Addressing Mode 2		Addressing Mode 2 (Pr	Addressing Mode 2 (Privileged)			
Immediate offset	[Rn, #+/-12bit_Offset]	Immediate offset	[Rn, #+/-12bit_Offset]			
Register offset	[Rn, +/-Rm]	Register offset	[Rn, +/-Rm]			
Scaled register offset	[Rn, +/-Rm, LSL #5bit_shift_imm]	Scaled register offset	[Rn, +/-Rm, LSL #5bit_shift_imm]			
	[Rn, +/-Rm, LSR #5bit_shift_imm]		[Rn, +/-Rm, LSR #5bit_shift_imm]			
	[Rn, +/-Rm, ASR #5bit_shift_imm]		[Rn, +/-Rm, ASR #5bit_shift_imm]			
	[Rn, +/-Rm, ROR #5bit_shift_imm]		[Rn, +/-Rm, ROR #5bit_shift_imm]			
	[Rn, +/-Rm, RRX]		[Rn, +/-Rm, RRX]			
Pre-indexed offset		Post-indexed offset	Post-indexed offset			
Immediate	[Rn, #+/-12bit_Offset]!	Immediate	[Rn], #+/-12bit_Offset			
Register	[Rn, +/-Rm]!	Register	[Rn], +/-Rm			
Scaled register	[Rn, +/-Rm, LSL #5bit_shift_imm]!	Scaled register	[Rn], +/-Rm, LSL #5bit_shift_imm			
	[Rn, +/-Rm, LSR #5bit_shift_imm]!		[Rn], +/-Rm, LSR #5bit_shift_imm			
	[Rn, +/-Rm, ASR #5bit_shift_imm]!		[Rn], +/-Rm, ASR #5bit_shift_imm			
	[Rn, +/-Rm, ROR #5bit_shift_imm]!		[Rn], +/-Rm, ROR #5bit_shift_imm			
	[Rn, +/-Rm, RRX]!		[Rn, +/-Rm, RRX]			
Post-indexed offset		Addressing Mode 4	Stack Type Load	Stack Type Store		
Immediate	[Rn], #+/-12bit_Offset	IA Increment After	FD Full Descending	EA Empty Ascending		
Register	[Rn], +/-Rm	IB Increment Before	ED Empty Descending	FA Full Ascending		
Scaled register	[Rn], +/-Rm, LSL #5bit_shift_imm	DA Decrement After	FA Full Ascending	ED Empty Descending		
	[Rn], +/-Rm, LSR #5bit_shift_imm	DB Decrement Before	EA Empty Ascending	FD Full Descending\		
	[Rn], +/-Rm, ASR #5bit_shift_imm	Oprnd2				
	[Rn], +/-Rm, ROR #5bit_shift_imm	Immediate value	#32bit_Imm			
	[Rn, +/-Rm, RRX]	Logical shift left	Rm LSL #5bit_Imm			
Addressing Mode 3 - Signed Byte and Halfword Data Transfer		Logical shift right	Rm LSR #5bit_Imm			
Immediate offset	[Rn, #+/-8bit_Offset]	Arithmetic shift right	Rm ASR #5bit_Imm			
Pre-indexed	[Rn, #+/-8bit_Offset]!	Rotate right	Rm ROR #5bit_Imm			
Pre-indexed	[Rn], #+/-8bit_Offset	Register	Rm			
Register	[Rn, +/-Rm]	Logical shift left	Rm LSL Rs			
Pre-indexed	[Rn, +/-Rm]!	Logical shift right	Rm LSR Rs			
Post-indexed	[Rn], +/-Rm	Arithmetic shift right	Rm ASR Rs			
Addressing Mode 5 - Coprocessor Data Transfer		Rotate right	Rm ROR Rs	Rm ROR Rs		
Immediate offset	[Rn, #+/-(8bit_Offset*4)]	Rotate right extended	Rm RRX			
Pre-indexed	[Rn, #+/-(8bit_Offset*4)]!	Field				
Post-indexed	[Rn], #+/-(8bit_Offset*4)	Suffix	Sets			
Move		_c	Control field mask bit (bit 3)			
Move	MOV{cond}{S} Rd, <oprnd2></oprnd2>	_f	Flags field mask bit (bit 0)			
11046						





## **Quick Reference**

Sistemas Embebidos I ISEL - DEETC Inverno 2010/2011 V1.0 Nuno Cancelo LEIC 31401

## **Current Program Status Register (CPSR)**

Conjunto de instruções Thumb — Modo de funcionamento do processador

Zero

Negative

Montane   Mont	Move		Load		Store		
Actionable	Move	MOV{cond}{S} Rd, <oprnd2></oprnd2>	Word	LDR{cond} Rd, <a_mode2></a_mode2>	Word	STR{cond} Rd, <a_mode2></a_mode2>	
Mode	Move NOT	MVN{cond}{S} Rd, <oprnd2></oprnd2>	Word user privilege	LDR{cond}T Rd, <a_mode2p></a_mode2p>	Word user privilege	STR{cond}T Rd, <a_mode2p></a_mode2p>	
Mathitude   MacComp(1) S Rd, Rn, -Copma(2)	Arithmetic		Byte	LDR{cond}B Rd, <a_mode2></a_mode2>	Byte	STR{cond}B Rd, <a_mode2></a_mode2>	
Subtract with rarry   SB(cond)(5) Rd, Rn, <0pmd2>   Halfword   LDR(cond)H Rd, <a_mode3>   Multiple  </a_mode3>	Add	ADD{cond}{S} Rd, Rn, <oprnd2></oprnd2>	Byte user privilege	LDR{cond}BT Rd, <a_mode2p></a_mode2p>	Byte user privilege	STR{cond}BT Rd, <a_mode2p></a_mode2p>	
Subtract with carry   SBC(cand)(\$) Rd, Rn, <opmd2></opmd2>	Add with carry	ADC(cond)(S) Rd, Rn, <oprnd2></oprnd2>	Byte signed	LDR{cond}SB Rd, <a_mode3></a_mode3>	Halfword	STR{cond}H Rd, <a_mode3></a_mode3>	
Sub reverse sub   SB(cond){S} Rd, Rn, <opmd2>   Multiple   Increment Before   STM(cond)B Rd(I), <reglist>(^)   Sub rev. Sub. w carry   RSC(cond){S} Rd, Rn, <opmd2>   Block data operations   Increment After   Increment After   STM(cond)B Rd(I), <reglist>(^)   Condition   Condition</reglist></opmd2></reglist></opmd2>	Subtract	SUB{cond}{S} Rd, Rn, <oprnd2></oprnd2>	Halfword	LDR{cond}H Rd, <a_mode3></a_mode3>	Multiple		
Sub rev. Sub. v. carry	Subtract with carry	SBC{cond}{S} Rd, Rn, <oprnd2></oprnd2>	Halfword signed	LDR{cond}SH Rd, <a_mode3></a_mode3>	Block data operation	ns	
Multiply	Sub reverse sub	RSB{cond}{S} Rd, Rn, <oprnd2></oprnd2>	Multiple		Increment Before	STM{cond}IB Rd{!}, <reglist>{^}</reglist>	
MLA(cond){S} Rd, Rm, Rs, Rn	Sub rev. Sub. w carry	RSC{cond}{S} Rd, Rn, <oprnd2></oprnd2>	Block data operations		Increment After	STM{cond}IA Rd{!}, <reglist>{^}</reglist>	
Decrement Before   LDM(cond)DB Rd(f), regilat>\(^{\chick}\)   Slack operations   STM(cond)\(^{\chick}\) a, mode4S> Rd(f), regilat>\(^{\chick}\)   Slack operations   STM(cond)\(^{\chick}\) a, mode4S> Rd(f), regilat>\(^{\chick}\)   Slack operations   STM(cond)\(^{\chick}\) a, mode4S> Rd(f), regilat>\(^{\chick}\)   SWAP	Multiply	MUL{cond}{S} Rd, Rm, Rs	Increment Before	LDM{cond}IB Rd{!}, <reglist>{^}</reglist>	Decrement Before	STM{cond}DB Rd{!}, <reglist>{^}</reglist>	
Longrand accum long UMLAL (cond) (5) Ratlo, RdHi, Rm, Rs Stack operations LDM (cond) DA Rd(t), <pre></pre>	accumulate	MLA{cond}{S} Rd, Rm, Rs, Rn	Increment After	LDM{cond}IA Rd{!}, <reglist>{^}</reglist>	Decrement After	STM{cond}DA Rd{!}, <reglist>{^}</reglist>	
Signed long SMULL(cond){S} RdLo, RdHi, Rm, Rs Stack operations LDM(cond) <mode4l> Rd{1}, <reglist>&gt; SWAP  SWAC  SWAP</reglist></mode4l>	unsigned long	UMULL{cond}{S} RdLo, RdHi, Rm, Rs	Decrement Before	LDM{cond}DB Rd{!}, <reglist>{^}</reglist>	Stack operations	STM{cond} <a_mode4s> Rd{!}, <reglist></reglist></a_mode4s>	
Signed accum long SMLAL(cond)(S) RdLo, RdHi, Rm, Rs Stack restoreCPSR LDM(cond) <mode4l> Rd(f), &lt; reglist+pc&gt;^ Word SWP(cond) Rd, Rm, [Rn]  Compare CMP(cond) Rd, <opmd2> User registers LDM(cond) <mode4l> Rd, &lt; reglist&gt;^ Software Interrupt  Software Interrupt  Software Interrupt  Software Interrupt  Software Interrupt  SWI 24bit_Imm  Set TST(cond) Rn, <opmd2> EQ Equal  Key to Tables  Test quivalence TEQ(cond) Rn, <opmd2> NE Not equal  Condition Field (cond)  AND AND(cond)(S) Rd, Rn, <opmd2> CS Unsigned higher or same  CORR EDR(cond)(S) Rd, Rn, <opmd2> MI Negative  ORR ORR(cond)(S) Rd, Rn, <opmd2> MI Negative  PL Positive or zero   Bit Clear Bic(cond)(S) Rd, Rn, <opmd2> PL Positive or zero   Bit Clear Bic(cond) label  VC No overflow   VS Overflow  Software Interrupt  SWI 24bit_Imm  SWI</opmd2></opmd2></opmd2></opmd2></opmd2></opmd2></mode4l></opmd2></mode4l>	unsigned accum long	UMLAL{cond}{S} RdLo, RdHi, Rm, Rs	Decrement After	LDM{cond}DA Rd{!}, <reglist>{^}</reglist>	User registers	STM{cond} <a_mode4s> Rd{!}, <reglist>^</reglist></a_mode4s>	
Compare negative CMP(cond) Rd, <opmd2> Condition Field (cond) Compare negative CMN(cond) Rd, <opmd2> Condition Field (cond) Compare negative CMN(cond) Rd, <opmd2> Condition Field (cond) Software Interrupt SMI 24bit_Imm  Test Condition Rick Rick Condition Rick C</opmd2></opmd2></opmd2>	signed long	SMULL{cond}{S} RdLo, RdHi, Rm, Rs	Stack operations	LDM{cond} <a_mode4l> Rd{!}, <reglist></reglist></a_mode4l>	SWAP		
Compare negative CMN(cond) Rd, <opmd2> Condition Field (cond) Suffix Description Software Interrupt SWI 24bit_Imm  Test TST(cond) Rn, <opmd2> EQ Equal Key to Tables  Test equivalence TEQ(cond) Rn, <opmd2> NE Not equal (cond) Refer to Table Condition Field (cond)  AND AND(cond)(S) Rd, Rn, <opmd2> NE Not equal (cond) Refer to Table Condition Field (cond)  AND AND(cond)(S) Rd, Rn, <opmd2> CS Unsigned higher or same (field) Refer to Table Domd2  EGR EGR(cond)(S) Rd, Rn, <opmd2> CC Unsigned lower (field) Refer to Table Field  ORR ORR(cond)(S) Rd, Rn, <opmd2> MI Negative (regilst&gt; A comma-separated list of registers, enclosed in braces ( { and } ) )  Branch(Jump)  Branch B(cond) label VC No overflow 432bit_Imm A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits  Branch B(cond) label VC No overflow (a_mod2)&gt; Refer to Table Addressing Mode 2 (Privileged)  Exch instricts BX(cond) Rn  COprocessors  Data operations CDP(cond) p&lt;-cpnum&gt;, <op1>, Rd, CRn, CRn, <op>&gt; LT Less than (a_mode5&gt; Refer to Table Addressing Mode 4 (Load/Store)  The Coprocessor Refer to Table Addressing Mode 5 (Field) Refer to Table Add</op></op1></opmd2></opmd2></opmd2></opmd2></opmd2></opmd2></opmd2>	signed accum long	SMLAL{cond}{S} RdLo, RdHi, Rm, Rs	Stack restoreCPSR	LDM{cond} <a_mode4l> Rd{!}, <reglist+pc>^</reglist+pc></a_mode4l>	Word	SWP{cond} Rd, Rm, [Rn]	
Suffix   Description   Software Interrupt   SWI 24bit_Imm	Compare	CMP{cond} Rd, <oprnd2></oprnd2>	User registers	LDM{cond} <a_mode4l> Rd, <reglist>^</reglist></a_mode4l>	Byte	SWP{cond}B Rd, Rm, [Rn]	
Fest TST(cond) Rn, <oprnd2> EQ Equal (cond) Rn, <oprnd2> NE Not equal (cond) Refer to Table Condition Field (cond) Refer to Table Oprnd2 Refer to Table Addressing Mode 2 Refer to Table Addressing Mode 2 Refer to Table Addressing Mode 3 Refer to Table Addressing Mode 4 (Load/Store) Refer to Table Addressing Mode 5 Refer to Table Ad</oprnd2></oprnd2>	Compare negative CMN{cond} Rd, <oprnd2></oprnd2>		Condition Field {cond}		Software Interrupt		
TEST equivalence TEQ(cond) Rn, <oprmd2> NE Not equal  AND AND(cond){S} Rd, Rn, <oprmd2> CS Unsigned higher or same CDR EOR EOR(cond){S} Rd, Rn, <oprmd2> CC Unsigned lower CDR ORR(cond){S} Rd, Rn, <oprmd2> CC Unsigned lower CDR ORR(cond){S} Rd, Rn, <oprmd2> MI Negative CDR DORR ORR(cond){S} Rd, Rn, <oprmd2> MI Negative CDR Combility Clear CDR Cond){S} Rd, Rn, <oprmd2> MI Negative CDR Cond){S} Rd, Rn, <oprmd2> CDR Cond){S} Rd, Rn, <oprmd2> MI Negative CDR Cond){S} Rd, Rn, <oprmd2> CDR Cond){S} Refer to Table Addressing Mode 2 CDR Cond){S} Refer to Table Addressing Mode 3 CDR Cond){S} Refer to Table Addressing Mode 3 CDR Cond){S} Refer to Table Addressing Mode 4 CDR Cond){S} Refer to Table Addressing Mode 5 CDR Con</oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2></oprmd2>	Logical		Suffix	Description	Software Interrupt	SWI 24bit_Imm	
AND	Test	TST{cond} Rn, <oprnd2></oprnd2>	EQ	Equal	Key to Tables		
EOR EOR(cond)(S) Rd, Rn, <oprnd2> CC Unsigned lower (field) Refer to Table Field  ORR ORR(cond)(S) Rd, Rn, <oprnd2> MI Negative &lt; reglist&gt; A comma-separated list of registers, enclosed in braces ({ and } )  Bit Clear BIC(cond)(S) Rd, Rn, <oprnd2> PL Positive or zero</oprnd2></oprnd2></oprnd2>	Test equivalence	TEQ{cond} Rn, <oprnd2></oprnd2>	NE	Not equal	{cond}	Refer to Table Condition Field {cond}	
ORR ORR(cond){S} Rd, Rn, <oprnd2> MI Negative &lt; reglist&gt; A comma-separated list of registers, enclosed in braces ( { and } } Bit Clear  BIC{cond}{S} Rd, Rn, <oprnd2> PL Positive or zero #32bit_Imm A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits</oprnd2></oprnd2>	AND	AND{cond}{S} Rd, Rn, <oprnd2></oprnd2>	CS	Unsigned higher or same	<oprnd2></oprnd2>	Refer to Table Oprnd2	
Bit Clear BIC{cond}{S} Rd, Rn, <oprnd2> PL Positive or zero }  Branch(Jump) VS Overflow #32bit_Imm A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits  Branch B{cond} label VC No overflow <a_mode2> Refer to Table Addressing Mode 2 with link BL{cond} label Unsigned higher <a_mode2p> Refer to Table Addressing Mode 2 (Privileged)  Exch instr set BX{cond} Rn LS Unsigned lower or same <a_mode3> Refer to Table Addressing Mode 3  Coprocessors  GE Greater or equal <a_mode4l s=""> Refer to Table Addressing Mode 4 (Load/Store)  Data operations CDP{cond} p<cpnum>, <op1>, CRd, CRn, CRm, <op2> LT Less than <a_mode5> Refer to Table Addressing Mode 5  Sets condition codes (optional)  Movto coproc RC{cond} p<cpnum>, <op1>, Rd, CRn, CRm, <op2> GT Greater than Sets than or equal By the operation (optional)  Less than or equal By the operation (optional)</op2></op1></cpnum></a_mode5></op2></op1></cpnum></a_mode4l></a_mode3></a_mode2p></a_mode2></oprnd2>	EOR	EOR{cond}{S} Rd, Rn, <oprnd2></oprnd2>	CC	Unsigned lower	{field}	Refer to Table Field	
Branch (Jump)  Branch B{cond} label  VC No overflow  *32bit_Imm A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits  **Refer to Table Addressing Mode 2  **Refer to Table Addressing Mode 2  **Refer to Table Addressing Mode 2 (Privileged)  **Exch instr set BX{cond} Rn  **Coprocessors  **GE**  Greater or equal  **Coprocessors  **Data operations  **Cop**  **Cop**  **Cop**  **Cop**  **Cop**  **Cop**  **Cop**  **Refer to Table Addressing Mode 3  **Cop**  **C	ORR	ORR{cond}{S} Rd, Rn, <oprnd2></oprnd2>	MI	Negative	<reglist></reglist>	})	
Branch B{cond} label VC No overflow even number of bits  Branch B{cond} label VC No overflow <a_mode2> Refer to Table Addressing Mode 2  with link BL{cond} label HI Unsigned higher <a_mode2p> Refer to Table Addressing Mode 2 (Privileged)  Exch instr set BX{cond} Rn LS Unsigned lower or same   Coprocessors  GE Greater or equal   Ca_mode4L/S&gt; Refer to Table Addressing Mode 3   Coprocessors  CDP{cond} p<cpnum>, <op1>, CRd, CRn, CRm, <op2> LT Less than   Amount or go coproc MRC{cond}p<cpnum>, <op1>, Rd, CRn, CRm, <op2> GT Greater than   Movto coproc reg MCR{cond} p<cpnum>, <op1>, Rd, CRn, CRm, <op2> LE Less than or equal   Byte operation (optional)</op2></op1></cpnum></op2></op1></cpnum></op2></op1></cpnum></a_mode2p></a_mode2>	Bit Clear	BIC{cond}{S} Rd, Rn, <oprnd2></oprnd2>	PL	Positive or zero	#22k# T		
with link BL{cond} label HI Unsigned higher <a_mode2p> Refer to Table Addressing Mode 2 (Privileged)  Exch instr set BX{cond} Rn LS Unsigned lower or same <a_mode3p> Refer to Table Addressing Mode 3 (Privileged)  Coprocessors  GE Greater or equal <a_mode4l s=""> Refer to Table Addressing Mode 4 (Load/Store)  Lase than CDP{cond} p<cpnum>, <op1>, CRd, CRn, CRm, <op2> LT Less than <a_mode5p> Refer to Table Addressing Mode 5 (Privileged)  Movto reg coproc MRC{cond}p<cpnum>, <op1>, Rd, CRn, CRm, <op2> GT Greater than Sets than or equal Byte operation (optional)  Movto coproc reg MCR{cond} p<cpnum>, <op1>, Rd, CRn, CRm, <op2> LE Less than or equal Byte operation (optional)  All the sets than or equal Byte operation (optional)</op2></op1></cpnum></op2></op1></cpnum></a_mode5p></op2></op1></cpnum></a_mode4l></a_mode3p></a_mode2p>	Branch(Jump)		VS	Overflow	#32bit_1mm		
Exch instr set BX{cond} Rn  LS Unsigned lower or same <a_mode3> Refer to Table Addressing Mode 3  Coprocessors  GE Greater or equal  Ca_mode4L/S&gt; Refer to Table Addressing Mode 4 (Load/Store)  <a_mode4l s=""> Refer to Table Addressing Mode 4 (Load/Store)  Ca_mode4L/S&gt; Refer to Table Addressing Mode 4 (Load/Store)  Ca_mode5&gt; Refer to Table Addressing Mode 5  Coprocessors  Movto reg coproc MRC{cond}p<cpnum>, <op1>, Rd, CRn, CRm, <op2> GT Greater than  Movto coproc reg MCR{cond} p <cpnum>, <op1>, Rd, CRn, CRm, <op2> LE Less than or equal  Always  Al</op2></op1></cpnum></op2></op1></cpnum></a_mode4l></a_mode3>	Branch	B{cond} label	VC	No overflow	<a_mode2></a_mode2>	Refer to Table Addressing Mode 2	
GE Greater or equal <a mode4l="" s=""> Refer to Table Addressing Mode 4 (Load/Store)  Coprocessors  CDP(cond) p &lt; cpnum&gt;, &lt; op1&gt;, CRd, CRn, CRm, &lt; op2&gt; LT Less than  Movto reg coproc MRC(cond)p &lt; cpnum&gt;, &lt; op1&gt;, Rd, CRn, CRm, &lt; op2&gt; GT Greater than  Movto coproc reg MCR(cond) p &lt; cpnum&gt;, &lt; op1&gt;, Rd, CRn, CRm, &lt; op2&gt; LE Less than or equal  Always  Alway</a>	with link	BL{cond} label	HI	Unsigned higher	<a_mode2p></a_mode2p>	Refer to Table Addressing Mode 2 (Privileged)	
Data operations  CDP{cond} p < cpnum>, < op1>, CRd, CRn, CRm, < op2>  LT  Less than <a_mode5> Refer to Table Addressing Mode 5  Sets condition codes (optional)  Movto coproc reg  MCR{cond} p &lt; cpnum&gt;, &lt; op1&gt;, Rd, CRn, CRm, &lt; op2&gt;  LE  Less than  Sets condition codes (optional)  Byte operation (optional)  Byte operation (optional)</a_mode5>	Exch instr set	BX{cond} Rn	LS	Unsigned lower or same	<a_mode3></a_mode3>	Refer to Table Addressing Mode 3	
Movto reg coproc MRC{cond}p <cpnum>,<op1>,Rd,CRn,CRm,<op2> GT Greater than Sets condition codes (optional)  Movto coproc reg MCR{cond} p<cpnum>, <op1>, Rd, CRn, CRm, <op2> LE Less than or equal Byte operation (optional)  Movto coproc reg MCR{cond} p<cpnum>, CRd, ca mode Sets condition codes (optional)  Movto coproc reg MCR{cond} p<cpnum>, cop1&gt;, Rd, CRn, CRm, <op2> LE Less than or equal Byte operation (optional)</op2></cpnum></cpnum></op2></op1></cpnum></op2></op1></cpnum>	Coprocessors		GE	Greater or equal	<a_mode4l s=""></a_mode4l>	Refer to Table Addressing Mode 4 (Load/Store)	
Movto coproc reg MCR{cond} p <cpnum>, <op1>, Rd, CRn, CRm, <op2> LE Less than or equal Byte operation (optional)  Byte operation (optional)</op2></op1></cpnum>	Data operations	CDP{cond} p <cpnum>, <op1>, CRd, CRn, CRm, <op2></op2></op1></cpnum>	LT	Less than	<a_mode5></a_mode5>	Refer to Table Addressing Mode 5	
byte operation (optional)	Movto reg coproc	MRC{cond}p <cpnum>,<op1>,Rd,CRn,CRm,<op2></op2></op1></cpnum>	GT	Greater than	S	Sets condition codes (optional)	
.oad LDC(cond) p <cpnum>, CRd, <a_mode5> AL Always</a_mode5></cpnum>	Movto coproc reg	MCR{cond} p <cpnum>, <op1>, Rd, CRn, CRm, <op2></op2></op1></cpnum>	LE	Less than or equal	В	Byte operation (optional)	
11 Hallword Operation (Optional)	Load	LDC{cond} p <cpnum>, CRd, <a_mode5></a_mode5></cpnum>	AL	Always	Н	Halfword operation (optional)	
Store STC{cond} p <cpnum>, CRd, <a_mode5> T Forces address translation.</a_mode5></cpnum>	Store	STC{cond} p <cpnum>, CRd, <a_mode5></a_mode5></cpnum>			Т	Forces address translation.	