ARM Instruction Set Quick Reference Card

{cond}	Refer to Table Condition Field (cond)	
<oprnd2></oprnd2>	Refer to Table Oprnd2	
{field}	Refer to Table Field	
	Sets condition codes (optional)	
Д	Byte operation (optional)	
н	Halfword operation (optional)	
E	Forces address translation. Cannot be used with pre-indexed addresses	
<a_model></a_model>	Refer to Table Addressing Mode 1	
<a_mode2></a_mode2>	Refer to Table Addressing Mode 2	
<a_mode3></a_mode3>	Refer to Table Addressing Mode 3	
<a_mode4></a_mode4>	Refer to Table Addressing Mode 4	
<a_mode5></a_mode5>	Refer to Table Addressing Mode 5	
<a_mode6></a_mode6>	Refer to Table Addressing Mode 6	
#32_Bit_Immed	A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits	

Operation		Assembler	Supdates	Action	Notes
Move	to register to SPSR rto CPSR ate to SPSR flags ate to CPSR flags	MOV{cond}{S} Rd, <oprnd2> WVN{cond}{S} Rd, <oprnd2> WVN{cond}{S} Rd, <oprnd2> WRS{cond} Rd, SPSR MRS{cond} Rd, CPSR MSR{cond} CPSR{field}, Rm MSR{cond} CPSR{field}, Rm MSR{cond} CPSR{field}, Rm MSR{cond} CPSR_ff, #32_Bit_Immed MSR{cond} CPSR_f, #32_Bit_Immed</oprnd2></oprnd2></oprnd2>	0 0 N N Z Z	Rd:= <opmd2> Rd:= OxFFFFFFF EOR <opmd2> Rd:= SPSR Rd:= CPSR SPSR:= Rm CPSR:= Rm SPSR:= Rm SPSR:= #32_Bit_Immed CPSR:= #32_Bit_Immed</opmd2></opmd2>	Architecture 3, 3M and 4 only
	Add with carry Subtract with carry reverse subtract rever	ADD{cond}{S} Rd, Rn, <oprnd2> ADC{cond}{S} Rd, Rn, <oprnd2> SUB{cond}{S} Rd, Rn, <oprnd2> SBC{cond}{S} Rd, Rn, <oprnd2> SBC{cond}{S} Rd, Rn, <oprnd2> RSB{cond}{S} Rd, Rn, <oprnd2> RSC{cond}{S} Rd, Rn, <oprnd2> RMLA{cond}{S} Rd, Rm, Rs, Rn UMULL{cond}{S} Rd, Rm, Rs, Rn UMULL{cond}{S} RdHi, RdLo, Rm, Rs SMULL{cond}{S} RdHi, RdLo, Rm, Rs SMULL{cond}{S} RdHi, RdLo, Rm, Rs</oprnd2></oprnd2></oprnd2></oprnd2></oprnd2></oprnd2></oprnd2>	>>>>> 000000 NNNNN NNN N N N ZZZZZZ ZZZ Z Z Z	Rd:= Rn + < Opmd2> Rd:= Rn + < Opmd2> + Carry Rd:= Rn - < Opmd2> + Carry Rd:= Rn - < Opmd2> NG(Carry) Rd:= Copmd2> - NOT(Carry) Rd:= < Opmd2> - Rn - NOT(Carry) Rd:= < Opmd2> - Rn - NOT(Carry) Rd:= Rm* Rs Rd:= (Rm* Rs) + Rn RdH:= (Rm* Rs) + Rn RdH:= (Rm* Rs) + RdH:= (Rm* Rs) RdLo:= (Rm* Rs) RdH:= (Rm* Rs) RdH:= (Rm* Rs) RdH:= Signed(Rm* Rs) RdH:= Signed(Rm	Not in Architecture 1 Not in Architecture 1 Architecture 3M and 4 only Architecture 3M and 4 only Architecture 3M and 4 only
				Carry From (Rm*Rs) (31:0)-RadLo)) CPSR flags:= Rn - < Opmd2> CPSR flags:= Rn + < Opmd2> CPSR flags:= Rn AND < Opmd2> CPSR flags:= Rn EOR < Opmd2> Rd:= Rn AND < Opmd2> Rd:= Rn EOR < Opmd2> Rd:= Rn ORR < Opmd2>	Does not update the V flag See Table Oprnd2

ARM Instruction Set Quick Reference Card

				Netes
Operation	-	Assembler	Action	Notes
Branch	Branch with link and exchange instruction set	B{cond} label BL{cond} label BX{cond} Rn	R15:= address R14:=R15, R15:= address R15:=Rn, T bit:= Rn[0]	Architecture 4 with Thumb only Thumb state; Rn[0] = 0 ARM state; Rn[0] =1
Load	Word with user-mode privilege Byte with user-mode privilege signed Halfword signed	LDR (cond) Rd, <a_model> LDR (cond) T Rd, <a_model> LDR (cond) B Rd, <a_model> LDR (cond) BT Rd, <a_mode2> LDR (cond) SB Rd, <a_mode3> LDR (cond) SB Rd, <a_mode3> LDR (cond) SH Rd, <a_mode3> LDR (cond) SH Rd, <a_mode3></a_mode3></a_mode3></a_mode3></a_mode3></a_mode2></a_model></a_model></a_model>	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:= [haftword value from address] Loads bits 0 to 15 and sets bits 16-31 to 0 Rd:= [signed haftword value from address] Loads bits 0 to 15 and sets bits 16-31 to bit 15	Architecture 4 only Architecture 4 only Architecture 4 only
	Block data operations Increment Before Increment After Decrement Before Decrement After Stack operations and restore CPSR User registers	<pre>LDM{cond}IB Rd{!}, <rrgs>{^} LDM{cond}IA Rd{!}, <rrgs>{^} LDM{cond}DB Rd{!}, <rrgs>{^^} LDM{cond}DB Rd{!}, <rrgs>{^^} LDM{cond}DA Rd{!}, <rrgs>{^^} LDM{cond} <a_mode4> Rd{!}, <rrgisters> LDM{cond} <a_mode4> Rd{!}, <rrgisters+pc> LDM{cond} <a_mode4> Rd, <rrgisters+pc> LDM{cond} <a_mode4> Rd, <rrgisters>^</rrgisters></a_mode4></rrgisters+pc></a_mode4></rrgisters+pc></a_mode4></rrgisters></a_mode4></rrgs></rrgs></rrgs></rrgs></rrgs></pre>	tion (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 After Decrement After Stack operations User registers	<pre>STR{cond} Rd, <a_model> STRT{cond} Rd, <a_model> STRB{cond} Rd, <a_model> STRBT{cond} Rd, <a_model> STRBT{cond} Rd, <a_mode2> STRT{cond} Rd, <a_mode3> STR{cond} IB Rd{!}, <aregisters>{^} STM{cond} IB Rd{!}, <aregisters>{^} STM{cond} DB Rd{!}, <aregisters>{^} STM{cond} DB Rd{!}, <aregisters>{^} STM{cond} DB Rd{!}, <aregisters>{^} STM{cond} PA Rd{!}, <aregisters>{^} STM{cond} AR Rd{!}</aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></aregisters></a_mode3></a_mode2></a_model></a_model></a_model></a_model></pre>	[address]:= Rd [address]:= byte value from Rd [address]:= halfword value from Rd Stack manipulation (push)	Architecture 4 only ! sets the W bit (updates the base register after the transfer ^ sets the S bit
Swap	Word Byte	<pre>SWP{cond} Rd, Rm, [Rn] SWP{cond}B Rd, Rm, [Rn]</pre>		Not in Architecture 1 or 2 Not in Architecture 1 or 2
Coprocessors	Data operations Move to ARM reg from ARM reg Load Store	CDP{cond} peopnum>, copl>, CRd, CRn, CRm, cop2> MRC{cond} peopnum>, copl>, Rd, CRn, CRm, cop2> MCR{cond} peopnum>, copl>, Rd, CRn, CRm, cop2> LDC{cond} peopnum>, CRd, ca_mode6> STC{cond} peopnum>, CRd, ca_mode6>		Not in Architecture 1
Software Interrupt		SWI #24_Bit_Value		24-bit immediate value



ARM Addressing Modes Quick Reference Card

Addressing Mode 1	
Immediate offset	[Rn, #+/-12_Bit_Offset]
Register offset	[Rn, +/-Rm]
Scaled register offset	[Rn, +/-Rm, LSL #shift_imm]
	[Rn, +/-Rm, LSR #shift_imm]
	[Rn, +/-Rm, ASR #shift_imm]
	[Rn, +/-Rm, ROR #shift_imm]
	[Rn, +/-Rm, RRX]
Pre-indexed offset	
Immediate	[Rn, #+/-12_Bit_Offset]!
Register	[Rn, +/-Rm]!
Scaled register	[Rn, +/-Rm, LSL #shift_imm]!
	[Rn, +/-Rm, LSR #shift_imm]!
	[Rn, +/-Rm, ASR #shift_imm]!
	[Rn, +/-Rm, ROR #shift_imm]!
	[Rn, +/-Rm, RRX]!
Post-indexed offset	
Immediate	[Rn], #+/-12_Bit_Offset
Register	[Rn], +/-Rm
Scaled register	[Rn], +/-Rm, LSL #shift_imm
	[Rn], +/-Rm, LSR #shift_imm
	[Rn], +/-Rm, ASR #shift_imm
	[Rn], +/-Rm, ROR #shift_imm
	[Rn, +/-Rm, RRX]

Addisoning Mode 2	
Addressing Mode 2	
Immediate offset	[Rn, #+/-12_Bit_Offset]
Register offset	[Rn, +/-Rm]
Scaled register offset	[Rn, +/-Rm, LSL #shift_imm]
	[Rn, +/-Rm, LSR #shift_imm]
	[Rn, +/-Rm, ASR #shift_imm]
	[Rn, +/-Rm, ROR #shift_imm]
	[Rn, +/-Rm, RRX]
Post-indexed offset	
Immediate	[Rn], #+/-12_Bit_Offset
Register	[Rn], +/-Rm
Scaled register	[Rn], +/-Rm, LSL #shift_imm
	[Rn], +/-Rm, LSR #shift_imm
	[Rn], +/-Rm, ASR #shift_imm
	[Rn], +/-Rm, ROR #shift_imm
	[Rn, +/-Rm, RRX]

Immediate offset	Immediate offset [Rn, #+/-8_Bit_Offset]
Pre-indexed	[Rn, #+/-8_Bit_Offset]!
Post-indexed	[Rn], #+/-8_Bit_Offset
Register	[Rn, +/-Rm]
Pre-indexed	[Rn, +/-Rm]!
Post-indexed	[Rn], +/-Rm

Register	[Rn, +/-Rm]		
Pre-indexed	[Rn, +/-Rm]!		
Post-indexed	[Rn], +/-Rm	Addr	ressir
		Ad	Address
Addressing Mode 6 - Coprocessor Data Transfer		IA	_
Immediate offset	[Rn, #+/-(8_Bit_Offset*4)]	IB	~
Pre-indexed	[Rn, #+/-(8_Bit_Offset*4)]!	DA	_
Post-indexed	[Rn], #+/-(8_Bit_Offset*4)	DB	~

Oprnd2	
Immediate value	#32_Bit_Immed
Logical shift left	Rm LSL #5_Bit_Immed
Logical shift right	Rm LSR #5_Bit_Immed
Arithmetic shift right	Rm ASR #5_Bit_Immed
Rotate right	Rm ROR #5_Bit_Immed
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
ا	Control field mask bit (bit 3)
J.	Flags field mask bit (bit 0)
ω	Status field mask bit (bit 1)
×ı	Extension field mask bit (bit 2)
Condition Field (cond)	eld {cond}
Suffix	Description
ÕЭ	Equal
NE	Not equal
CS	Unsigned higher or same
GG	Unsigned lower
MI	Negative
PL	Positive or zero
NS	Overflow
VC	No overflow
HI	Unsigned higher

O H	Edual
NE	Not equal
CS	Unsigned higher or same
SS	Unsigned lower
MI	Negative
PL	Positive or zero
NS	Overflow
VC	No overflow
IH	Unsigned higher
LS	Unsigned lower or same
GE	Greater or equal
LT	Less than
E	Greater than
ILE	Less than or equal
AL	Always
ng Mode 4	
sing Mode	Stack Type
Increment After	FD Full Descending
Increment Before	ED Empty Descending

Address	ddressing Mode 4		
Addre	dressing Mode	Stack 1	Гуре
IA	Increment After	ПĒ	Full Descending
IB	Increment Before	ED	Empty Descending
DA	Decrement After	FA	Full Ascending
DB	Decrement Before	EA	Empty Ascending

Address	Addressing Mode 5		
Addre	Addressing Mode	Stack Type	Гуре
IA	Increment After	EA	Empty Ascending
IB	Increment Before	FA	Full Ascending
DA	Decrement After	ED	Empty Descending
DB	Decrement Before	FD	Full Descending