=1时，跳转到绝对地
                       ; 址0x1234处
B       0x1234         ;跳转到0x1234处
```

- ARM指令——分支指令

助记符		说明	操作	条件码位置
B	label	分支指令	PC←label	B{cond}
BL	label	带链接的分支指令	LR←PC-4，PC←label	BL{cond}
BX	Rm	带状态切换的分支指令	PC←label，切换处理器状态	BX{cond}

带链接的分支指令——BL指令适用于子程序调用，使用该指令后，下一条指令的地址被拷贝到R14(即LR) 连接寄存器中，然后跳转到指定地址运行程序。跳转范围限制在当前指令的±32M字节地址内。指令格式如下：　　BL{cond}　　Label

2. 程序跳转到目标地址Label继续执行，当子程序执行结束后，将LR寄存器内容存入PC，返回调用函数继续执行



3.3 ARM指令的使用

- ARM指令——分支指令

助记符		说明	操作	条件码位置
B	label	分支指令	PC←Label	B{cond}
BL	label	带链接的分支指令	LR←PC-4，PC←Label	BL{cond}
BX	Rm	带状态切换的分支指令	PC←Label，切换处理器状态	BX{cond}

带链接的分支指令—BL指令适用于子程序调用，使用该指令后，下一条指令的地址被拷贝到R14(即LR) 连接寄存器中，然后跳转到指定地址运行程序。跳转范围限制在当前指令的±32M字节地址内。指令格式如下：

```
BL{cond}    Label
```

应用示例：

```
BL          DELAY                    ; 调用子程序DELAY
```

- ARM指令——分支指令

助记符	说明	操作	条件码位置
B label	分支指令	PC←label	B{cond}
BL label	带链接的分支指令	LR←PC-4，PC←label	BL{cond}
BX Rm	带状态切换的分支指令	PC←label，切换处理器状态	BX{cond}

带状态切换的分支指令—BX指令，该指令可以根据跳转地址（Rm）的最低位来切换处理器状态。其跳转范围限制在当前指令的±32M字节地址内(ARM指令为字对齐，最低2位地址固定为0)。指令格式如下：

BX{cond}　　Rm

跳转地址Rm[0]	跳转后	
	CPSR标志T位	处理器状态
0	0	ARM
1	1	Thumb

- ARM指令——分支指令

助记符	说明	操作	条件码位置
B label	分支指令	PC←label	B{cond}
BL label	带链接的分支指令	LR←PC-4，PC←label	BL{cond}
BX Rm	带状态切换的分支指令	PC←label，切换处理器状态	BX{cond}

带状态切换的分支指令—BX指令，该指令可以根据跳转地址（Rm）的最低位来切换处理器状态。其跳转范围限制在当前指令的±32M字节地址内(ARM指令为字对齐，最低2位地址固定为0)。指令格式如下：

```
BX{cond}    Rm
```

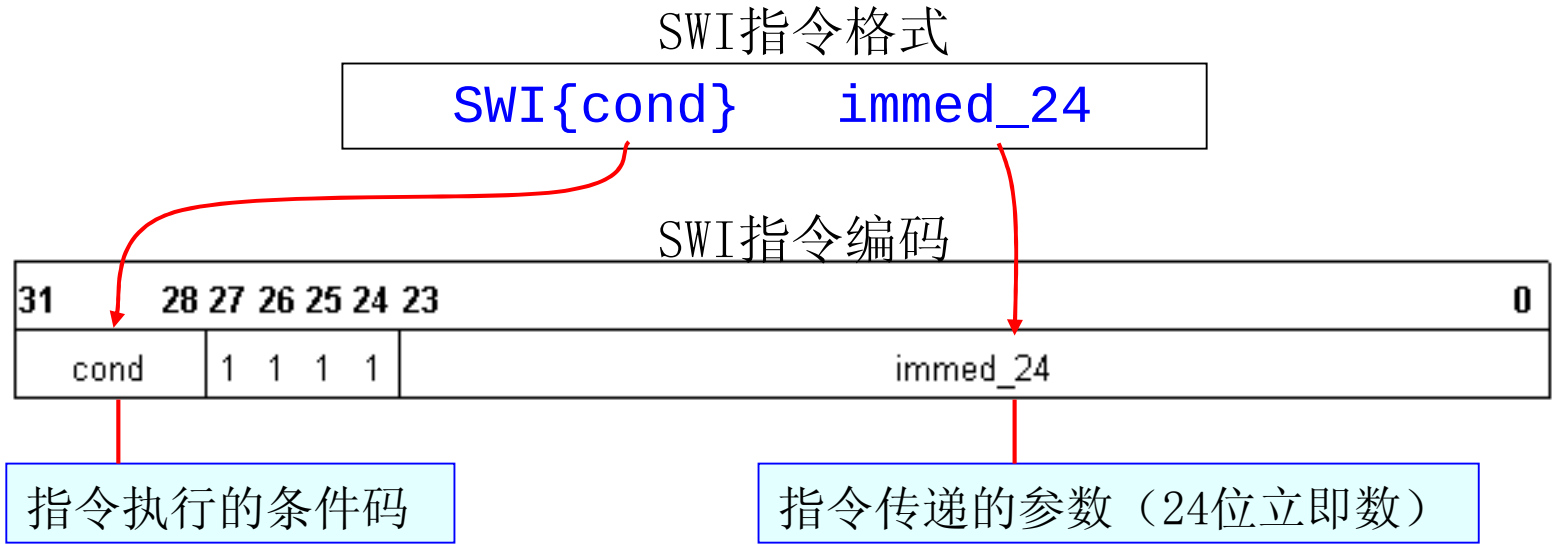
应用示例：

```
ADRL    R0,ThumbFun+1  ;将Thumb程序的入口地址加1存入R0
BX      R0             ;跳转到R0指定的地址，
                       ;并根据R0的最低位来切换处理器状态
```

- ARM杂项指令——软中断指令

SWI指令用于产生软中断，从而实现在从用户模式变换到管理模式，并且将CPSR保存到管理模式的SPSR(spsr_svc)中，然后程序跳转到SWI异常入口。在其它模式下也可使用SWI指令，处理器同样地切换到管理模式。

该指令主要用于用户程序调用操作系统的系统服务，操作系统在SWI异常处理程序中进行相应的系统服务。



- ARM杂项指令——软中断指令

根据SWI指令传递的参数SWI异常处理程序可以作出相应的处理。SWI指令传递参数有以下两种方法.

▪指令中的24位立即数指定了用户请求的服务类型，参数通过通用寄存器传递。

```
MOV     R0,#34          ;设置子功能号为34
SWI     12              ;调用12号软中断
```

▪指令中的24位立即数被忽略，用户请求的服务类型由寄存器R0的值决定，参数通过其它的通用寄存器传递。

```
MOV     R0,#12          ;调用12号软中断
MOV     R1,#34          ;设置子功能号为34
SWI     0
```

- ARM杂项指令——软中断指令

在SWI异常中断处理程序中，取出SWI指令中立即数的步骤为：

- 首先确定引起软中断的SWI指令是ARM指令还是Thumb指令，这可通过对SPSR访问得到；
- 然后取得该SWI指令的地址，这可通过访问LR寄存器得到；
- 接着读出该SWI指令，分解出立即数。

```
SWI_Handler
    STMFD   SP!, {R0-R3, R12, LR}   ; 现场保护
    MRS     R0, SPSR                ; 读取SPSR
    STMFD   SP!, {R0}               ; 保存SPSR
    TST     R0, #0x20               ; 测试T标志位
    LDRNEH  R0, [LR,#-2]            ; 若是Thumb指令，读取指令码(16位)
    BICNE   R0, R0, #0xFF00         ; 取得Thumb指令的8位立即数
    LDREQ   R0, [LR,#-4]            ; 若是ARM指令，读取指令码(32位)
    BICEQ   R0, R0, #0xFF000000     ; 取得ARM指令的24位立即数
            ...
    LDMFD   SP!, {R0-R3, R12, PC}^ ; SWI异常中断返回
```

Q & A