

Thumb® 16 位指令集

快速参考卡

本卡列出了版本低于 ARM®v6T2 的支持 Thumb 的处理器中可用的所有 Thumb 指令。此外，还列出了所有 Thumb-2 16 位指令。  
除非另外注明，否则本卡中显示的指令均为 Thumb-2 16 位指令。  
除非指定，否则所有寄存器都为 Lo (R0-R7)。Hi 寄存器为 R8-R15。

表关键字				
\$	请参阅表 <b>ARM 体系结构版本</b> 。		<lreglist+LR>	以逗号隔开的 Lo 寄存器列表。加上 LR，括在大括号 { 和 } 内。
<lreglist>	以逗号隔开的 Lo 寄存器列表，括在大括号 { 和 } 内。		<lreglist+PC>	以逗号隔开的 Lo 寄存器列表。加上 PC，括在大括号 { 和 } 内。

运算		\$	汇编程序	更新	操作	注释
移动	立即数	6	MOVS Rd, #<imm>	N Z	Rd := imm	imm 范围为 0-255。
	Lo 到 Lo		MOVS Rd, Rm	N Z	Rd := Rm	LSLS Rd, Rm, #0 的同义词
	Hi 到 Lo、Lo 到 Hi、Hi 到 Hi		MOV Rd, Rm		Rd := Rm	不是 Lo 到 Lo。
	任何寄存器之间		MOV Rd, Rm		Rd := Rm	任何寄存器之间。
加法	立即数 3	T2	ADDS Rd, Rn, #<imm>	N Z C V	Rd := Rn + imm	imm 范围为 0-7。
	所有寄存器 Lo		ADDS Rd, Rn, Rm	N Z C V	Rd := Rn + Rm	
	Hi 到 Lo、Lo 到 Hi、Hi 到 Hi		ADD Rd, Rd, Rm		Rd := Rd + Rm	不是 Lo 到 Lo。
	任何寄存器之间		ADD Rd, Rd, Rm		Rd := Rd + Rm	任何寄存器之间。
	立即数 8		ADDS Rd, Rd, #<imm>	N Z C V	Rd := Rd + imm	imm 范围为 0-255。
	带进位		ADCS Rd, Rd, Rm	N Z C V	Rd := Rd + Rm + C-bit	
	值与 SP		ADD SP, SP, #<imm>		SP := SP + imm	imm 范围为 0-508 (字对齐)。
	SP 所存储的地址		ADD Rd, SP, #<imm>		Rd := SP + imm	imm 范围为 0-1020 (字对齐)。
减法	Lo 到 Lo		ADRS Rd, <label>		Rd := label	标签范围为 PC 到 PC+1020 (字对齐)。
	立即数 3		SUBS Rd, Rn, Rm	N Z C V	Rd := Rn - Rm	
	立即数 8		SUBS Rd, Rn, #<imm>	N Z C V	Rd := Rn - imm	imm 范围为 0-7。
	带进位		SUBS Rd, Rd, #<imm>	N Z C V	Rd := Rd - imm	imm 范围为 0-255。
	来自 SP 的值		SBCS Rd, Rd, Rm	N Z C V	Rd := Rd - Rm - NOT C 位	
	求反		SUB SP, SP, #<imm>		SP := SP - imm	imm 范围为 0-508 (字对齐)。
乘法	乘法		RSBS Rd, Rn, #0	N Z C V	Rd := - Rn	同义词 NEG Rd, Rn
			MULS Rd, Rd, Rm	N Z * *	Rd := Rm * Rd	* C 和 V 标记在 4T 中不可预测， 在 ST 及更高版本中保持不变
比较			CMP Rn, Rm	N Z C V	更新 Rn - Rm 的 CPSR 标记	可为 Lo 和 Lo、Lo 和 Hi、Hi 和 Lo 或者 Hi 和 Hi。
	负数		CMN Rn, Rm	N Z C V	更新 Rn + Rm 的 CPSR 标记	
	立即数		CMP Rn, #<imm>	N Z C V	更新 Rn - imm 的 CPSR 标记	imm 范围为 0-255。
逻辑	和		ANDS Rd, Rd, Rm	N Z	Rd := Rd AND Rm	
	异或		EORS Rd, Rd, Rm	N Z	Rd := Rd EOR Rm	
	或		ORRS Rd, Rd, Rm	N Z	Rd := Rd OR Rm	
	位清除		BICS Rd, Rd, Rm	N Z	Rd := Rd AND NOT Rm	
	取反移动		MVNS Rd, Rd, Rm	N Z	Rd := NOT Rm	
	测试位		TST Rn, Rm	N Z	更新 Rn AND Rm 的 CPSR 标记	
移位/循环	逻辑左移		LSLS Rd, Rm, #<shift>	N Z C*	Rd := Rm << shift	允许移动 0-31 位。如果移位为 0，则不会影响 * C 标记。
			LSLS Rd, Rd, Rs	N Z C*	Rd := Rd << Rs[7:0]	如果 Rs[7:0] 为 0，则不会影响 * C 标记。
	逻辑右移		LSRS Rd, Rm, #<shift>	N Z C	Rd := Rm >> shift	允许移动 1-32 位。
			LSRS Rd, Rd, Rs	N Z C*	Rd := Rd >> Rs[7:0]	如果 Rs[7:0] 为 0，则不会影响 * C 标记。
	算术右移		ASRS Rd, Rm, #<shift>	N Z C	Rd := Rm ASR shift	允许移动 1-32 位。
			ASRS Rd, Rd, Rs	N Z C*	Rd := Rd ASR Rs[7:0]	如果 Rs[7:0] 为 0，则不会影响 * C 标记。
	向右循环移		RORS Rd, Rd, Rs	N Z C*	Rd := Rd ROR Rs[7:0]	如果 Rs[7:0] 为 0，则不会影响 * C 标记。

Thumb 16 位指令集  
快速参考卡

运算		\$	汇编程序	操作	注释
加载	带直接偏移量, 字		LDR Rd, [Rn, #<imm>]	Rd := [Rn + imm]	imm 为 0-124 范围内的 4 的倍数。
	半字		LDRH Rd, [Rn, #<imm>]	Rd := ZeroExtend([Rn + imm][15:0])	清除位 31:16。imm 范围为 0-62, 偶数。
	字节		LDRB Rd, [Rn, #<imm>]	Rd := ZeroExtend([Rn + imm][7:0])	清除位 31:8。imm 范围为 0-31。
	带寄存器偏移量, 字		LDR Rd, [Rn, Rm]	Rd := [Rn + Rm]	
	半字		LDRH Rd, [Rn, Rm]	Rd := ZeroExtend([Rn + Rm][15:0])	清除位 31:16
	有符号半字		LDRSH Rd, [Rn, Rm]	Rd := SignExtend([Rn + Rm][15:0])	将位 31:16 设置为位 15
	字节		LDRB Rd, [Rn, Rm]	Rd := ZeroExtend([Rn + Rm][7:0])	清除位 31:8
	有符号字节		LDRSB Rd, [Rn, Rm]	Rd := SignExtend([Rn + Rm][7:0])	将位 31:8 设置为位 7
存储	相对 PC		LDR Rd, <label>	Rd := [label]	标签范围为 PC 到 PC+1020 (字对齐)。
	相对 SP		LDR Rd, [SP, #<imm>]	Rd := [SP + imm]	imm 为 0-1020 范围内的 4 的倍数。
	多个寄存器, 不包括基址寄存器		LDM Rn!, <loreglist>	加载寄存器列表 (不包括 Rn)	始终更新基址寄存器, 之后增加。
	多个寄存器, 包括基址寄存器		LDM Rn, <loreglist>	加载寄存器列表 (包括 Rn)	永不更新基址寄存器, 之后增加。
	带直接偏移量, 字		STR Rd, [Rn, #<imm>]	[Rn + imm] := Rd	imm 为 0-124 范围内的 4 的倍数。
	半字		STRH Rd, [Rn, #<imm>]	[Rn + imm][15:0] := Rd[15:0]	忽略 Rd[31:16]。imm 范围为 0-62, 偶数。
	字节		STRB Rd, [Rn, #<imm>]	[Rn + imm][7:0] := Rd[7:0]	忽略 Rd[31:8]。imm 范围为 0-31。
	带寄存器偏移量, 字		STR Rd, [Rn, Rm]	[Rn + Rm] := Rd	
推入	半字		STRH Rd, [Rn, Rm]	[Rn + Rm][15:0] := Rd[15:0]	忽略 Rd[31:16]
	字节		STRB Rd, [Rn, Rm]	[Rn + Rm][7:0] := Rd[7:0]	忽略 Rd[31:8]
	相对 SP, 字		STR Rd, [SP, #<imm>]	[SP + imm] := Rd	imm 为 0-1020 范围内的 4 的倍数。
	多个寄存器		STM Rn!, <loreglist>	存储寄存器列表	始终更新基址寄存器, 之后增加。
	带链接的推入		PUSH <loreglist>	将寄存器推入满降序堆栈	
			PUSH <loreglist+LR>	将 LR 和寄存器推入满降序堆栈	
	弹出		POP <loreglist>	从满降序堆栈中弹出寄存器	
	弹出和返回	4T	POP <loreglist+PC>	弹出寄存器, 跳转到加载入 PC 的地址	
条件判断	带交换的弹出和返回	5T	POP <loreglist+PC>	如果 address[0] = 0, 则弹出、跳转并更改为 ARM 状态	
	条件判断	T2	IT{pattern} {cond}	依据不同的模式, 最多由连续四条条件指令句组成。模式是一个字符串, 最多三个字母。所有字母都可为 T (然后) 或 E (否则)。	IT 之后的第一条指令可有条件 cond。如果相应的字母为 T, 则后续指令可有条件 cond, 如果相应的字母为 E, 则为 cond 的反面情况。 请参阅表 <b>条件字段</b> 。
	条件跳转		B{cond} <label>	如果 {cond}, 则 PC := label	label 必须位于当前指令的 - 252 到 + 258 字节范围内。 请参阅表 <b>条件字段</b> 。
	比较, 如果为 (非) 零, 则跳转	T2	CB{N}Z Rn,<label>	如果 Rn {==   !=} 0, 则 PC := label	label 必须位于当前指令的 +4 到 +130 字节范围内。
	无条件跳转		B <label>	PC := label	label 必须位于当前指令的 ±2KB 范围之内。
	带链接的长跳转		BL <label>	LR := 下一指令的地址, PC := label	这是一个 32 位指令。 label 必须位于当前指令的 ±4MB 范围之内 (T2: ±16MB)。
	跳转并交换		BX Rm	PC := Rm AND 0xFFFFFFFFE	如果 Rm[0] = 0, 则更改为 ARM 状态。
	带链接和交换的跳转	5T	BLX <label>	LR := 下一指令的地址, PC := label 更改为 ARM	这是一个 32 位指令。 label 必须位于当前指令的 ±4MB 范围之内 (T2: ±16MB)。
反转	带链接和交换的跳转	5T	BLX Rm	LR := 下一指令的地址, PC := Rm AND 0xFFFFFFFFE	如果 Rm[0] = 0, 则更改为 ARM 状态
	有符号, 半字到字	6	SXTH Rd, Rm	Rd[31:0] := SignExtend(Rm[15:0])	
	有符号, 字节到字	6	SXTB Rd, Rm	Rd[31:0] := SignExtend(Rm[7:0])	
	无符号, 半字到字	6	UXTH Rd, Rm	Rd[31:0] := ZeroExtend(Rm[15:0])	
	无符号, 字节到字	6	UXTB Rd, Rm	Rd[31:0] := ZeroExtend(Rm[7:0])	
	字中的字节	6	REV Rd, Rm	Rd[31:24] := Rm[7:0], Rd[23:16] := Rm[15:8], Rd[15:8] := Rm[23:16], Rd[7:0] := Rm[31:24]	
	两个半字中的字节	6	REV16 Rd, Rm	Rd[15:8] := Rm[7:0], Rd[7:0] := Rm[15:8], Rd[31:24] := Rm[23:16], Rd[23:16] := Rm[31:24]	
	低半字中的字节, 符号扩展	6	REVSH Rd, Rm	Rd[15:8] := Rm[7:0], Rd[7:0] := Rm[15:8], Rd[31:16] := Rm[7] * &FFFF	

# Thumb 16 位指令集

## 快速参考卡

运算		S	汇编程序	操作	注释
处理器 状态 更改	超级用户调用		SVC <immed_8>	超级用户调用处理器异常	在指令中编码的 8 位立即值。以前为 SWI。  <endianness> 可为 BE（大端）或 LE（小端）。 在指令中编码的 8 位立即值。
	更改处理器状态	6	CPSID <iflags>	禁用指定的中断	
		6	CPSIE <iflags>	启用指定的中断	
	设置端序	6	SETEND <endianness>	为加载和保存设置端序	
	断点	5T	BKPT <immed_8>	预取中止或进入调试状态	
无操作	无操作	6	NOP	无操作，可能不花费任何时间。	
提示	设置事件	T2	SEV	向多处理器系统发送事件信号。	如果不执行，则无操作。
	等待事件	T2	WFE	等待事件、IRQ、FIQ、不精确的中止或调试进入请求。	如果不执行，则无操作。
	等待中断	T2	WFI	等待 IRQ、FIQ、不精确的中止或调试进入请求。	如果不执行，则无操作。
	Yield	T2	YIELD	生成对其他线程的控制。	如果不执行，则无操作。

条件字段	
助记符	说明
EQ	等于
NE	不等于
CS / HS	进位设置/无符号大于或相同
CC / LO	进位清除/无符号小于
MI	负数
PL	正数或零
VS	溢出
VC	无溢出
HI	无符号大于
LS	无符号小于或相同
GE	有符号大于或等于
LT	有符号小于
GT	有符号大于
LE	有符号小于或等于
AL	始终。不可用于 B{cond} 中

对于版本早于 ARMv6T2 的处理器，在 Thumb 代码中，cond 只可用于条件跳转（B{cond}）指令中。

在 Thumb-2 代码中，cond 可用于所有这些指令中（除了 CBZ、CBNZ、CPSID、CPSIE、IT 和 SETEND）。  
条件是在前面的 IT 指令中进行编码的（B{cond} 指令除外）。  
如果汇编语言源文件中显式提供了 IT 指令，  
则指令中的条件必须与相应的 IT 指令相匹配。

ARM 体系结构版本	
4T	ARM v4 及更高版本的所有 Thumb 版本。
5T	ARM v5 及更高版本的所有 Thumb 版本。
6	ARM v6 及更高版本的所有 Thumb 版本。
T2	ARM v6 及更高版本的所有 Thumb-2 版本。

### 所有权声明

带有® 或™ 标记的词语和徽标是 ARM 公司的注册商标或商标。此处提及的其他品牌和名称可能是其各自所有者的商标。

除非事先得到版权所有人的书面许可，否则不得以任何形式修改或复制本文档包含的部分或全部信息以及产品说明。

本文档描述的产品还将不断发展和完善。ARM 公司将如实提供本文档所述产品的所有特性及其使用方法。但是，所有暗示或明示的担保，包括但不限于对特定用途适销性或适用性的担保，均不包括在内。

本参考卡旨在帮助读者使用产品。对由于使用本参考卡中的任何信息，或由于本参考卡的信息错误、遗漏，以及产品的错误使用所造成的任何损失，ARM 公司概不负责。

### 文档编号

ARM QRC 0006D

### 变更记录

编号	日期	变更
A	2004 年 11 月	第一版
B	2005 年 5 月	RVCT 2.2 SP1
C	2006 年 3 月	RVCT 3.0
D	2007 年 3 月	RVCT 3.1