1. **A8 处理器工作模式**

这点和以往的 arm7 arm11 是一致的。有8个工作模式：

* 用户模式是通常的ARM程序的执行状态，并用于执行大多数应用程序
* 快速中断（FIQ）模式是用于处理快速中断
* 中断（IRQ）模式是用于通用的中断处理
* 管理模式是一个操作系统保护模式
* 中断模式是当数据出错或者预取指令出错时进入的模式
* 系统模式是操作系统的一个特权用户模式
* 未定义的模式进入一个未定义指令异常发生时
* 监控方式为安全扩展安全模式安全监控代码。

除了用户模式模式统称为特权模式。特权模式用于服务中断或异常，或访问受保护的资源。下面的表格显示了处理器的结构模式。

| **Modes** | **Mode type** | **Security state of core** | |
| --- | --- | --- | --- |
|  |  | **NS bit = 1** | **NS bit = 0** |
| User | User | Nonsecure | Secure |
| FIQ | Privileged | Nonsecure | Secure |
| IRQ | Privileged | Nonsecure | Secure |
| Supervisor | Privileged | Nonsecure | Secure |
| Abort | Privileged | Nonsecure | Secure |
| Undefined | Privileged | Nonsecure | Secure |
| System | Privileged | Nonsecure | Secure |
| Monitor | Privileged | Secure | Secure |

1. **寻址方式**
2. **立即寻址**

例：

ADD R0, R0， #0x1 ；R0 <- R0 + 1

ADD R0, R0, #0x3f ；R0 <- R0 + 0x3f

以上两条指令，第二个源操作数为立即数。

1. **寄存器寻址**

例：

ADD R0， R1， R2 ；R0 <- R1 + R2

将寄存器 R1 和 R2 存放的内容相加，结果存放在寄存器 R0 中。

1. **寄存器间接寻址**

例：

LDR R0， 。 [R1] ；R0 <- [R1]

STR R0， [R1] ；[R1] <- R0

第一条指令，将 R1 的值作为地址，取出存储器中的数据传送到 R0 中；第二条指令，将 R0 的数据存储到 R1 的值为地址的存储器里面。

1. **多寄存器寻址**

例：

LDMIA R0， {R1，R2，R3，R4} ；R1 <- [R0],R2 <- [R0 + 4],

；R3 <- [R0 +8]，R3 <- [R0 + 12]

STMIA R0， {R3-R5，R10} ；[R0] <- R3，[R0 + 4] <- R4

；[R0 + 8] <- R5，[R5 + 12] <- R10

这种寻址方式可以用一条指令完成传送最多 16 个通用寄存器的值。指令中的后缀 IA 表示在每次执行完加载/存储操作后，R0 按字长增加。上面二条指令中，前者表示读出以 R0 的值为地址里面的内容赋给 R1、读出R0+4 的值为地址里面的内容赋给 R2、依次类推；后者表示将 R3~R5、R10 的数据保存到 R0 指向的地址，R0 的地址按字长增加。

1. **常用ARM指令集**

**1. 跳转指令**

跳转指令用于实现程序流程的跳转，在 ARM 程序中有两种方法可以实现程序

流程的跳转：

① 使用专门的跳转指令。

② 直接向程序计数器 PC 写入跳转地址值。

ARM 汇编中总共有四种跳转指令，如下：

① B 跳转指令（注:只能跳转前后 32MB 的地址空间）B 指令编码中的跳转地址值是相对当前 PC 的一个偏移量，经汇编器计算得到跳转的绝对地址。

② BL 带返回的跳转指令，BL 指令在执行跳转的同时将转移指令的下一条指令的地址复制到当前处理器模式下的链接地址 LR 中，这一般用于子程序的调用和返回。

③ BLX 带返回和状态切换的跳转指令

④ BX 带状态切换的跳转指令

**2. 数据处理指令**

数据处理指令分类：

① 数据传送指令

例：

MOV R1，R0 ；将寄存器 R0 的值传送到寄存器 R1

MOV R1，R0，LSL #3 ；将寄存器 R0 的值左移 3 位后传送到 R1

② 比较指令

例：

CMP R1，#100 ；将寄存器 R1 的值与立即数 100 相减，并根据结果设置 CPSR 的标志位。

TST R1，#0xfffe ；将寄存器 R1 的值与立即数 0xfffe 按位与，并根据结果设置 CPSR 的标志位。

③ 算数运算指令

例：

ADD R0， R1， #256 ；R0 = R1 + 256

SUB R0， R1， R2 ；R0 = R1 –R2

RSB R0， R2， R3， LSL #1 ；R0 = (R3 << 1) – R2

④ 逻辑运算指令

例：

ADD R0，R0， #3 ；该指令保持 R0 的 0、1 位，其余位清零。

ORR R0，R0， #3 ；该指令置 R0 的 0、1 位为 1，其余位保持不变。

BIC R0， R0， #0xf8 ；该指令清除 R0 的高 5 位，其余位保持不变。

**3. 乘法指令**

例:

MUL R0, R1, R2 ；R0 = R1 \* R2

**4. 加载/存储 指令**

ARM 处理器支持 加载/存储指令用于在寄存器和存储器之间传送数据

例：

LDR R0， [R1，R2] ；将存储器地址为 R1+R2 的字数据读入寄存器R0

LDR R0， [R1，R2]! ；将存储器地址为 R1+R2 的字数据读入寄存 R0并将新地址R1+R2 写入 R1

STR R0，[R1] , #8 ；将 R0 中的字数据写入以 R1 为地址的存储器中，并将新地址 R1+8 写入 R1

STR R0，[R1,#8] ；将 R0 中的字数据写入以 R1+8 为地址的存储器中

**5. 程序状态寄存器访问指令**

① MSR 通用寄存器到程序状态寄存器的数据传送指令

② MRS 程序状态寄存器到通用寄存器的数据传送指令

例：

MSR CPSR, R0 ；传送 R0 的内容到 CPSR 寄存器

MRS R0, CPSR ；传送 CPSR 的内容到 R0 寄存器