

ARM Instruction Set
Quick Reference Card

Key to Tables		
{cond}		Refer to Table Condition Field {cond}
<Oprnd2>		Refer to Table Oprnd2
{field}		Refer to Table Field
S		Sets condition codes (optional)
B		Byte operation (optional)
H		Halfword operation (optional)
T		Forces address translation. Cannot be used with pre-indexed addresses
<a_mode2>		Refer to Table Addressing Mode 2
<a_mode2P>		Refer to Table Addressing Mode 2 (Privileged)
<a_mode3>		Refer to Table Addressing Mode 3
<a_mode4L>		Refer to Table Addressing Mode 4 (Load)
<a_mode4S>		Refer to Table Addressing Mode 4 (Store)
<a_mode5>		Refer to Table Addressing Mode 5
#32bit_Imm		A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits
<reglist>		A comma-separated list of registers, enclosed in braces ({ and })

Operation		Assembler	S updates	Action	Notes
Move	Move	MOV{cond}{S} Rd, <Oprnd2>	N Z C	Rd:= <Oprnd2>	<i>Architecture 3, 3M and 4 only</i> <i>Architecture 3, 3M and 4 only</i> <i>Architecture 3, 3M and 4 only</i> <i>Architecture 3, 3M and 4 only</i> <i>Architecture 3, 3M and 4 only</i> <i>Architecture 3, 3M and 4 only</i> <i>Architecture 3, 3M and 4 only</i>
	NOT	MVN{cond}{S} Rd, <Oprnd2>	N Z C	Rd:= 0xFFFFFFFF EOR <Oprnd2>	
	SPSR to register	MRS{cond} Rd, SPSR		Rd:= SPSR	
	CPSR to register	MRS{cond} Rd, CPSR		Rd:= CPSR	
	register to SPSR	MSR{cond} SPSR{field}, Rm		SPSR:= Rm	
	register to CPSR	MSR{cond} CPSR{field}, Rm		CPSR:= Rm	
	immediate to SPSR flags	MSR{cond} SPSR_f, #32bit_Imm		SPSR:= #32-bit_Imm	
	immediate to CPSR flags	MSR{cond} CPSR_f, #32bit_Imm		CPSR:= #32-bit_Imm	
ALU	Arithmetic				<i>Not in Architecture 1</i> <i>Not in Architecture 1</i> <i>Architecture 3M and 4 only</i> <i>Architecture 3M and 4 only</i> <i>Architecture 3M and 4 only</i> <i>Architecture 3M and 4 only</i> <i>Architecture 3M and 4 only</i>
	Add	ADD{cond}{S} Rd, Rn, <Oprnd2>	N Z C V	Rd:= Rn + <Oprnd2>	
	with carry	ADC{cond}{S} Rd, Rn, <Oprnd2>	N Z C V	Rd:= Rn + <Oprnd2> + Carry	
	Subtract	SUB{cond}{S} Rd, Rn, <Oprnd2>	N Z C V	Rd:= Rn - <Oprnd2>	
	with carry	SBC{cond}{S} Rd, Rn, <Oprnd2>	N Z C V	Rd:= Rn - <Oprnd2> - NOT(Carry)	
	reverse subtract	RSB{cond}{S} Rd, Rn, <Oprnd2>	N Z C V	Rd:= <Oprnd2> - Rn	
	reverse subtract with carry	RSC{cond}{S} Rd, Rn, <Oprnd2>	N Z C V	Rd:= <Oprnd2> - Rn - NOT(Carry)	
	Multiply	MUL{cond}{S} Rd, Rm, Rs	N Z	Rd:= Rm * Rs	
	accumulate	MLA{cond}{S} Rd, Rm, Rs, Rn	N Z	Rd:= (Rm * Rs) + Rn	
	unsigned long	UMULL{cond}{S} RdLo, RdHi, Rm, Rs	N Z	RdHi:= (Rm*Rs)[63:32] RdLo:= (Rm*Rs)[31:0]	
	unsigned accumulate long	UMLAL{cond}{S} RdLo, RdHi, Rm, Rs	N Z	RdLo:= (Rm*Rs)[31:0] + RdLo RdHi:=(Rm*Rs)[63:32] + RdHi + CarryFrom ((Rm*Rs)[31:0] +RdLo))	
	signed long	SMULL{cond}{S} RdLo, RdHi, Rm, Rs	N Z	RdHi:= signed(Rm*Rs)[63:32] RdLo:= signed(Rm*Rs)[31:0]	
	signed accumulate long	SMLAL{cond}{S} RdLo, RdHi, Rm, Rs	N Z	RdLo:= signed(Rm*Rs)[31:0] + RdLo RdHi:= signed(Rm*Rs)[63:32] + RdHi + CarryFrom ((Rm*Rs)[31:0] +RdLo))	
	Compare	CMP{cond} Rd, <Oprnd2>	N Z C V	CPSR flags:= Rn - <Oprnd2>	
	negative	CMN{cond} Rd, <Oprnd2>	N Z C V	CPSR flags:= Rn + <Oprnd2>	
	Logical				
	Test	TST{cond} Rn, <Oprnd2>	N Z C	CPSR flags:= Rn AND <Oprnd2>	
	Test equivalence	TEQ{cond} Rn, <Oprnd2>	N Z C	CPSR flags:= Rn EOR <Oprnd2>	
	AND	AND{cond}{S} Rd, Rn, <Oprnd2>	N Z C	Rd:= Rn AND <Oprnd2>	
	EOR	EOR{cond}{S} Rd, Rn, <Oprnd2>	N Z C	Rd:= Rn EOR <Oprnd2>	
	ORR	ORR{cond}{S} Rd, Rn, <Oprnd2>	N Z C	Rd:= Rn OR <Oprnd2>	
	Bit Clear	BIC{cond}{S} Rd, Rn, <Oprnd2>	N Z C	Rd:= Rn AND NOT <Oprnd2>	
	Shift/Rotate				See Table Oprnd2

ARM Instruction Set Quick Reference Card

Operation		Assembler	Action	Notes
Branch	Branch with link and exchange instruction set	B{cond} label BL{cond} label BX{cond} Rn	R15:= address of label R14:=R15-4, R15:= address of label R15:=Rn, T bit:= Rn[0]	Address calculated pc-relative <i>Architecture 4 with Thumb only to Thumb state; Rn[0] = 1 to ARM state; Rn[0] = 0</i>
Load	Word with user-mode privilege Byte with user-mode privilege signed Halfword signed Multiple Block data operations Increment Before Increment After Decrement Before Decrement After Stack operations and restore CPSR User registers	LDR{cond} Rd, <a_mode2> LDR{cond}T Rd, <a_mode2P> LDR{cond}B Rd, <a_mode2> LDR{cond}BT Rd, <a_mode2P> LDR{cond}SB Rd, <a_mode3> LDR{cond}H Rd, <a_mode3> LDR{cond}SH Rd, <a_mode3> LDM{cond}IB Rd{!}, <reglist>{^} LDM{cond}IA Rd{!}, <reglist>{^} LDM{cond}DB Rd{!}, <reglist>{^} LDM{cond}DA Rd{!}, <reglist>{^} LDM{cond}<a_mode4L> Rd{!}, <reglist> LDM{cond}<a_mode4L> Rd{!}, <reglist+pc>^ LDM{cond}<a_mode4L> Rd, <reglist>^	Rd:= [address] Rd:= [byte value from address] Loads bits 0 to 7 and sets bits 8-31 to 0 Rd:= [signed byte value from address] Loads bits 0 to 7 and sets bits 8-31 to bit 7 Rd:= [halfword value from address] Loads bits 0 to 15 and sets bits 16-31 to 0 Rd:= [signed halfword value from address] Loads bits 0 to 15 and sets bits 16-31 to bit 15 Stack manipulation (pop)	 ! sets the W bit (updates the base register after the transfer) ^ sets the S bit ! sets the W bit (updates the base register after the transfer)
Store	Word with user-mode privilege Byte with user-mode privilege Halfword Multiple Block data operations Increment Before Increment After Decrement Before Decrement After Stack operations User registers	STR{cond} Rd, <a_mode2> STR{cond}T Rd, <a_mode2P> STR{cond}B Rd, <a_mode2> STR{cond}BT Rd, <a_mode2P> STR{cond}H Rd, <a_mode3> STM{cond}IB Rd{!}, <reglist>{^} STM{cond}IA Rd{!}, <reglist>{^} STM{cond}DB Rd{!}, <reglist>{^} STM{cond}DA Rd{!}, <reglist>{^} STM{cond}<a_mode4S> Rd{!}, <reglist> STM{cond}<a_mode4S> Rd{!}, <reglist>^	[address]:= Rd [address]:= byte value from Rd [address]:= halfword value from Rd Stack manipulation (push)	 ! sets the W bit (updates the base register after the transfer) ^ sets the S bit
Swap	Word Byte	SWP{cond} Rd, Rm, [Rn] SWP{cond}B Rd, Rm, [Rn]		<i>Not in Architecture 1 or 2</i> <i>Not in Architecture 1 or 2</i>
Coprocessors	Data operations Move to ARM reg from coproc Move to coproc from ARM reg Load Store	CDP{cond} p<cpnum>, <op1>, CRd, CRn, CRm, <op2> MRC{cond} p<cpnum>, <op1>, Rd, CRn, CRm, <op2> MCR{cond} p<cpnum>, <op1>, Rd, CRn, CRm, <op2> LDC{cond} p<cpnum>, CRd, <a_mode5> STC{cond} p<cpnum>, CRd, <a_mode5>		<i>Not in Architecture 1</i>
Software Interrupt		SWI 24bit_Imm	Causes a software interrupt processor exception	24-bit immediate value encoded within the instruction.

ARM Addressing Modes

Quick Reference Card

Addressing Mode 2	
Immediate offset	[Rn, #+/-12bit_Offset]
Register offset	[Rn, +/-Rm]
Scaled register offset	[Rn, +/-Rm, LSL #5bit_shift_imm] [Rn, +/-Rm, LSR #5bit_shift_imm] [Rn, +/-Rm, ASR #5bit_shift_imm] [Rn, +/-Rm, ROR #5bit_shift_imm] [Rn, +/-Rm, RRX]
Pre-indexed offset	
Immediate	[Rn, #+/-12bit_Offset]!
Register	[Rn, +/-Rm]!
Scaled register	[Rn, +/-Rm, LSL #5bit_shift_imm]! [Rn, +/-Rm, LSR #5bit_shift_imm]! [Rn, +/-Rm, ASR #5bit_shift_imm]! [Rn, +/-Rm, ROR #5bit_shift_imm]! [Rn, +/-Rm, RRX]!
Post-indexed offset	
Immediate	[Rn], #+/-12bit_Offset
Register	[Rn], +/-Rm
Scaled register	[Rn], +/-Rm, LSL #5bit_shift_imm [Rn], +/-Rm, LSR #5bit_shift_imm [Rn], +/-Rm, ASR #5bit_shift_imm [Rn], +/-Rm, ROR #5bit_shift_imm [Rn], +/-Rm, RRX]

Addressing Mode 2 (Privileged)	
Immediate offset	[Rn, #+/-12bit_Offset]
Register offset	[Rn, +/-Rm]
Scaled register offset	[Rn, +/-Rm, LSL #5bit_shift_imm] [Rn, +/-Rm, LSR #5bit_shift_imm] [Rn, +/-Rm, ASR #5bit_shift_imm] [Rn, +/-Rm, ROR #5bit_shift_imm] [Rn, +/-Rm, RRX]
Post-indexed offset	
Immediate	[Rn], #+/-12bit_Offset
Register	[Rn], +/-Rm
Scaled register	[Rn], +/-Rm, LSL #5bit_shift_imm [Rn], +/-Rm, LSR #5bit_shift_imm [Rn], +/-Rm, ASR #5bit_shift_imm [Rn], +/-Rm, ROR #5bit_shift_imm [Rn], +/-Rm, RRX]

Addressing Mode 3 - Signed Byte and Halfword Data Transfer	
Immediate offset	[Rn, #+/-8bit_Offset]
Pre-indexed	[Rn, #+/-8bit_Offset]!
Post-indexed	[Rn], #+/-8bit_Offset
Register	[Rn, +/-Rm]
Pre-indexed	[Rn, +/-Rm]!
Post-indexed	[Rn], +/-Rm

Addressing Mode 5 - Coprocessor Data Transfer	
Immediate offset	[Rn, #+/- (8bit_Offset*4)]
Pre-indexed	[Rn, #+/- (8bit_Offset*4)]!
Post-indexed	[Rn], #+/- (8bit_Offset*4)

Addressing Mode 4 (Load)			
Addressing Mode		Stack Type	
IA	Increment After	FD	Full Descending
IB	Increment Before	ED	Empty Descending
DA	Decrement After	FA	Full Ascending
DB	Decrement Before	EA	Empty Ascending

Addressing Mode 4 (Store)			
Addressing Mode		Stack Type	
IA	Increment After	EA	Empty Ascending
IB	Increment Before	FA	Full Ascending
DA	Decrement After	ED	Empty Descending
DB	Decrement Before	FD	Full Descending

Oprnd2	
Immediate value	#32bit_Imm
Logical shift left	Rm LSL #5bit_Imm
Logical shift right	Rm LSR #5bit_Imm
Arithmetic shift right	Rm ASR #5bit_Imm
Rotate right	Rm ROR #5bit_Imm
Register	Rm
Logical shift left	Rm LSL Rs
Logical shift right	Rm LSR Rs
Arithmetic shift right	Rm ASR Rs
Rotate right	Rm ROR Rs
Rotate right extended	Rm RRX

Field		
Suffix	Sets	
_C	Control field mask bit	(bit 3)
_f	Flags field mask bit	(bit 0)
_s	Status field mask bit	(bit 1)
_x	Extension field mask bit	(bit 2)

Condition Field {cond}	
Suffix	Description
EQ	Equal
NE	Not equal
CS	Unsigned higher or same
CC	Unsigned lower
MI	Negative
PL	Positive or zero
VS	Overflow
VC	No overflow
HI	Unsigned higher
LS	Unsigned lower or same
GE	Greater or equal
LT	Less than
GT	Greater than
LE	Less than or equal
AL	Always