Command	Detail	OP	Cycles	Opcode	Arch
ADCccS R0, R1, *R2* ,*shft*	Add with Carry	R0 = R1+R2+C	1	0101	
ADDccS R0, R1, *R2* ,*shft*	Add	R0 = R1+R2	1	0100	
ANDccS R0, R1, *R2* ,*shft*	Bitwise And	R0 = R1 and R2	1	0000	
Bcc addr	Branch (JP)	R15=addr	3		
BICccS R0, R1, *R2* ,*shft*	Bit Clear	R0 = R1 and (CPL R2)	1	1110	
BXJcc R0	Branch and change to Jazelle state				6
BKPT imm	Breakpoint				5
BLcc addr	Branch and Link (CALL)	R14=R15… R15=addr	3		
BLX addr,BLXcc R0	Branch , link and exchange				5Tb
BXcc R0	Branch and exchange	R15=Rn… Tbit=Rn[0]			5tb
CDPcc #,e,Crd,Crn,Crm,e2 CDO	Coprocessor data processing				2,5
CLZcc R0, R1	Count Leading Zeros				5
CMNccP R1, *R2* ,*shft*	Compare Negative	flags=R1+R2	1	1001	
CMPccP R1, *R2* ,*shft*	Compare	flags=R1-R2	1	1010	
CPSeeff #n	Change Processor state				6
CPYcc R0, R1	Copy one register to another	R0=R1			6
EORccS R0, R1, *R2* ,*shft*	Exclusive OR (XOR)	R0 = R1 xor R2	1	0001	
LDCccLTN #,Crd,addr,L LDC2	Load Coprocessor				2,5
LDMccmm R0!,{R1,R2...R3}	Load Multiple (POP)	Move R1,R2...R3→(R0)	1+		
LDRccBT R0,addr,*shft*	LoaD Register (B=8 bit / T=access in user mode)	R0=(addr)	3+	psuedo	
LDRccH R0,addr,*shft*	LoaD Register (16 bit)	R0=(addr)	3+		
LDRccD R0,addr	LoaD Register (64 bit)	R0=(addr),R1=(addr+4)	3+		5TE
LDREXcc R0,R1	LoaD Register and set memory exclusive	R0=(R1)	3+		6
LDRccSB R0,addr,*shft*	LoaD Register (8 bit signed)	R0=(addr)	3+		4
LDRccSH R0,addr,*shft*	LoaD Register (16 bit signed)	R0=(addr)	3+		4
MAR	Mover from registers to 40 bit acc				Xscale
MCRcc #,e,Rd,Crn,Crm,e2 MCR2	Move from registers to coprocessor				2,5,5Ed
MCRRcc #,e,Rd,Rn,Crn,Crm,e2 M	Move from 2 registers to coprocessor				5TE,6
MIA,MIAPH,MIAxy	Multiply with internal 40 bit accumulate				Xscale
MLAccS R0,R1,R2,R3	Multiply with Accumulate	R0=(R1*R2)+R3	16		2
MOVccS R0, *R2* ,*shft*	Move	R0 = R2	1	1101	
MRA	Multiply from 40 bit accumulator to registers				Xscale
MRCcc #,e,Rd,Crn,Crm,e2 ,MRC	Coprocessor Register transfer				2,5
MRRCcc #,e,Rd,Rn,Crm, MRC2	Move from coprocessor to 2 regs				5E
MRScc R0,flags	Move from CPSR/SPSR to register	=PSR			3
MSRcc fields,#n/R0	Move from register to CPSR	PSR=Rm			3
MULccS R0, R1, R2	Multiply	R0=R1*R2	16		2
MVNccS R0, R2 ,*shft*	Move Negative	R0 = -R2	1	1111	
ORRccS R0, R1, R2 ,*shft*	Inclusive Or	R0 = R1 or R2	1	1100	
PKHBTcc R0, R1, R2 ,*shft*	Pack Halfword Bottom/Top (L from R1 / H from R2)	R0=R2H+R1L			6
PKHTBcc R0, R1, R2 ,*shft*	Pack Halfword Top/Bottom (H from R1 / L from R2)	R0=R1H+R2L			6
PLD mode	Cache Preload				5E
QADDcc R0, R1, R2	Saturating Arithmatic				5Exp
QADD16cc R0, R1, R2	Saturating Arithmatic (16 bit)				6
QADD8cc R0, R1, R2	Saturating Arithmatic (8 bit)				6
QADDSUBXcc R0, R1, R2	Saturating Add and Subtract with Exchange				6
QDADDcc R0, R1, R2	Saturating Double and Add				5TE
QDSUBcc R0, R1, R2	Saturating Double and Subtract				5TE
QSUBcc R0, R1, R2	Saturating Subtract				5TE
QSUB16cc R0, R1, R2	Saturating Subtract (16 bit)				6
QSUB8cc R0, R1, R2	Saturating Subtract (8 bit)				6
QSUBADDXcc R0, R1, R2	Saturating Add and Subtract with Exchange				6
REVcc R0, R1	reverses the byte order in a 32-bit register.				6
REV16cc R0, R1	reverses the byte order in a 16-bit register.				6
REVSHcc R0, R1	reverses the byte order in a 16-bit register, and sign extend				6
RFE<mode> R0!	Return From Exception				6
RSBccS R0, R1, R2 ,*shft*	Reverse SuBtract	R0 = R2-R1	1	0011	
RSCccS R0, R1, R2 ,*shft*	Reverse Subtract with Carry	R0 = R2-R1+C-1	1	0111	
SADD16cc R0, R1, R2	Signed Add two 16 bit numbers				6
SADD8cc R0, R1, R2	Signed Add four 8-bit signed integer additions				6
SADDSUBXcc R0, R1, R2	Signed 16-bit Add and Subtract with Exchange				6
SBCccS R0, R1, R2 ,*shft*	Subtract with carry	R0 = R1-R2+C-1	1	0110	
SELcc R0, R1, R2	Select bytes from R1/R2 based on GE flags				6
SETEND <endian>	Set Endian mode				6
SHADD16cc R0, R1, R2	Signed Halving Add (16 bit)				6
SHADD8cc R0, R1, R2	Signed Halving Add (8 bit)				6
SHADDSUBXcc R0, R1, R2	Signed Halving Add and Subtract with Exchange (16 bit)				6
SHSUB16cc R0, R1, R2	Signed Halving Subtract (16 bit)				6
SHSUB8cc R0, R1, R2	Signed Halving Subtract (8 bit)				6
SHSUBADDXcc R0, R1, R2	Signed Halving Subtract and Add with Exchange (16 bit)				6
SMLALccS R0L, R1H, R2,R3	Signed Multiply-accumulate Long				
SMLALxycc R0L, R1H, R2,R3	Signed Multiply-accumulate Long				5TE
SMLAxycc	Signed Multiply-accumulate				5TE
SMLADXcc	Signed Multiply-accumulate Dual				6
SMLAWycc	Signed Multiply-accumulate Word B and T				5ExP
SMLSDXcc R0, R1, R2,R3	Signed Multiply Subtract accumulate Dual				6
SMLSLDXcc R0, R1, R2,R3	Signed Multiply Subtract accumulate LongDual				6
SMMLARcc R0, R1, R2,R3	Signed Most significant word Multiply Accumulate				6
SMMLSRcc R0, R1, R2,R3	Signed Most significant word Multiply Subtract				6
SMULLRcc R0, R1, R2	Signed Multiply (R=Round)				6
SMUADXcc R0, R1, R2	Signed Dual Multiply Add				6
SMULXYcc R0, R1, R2	Signed Multiply BB , BT , TB , or TT				ARMv5TE
SMULLcc R0L, R1H, R2,R3	Signed Multiply Long				ARMv5TE
SMULWYcc R0, R1, R2	Signed Multiply Word B and T				ARMv5TE
SMUSDXcc R0, R1, R2	Signed Dual Multiply Subtract				6
SRS<Mode> #mode!	Store Return State				6
SSAT16cc R0,#n, R1,*shft*	Signed Saturate (16 bit)				6
SSATcc R0,#n, R1,*shft*	Signed Saturate				6
SSUB16cc R0, R1, R2	Signed Subtract (16 bit)				6
SSUB8cc R0, R1, R2	Signed Subtract (8 bit)				6
SSUBADDXcc R0, R1, R2	Signed Subtract and Add with Exchange (16 bit)				6
STCccLTN #,Crd,addr,L STC2	Store to Coprosessor				2,5ExP
STMccmm R0,{R1,R2...R3}!	Store Multiple (PUSH)	Restore (R0)-> R1,R2...	2+		
STRccBT R0,(addr),*shft*	Store Register	(addr)=R0	2+		
STRccD R0,(addr)	Store Register (64 bit)	(addr)=R0,(addr+4)=R1	2+		ARMv5TE
STRccH R0,(addr)	Store Register (16 bit)	(addr)=R0	2+ (H=4+)		
STREXcc R0,R1,R2	Store Register Exclusive				6
SXTABcc R0,R1,R2,*shft*	Extract an 8 bit value, and sign extend				6
SXTAB16cc R0,R1,R2,*shft*	Extract two 8 bit value, and sign extend to 16 bits				6
SXTAHcc R0,R1,R2,*shft*	Extract a 16 bit value, and sign extend				6
SXTBcc R0,R1,*shft*	Take a 8-bit value from a register and sign extends it to 32 bits.				6
SXTB16cc R0,R1,*shft*	Take two 8-bit value from a register and sign extends it to 16 bits.				6
SXTHcc R0,R1,*shft*	Take two 16-bit value from a register and sign extend to 32 bits				6
SUBccS R0, R1, R2 ,*shft*	Subtract	R0 = R1-R2	1	0010	
SWIcc #n	Software Interrupt (RST)		3		
SWPccB r0,r1,[base]	Load r0 from [base],store r1 in [base]	Rd=Rn… Rn=Rd			3
TEQccP R1, R2 ,*shft*	Test inverted	flags=R1 xor R2	1	1001	
TSTccP R1, R2 ,*shft*	Test Masked	flags=R1 AND R2	1	1000	
UADD16cc R0,R1,R2	Unsigned Add (16 bit)				6
UADD8cc R0,R1,R2	Unsigned Add (8 bit)				6
UADDSUBXcc R0,R1,R2	Unsigned Add and Subtract with Exchange				6
UHADD16cc R0,R1,R2	Unsigned Halving Add (16 bit)				6
UHADD8cc R0,R1,R2	Unsigned Halving Add (8 bit)				6
UHSUB16cc R0,R1,R2	Unsigned Halving Subtract (16 bit)				6
UHSUB8cc R0,R1,R2	Unsigned Halving Subtract (8 bit)				6
USUBADDXcc R0,R1,R2	Unsigned Subtract and Add with Exchange				6
UMALccS R0L, R1H, R2,R3	Unsigned Multiply Accumulate Long				
UMULLccS R0L, R1H, R2,R3	Unsigned Multiply Long				6
UQADD16cc R0,R1,R2	Unsigned Saturating Add (16 bit)				6
UQADD8cc R0,R1,R2	Unsigned Saturating Add (8 bit)				6
UQADDSUBXcc R0,R1,R2	Unsigned Saturating Add and Subtract with Exchange				6
UQSUB16cc R0,R1,R2	Unsigned Saturating Subtract (16 bit)				6
UQSUB8cc R0,R1,R2	Unsigned Saturating Subtract (8 bit)				6
UQSUBADDXcc R0,R1,R2	Unsigned Saturating Subtract and Add with Exchange				6
USAD8cc R0,R1,R2	Unsigned Sum of Absolute Differences				6
USADA8cc R0,R1,R2,R3	Unsigned Sum of Absolute Differences and Accumulate				6
USATcc R0,#n, R1,*shft*	Unsigned Saturate				6
USAT16cc R0,#n, R1,*shft*	Unsigned Saturate (16 bit)				6
USUB16cc R0,R1,R2	Unsigned Subtract (16 bit)				6
USUB8cc R0,R1,R2	Unsigned Subtract (8 bit)				6
USUBADDXcc R0,R1,R2	Unsigned Subtract and Add with Exchange				6
UXTABcc R0,R1,R2,*shft*	Extract an 8 bit value and Zero extend				6
UXTAB16cc R0,R1,R2,*shft*	Extract two 8 bit values and Zero extend				6
UXTAHcc R0,R1,R2,*shft*	Extract an 16 bit value and Zero extend				6
UXTBcc R0,R1,*shft*	Extract an 8 bit value and Zero extend				6
UXTB16cc R0,R1,*shft*	Extract two 8 bit values and Zero extend				6
UXTHcc R0,R1,*shft*	Extract a 16 bit value and Zero Extend				6
ADRcc Rn,addr	Load relative address into register	R0=addr		psuedo	
ADRccL Rn,label	Load Long relative address into register			psuedo	
NOP	no operation			psuedo	
http://www.chibialiens.com/arm					