

Thumb® 16-bit Instruction Set

Quick Reference Card

This card lists all Thumb instructions available on Thumb-capable processors earlier than ARM®v6T2. In addition, it lists all Thumb-2 16-bit instructions. The instructions shown on this card are all 16-bit in Thumb-2, except where noted otherwise. All registers are Lo (R0-R7) except where specified. Hi registers are R8-R15.

Key to Tables

§	See Table ARM architecture versions .	<loreglist+LR>	A comma-separated list of Lo registers, plus the LR, enclosed in braces, { and }.
<loreglist>	A comma-separated list of Lo registers, enclosed in braces, { and }.	<loreglist+PC>	A comma-separated list of Lo registers, plus the PC, enclosed in braces, { and }.

Operation	§	Assembler	Updates	Action	Notes
Move	Immediate	MOVS Rd, #<imm>	N Z	Rd := imm	imm range 0-255.
	Lo to Lo	MOVS Rd, Rm	N Z	Rd := Rm	Synonym of LSLs Rd, Rm, #0
	Hi to Lo, Lo to Hi, Hi to Hi	MOV Rd, Rm		Rd := Rm	Not Lo to Lo.
	Any to Any	MOV Rd, Rm		Rd := Rm	Any register to any register.
Add	Immediate 3	ADDS Rd, Rn, #<imm>	N Z C V	Rd := Rn + imm	imm range 0-7.
	All registers Lo	ADDS Rd, Rn, Rm	N Z C V	Rd := Rn + Rm	
	Hi to Lo, Lo to Hi, Hi to Hi	ADD Rd, Rd, Rm		Rd := Rd + Rm	Not Lo to Lo.
	Any to Any	ADD Rd, Rd, Rm		Rd := Rd + Rm	Any register to any register.
	Immediate 8	ADDS Rd, Rd, #<imm>	N Z C V	Rd := Rd + imm	imm range 0-255.
	With carry	ADCS Rd, Rd, Rm	N Z C V	Rd := Rd + Rm + C-bit	
	Value to SP	ADD SP, SP, #<imm>		SP := SP + imm	imm range 0-508 (word-aligned).
	Form address from SP	ADD Rd, SP, #<imm>		Rd := SP + imm	imm range 0-1020 (word-aligned).
Form address from PC	ADR Rd, <label>		Rd := label	label range PC to PC+1020 (word-aligned).	
Subtract	Lo and Lo	SUBS Rd, Rn, Rm	N Z C V	Rd := Rn - Rm	
	Immediate 3	SUBS Rd, Rn, #<imm>	N Z C V	Rd := Rn - imm	imm range 0-7.
	Immediate 8	SUBS Rd, Rd, #<imm>	N Z C V	Rd := Rd - imm	imm range 0-255.
	With carry	SBCS Rd, Rd, Rm	N Z C V	Rd := Rd - Rm - NOT C-bit	
	Value from SP	SUB SP, SP, #<imm>		SP := SP - imm	imm range 0-508 (word-aligned).
Negate	RSBS Rd, Rn, #0	N Z C V	Rd := -Rn	Synonym: NEG Rd, Rn	
Multiply	Multiply	MULS Rd, Rd, Rm	N Z * *	Rd := Rm * Rd	* C and V flags unpredictable in §4T, unchanged in §5T and above
Compare		CMP Rn, Rm	N Z C V	update CPSR flags on Rn - Rm	Can be Lo to Lo, Lo to Hi, Hi to Lo, or Hi to Hi.
	Negative	CMN Rn, Rm	N Z C V	update CPSR flags on Rn + Rm	
	Immediate	CMP Rn, #<imm>	N Z C V	update CPSR flags on Rn - imm	imm range 0-255.
Logical	AND	ANDS Rd, Rd, Rm	N Z	Rd := Rd AND Rm	
	Exclusive OR	EORS Rd, Rd, Rm	N Z	Rd := Rd EOR Rm	
	OR	ORRS Rd, Rd, Rm	N Z	Rd := Rd OR Rm	
	Bit clear	BICS Rd, Rd, Rm	N Z	Rd := Rd AND NOT Rm	
	Move NOT	MVNS Rd, Rd, Rm	N Z	Rd := NOT Rm	
	Test bits	TST Rn, Rm	N Z	update CPSR flags on Rn AND Rm	
Shift/rotate	Logical shift left	LSLS Rd, Rm, #<shift>	N Z C*	Rd := Rm << shift	Allowed shifts 0-31. * C flag unaffected if shift is 0.
		LSLS Rd, Rd, Rs	N Z C*	Rd := Rd << Rs[7:0]	* C flag unaffected if Rs[7:0] is 0.
	Logical shift right	LSRS Rd, Rm, #<shift>	N Z C	Rd := Rm >> shift	Allowed shifts 1-32.
		LSRS Rd, Rd, Rs	N Z C*	Rd := Rd >> Rs[7:0]	* C flag unaffected if Rs[7:0] is 0.
	Arithmetic shift right	ASRS Rd, Rm, #<shift>	N Z C	Rd := Rm ASR shift	Allowed shifts 1-32.
		ASRS Rd, Rd, Rs	N Z C*	Rd := Rd ASR Rs[7:0]	* C flag unaffected if Rs[7:0] is 0.
	Rotate right	RORS Rd, Rd, Rs	N Z C*	Rd := Rd ROR Rs[7:0]	* C flag unaffected if Rs[7:0] is 0.

Thumb 16-bit Instruction Set

Quick Reference Card

Operation	§	Assembler	Action	Notes
Load	with immediate offset, word	LDR Rd, [Rn, #<imm>]	Rd := [Rn + imm]	imm range 0-124, multiple of 4. Clears bits 31:16. imm range 0-62, even. Clears bits 31:8. imm range 0-31.
		LDRH Rd, [Rn, #<imm>]	Rd := ZeroExtend([Rn + imm][15:0])	
	with register offset, word	LDRB Rd, [Rn, #<imm>]	Rd := ZeroExtend([Rn + imm][7:0])	Clears bits 31:16 Sets bits 31:16 to bit 15 Clears bits 31:8 Sets bits 31:8 to bit 7 label range PC to PC+1020 (word-aligned). imm range 0-1020, multiple of 4.
		LDR Rd, [Rn, Rm]	Rd := [Rn + Rm]	
	halfword	LDRH Rd, [Rn, Rm]	Rd := ZeroExtend([Rn + Rm][15:0])	Always updates base register, Increment After. Never updates base register, Increment After.
	signed halfword	LDRSH Rd, [Rn, Rm]	Rd := SignExtend([Rn + Rm][15:0])	
	byte	LDRB Rd, [Rn, Rm]	Rd := ZeroExtend([Rn + Rm][7:0])	label range PC to PC+1020 (word-aligned). imm range 0-1020, multiple of 4.
	signed byte	LDRSB Rd, [Rn, Rm]	Rd := SignExtend([Rn + Rm][7:0])	
	PC-relative	LDR Rd, <label>	Rd := [label]	Always updates base register, Increment After. Never updates base register, Increment After.
	SP-relative	LDR Rd, [SP, #<imm>]	Rd := [SP + imm]	
Multiple, not including base	LDM Rn!, <loreglist>	Loads list of registers (not including Rn)	Always updates base register, Increment After. Never updates base register, Increment After.	
Multiple, including base	LDM Rn, <loreglist>	Loads list of registers (including Rn)		
Store	with immediate offset, word	STR Rd, [Rn, #<imm>]	[Rn + imm] := Rd	imm range 0-124, multiple of 4. Ignores Rd[31:16]. imm range 0-62, even. Ignores Rd[31:8]. imm range 0-31.
		STRH Rd, [Rn, #<imm>]	[Rn + imm][15:0] := Rd[15:0]	
	with register offset, word	STRB Rd, [Rn, #<imm>]	[Rn + imm][7:0] := Rd[7:0]	Ignores Rd[31:16] Ignores Rd[31:8] imm range 0-1020, multiple of 4.
		STR Rd, [Rn, Rm]	[Rn + Rm] := Rd	
	halfword	STRH Rd, [Rn, Rm]	[Rn + Rm][15:0] := Rd[15:0]	Always updates base register, Increment After.
	byte	STRB Rd, [Rn, Rm]	[Rn + Rm][7:0] := Rd[7:0]	
	SP-relative, word	STR Rd, [SP, #<imm>]	[SP + imm] := Rd	Always updates base register, Increment After.
Multiple	STM Rn!, <loreglist>	Stores list of registers		
Push	Push	PUSH <loreglist>	Push registers onto full descending stack	
	Push with link	PUSH <loreglist>+LR	Push LR and registers onto full descending stack	
Pop	Pop	POP <loreglist>	Pop registers from full descending stack	
	Pop and return	4T POP <loreglist>+PC	Pop registers, branch to address loaded to PC	
	Pop and return with exchange	5T POP <loreglist>+PC	Pop, branch, and change to ARM state if address[0] = 0	
If-Then	If-Then	T2 IT{pattern} {cond}	Makes up to four following instructions conditional, according to pattern. pattern is a string of up to three letters. Each letter can be T (Then) or E (Else).	The first instruction after IT has condition cond. The following instructions have condition cond if the corresponding letter is T, or the inverse of cond if the corresponding letter is E. See Table Condition Field .
Branch	Conditional branch	B{cond} <label>	If {cond} then PC := label	label must be within -252 to +258 bytes of current instruction. See Table Condition Field .
	Compare, branch if (non) zero	T2 CB{N}Z Rn, <label>	If Rn {== !=} 0 then PC := label	label must be within +4 to +130 bytes of current instruction.
	Unconditional branch	B <label>	PC := label	label must be within ±2KB of current instruction.
	Long branch with link	BL <label>	LR := address of next instruction, PC := label	This is a 32-bit instruction. label must be within ±4MB of current instruction (T2: ±16MB).
	Branch and exchange	BX Rm	PC := Rm AND 0xFFFFFFF	Change to ARM state if Rm[0] = 0.
	Branch with link and exchange	5T BLX <label>	LR := address of next instruction, PC := label Change to ARM	This is a 32-bit instruction. label must be within ±4MB of current instruction (T2: ±16MB).
Extend	Branch with link and exchange	5T BLX Rm	LR := address of next instruction, PC := Rm AND 0xFFFFFFF	Change to ARM state if Rm[0] = 0
	Signed, halfword to word	6 SXTB Rd, Rm	Rd[31:0] := SignExtend(Rm[15:0])	
	Signed, byte to word	6 SXTB Rd, Rm	Rd[31:0] := SignExtend(Rm[7:0])	
	Unsigned, halfword to word	6 UXTB Rd, Rm	Rd[31:0] := ZeroExtend(Rm[15:0])	
Unsigned, byte to word	6 UXTB Rd, Rm	Rd[31:0] := ZeroExtend(Rm[7:0])		
Reverse	Bytes in word	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]	
	Bytes in both halfwords	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]	
	Bytes in low halfword, sign extend	6 REVSH Rd, Rm	Rd[15:8] := Rm[7:0], Rd[7:0] := Rm[15:8], Rd[31:16] := Rm[7] * &FFFF	

Thumb 16-bit Instruction Set

Quick Reference Card

Operation	§	Assembler	Action	Notes
Processor state change	Supervisor Call	SVC <immed_8>	Supervisor Call processor exception	8-bit immediate value encoded in instruction. Formerly SWI. <endianness> can be BE (Big Endian) or LE (Little Endian). 8-bit immediate value encoded in instruction.
	Change processor state	6 CPSID <iflags>	Disable specified interrupts	
		6 CPSIE <iflags>	Enable specified interrupts	
	Set endianness	6 SETEND <endianness>	Sets endianness for loads and saves.	
	Breakpoint	5T BKPT <immed_8>	Prefetch abort <i>or</i> enter debug state	
No Op	No operation	6 NOP	None, might not even consume any time.	
Hint	Set event	T2 SEV	Signal event in multiprocessor system.	No action if not implemented.
	Wait for event	T2 WFE	Wait for event, IRQ, FIQ, Imprecise abort, or Debug entry request.	No action if not implemented.
	Wait for interrupt	T2 WFI	Wait for IRQ, FIQ, Imprecise abort, or Debug entry request.	No action if not implemented.
	Yield	T2 YIELD	Yield control to alternative thread.	No action if not implemented.

Condition Field	
Mnemonic	Description
EQ	Equal
NE	Not equal
CS / HS	Carry Set / Unsigned higher or same
CC / LO	Carry Clear / Unsigned lower
MI	Negative
PL	Positive or zero
VS	Overflow
VC	No overflow
HI	Unsigned higher
LS	Unsigned lower or same
GE	Signed greater than or equal
LT	Signed less than
GT	Signed greater than
LE	Signed less than or equal
AL	Always. Do not use in B{cond}

In Thumb code for processors earlier than ARMv6T2, cond must not appear anywhere except in Conditional Branch (B{cond}) instructions.

In Thumb-2 code, cond can appear in any of these instructions (except CBZ, CBNZ, CPSID, CPSIE, IT, and SETEND).

The condition is encoded in a preceding IT instruction (except in the case of B{cond} instructions).

If IT instructions are explicitly provided in the Assembly language source file, the conditions in the instructions must match the corresponding IT instructions.

ARM architecture versions

4T	All Thumb versions of ARM v4 and above.
5T	All Thumb versions of ARM v5 and above.
6	All Thumb versions of ARM v6 and above.
T2	All Thumb-2 versions of ARM v6 and above.

Proprietary Notice

Words and logos marked with ® or ™ are registered trademarks or trademarks owned by ARM Limited. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This reference card is intended only to assist the reader in the use of the product. ARM Ltd shall not be liable for any loss or damage arising from the use of any information in this reference card, or any error or omission in such information, or any incorrect use of the product.

Document Number

ARM QRC 0006D

Change Log

Issue	Date	Change
A	November 2004	First Release
B	May 2005	RVCT 2.2 SP1
C	March 2006	RVCT 3.0
D	March 2007	RVCT 3.1