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 3 - Signed Byte and Halfword Data Transfer	
Immediate offset	[Rn, #+/-8bit_Offset]
Pre-indexed	[Rn, #+/-8bit_Offset]!
Pre-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 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 4	Stack Type Load	Stack Type Store
IA Increment After	FD Full Descending	EA Empty Ascending
IB Increment Before	ED Empty Descending	FA Full Ascending
DA Decrement After	FA Full Ascending	ED Empty Descending
DB Decrement Before	EA Empty Ascending	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

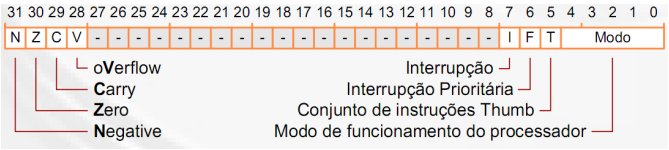
Field	
Suffix	**Sets**
_c	Control field mask bit (bit 3)
_f	Flags field mask bit (bit 0)
_s	Status field mask bit (bit 1)
_x	Extension field mask bit (bit 2)

Quick Reference

Sistemas Embebidos I
ISEL - DEETC
Inverno 2010/2011
V1.1
Nuno Cancelo
LEIC
31401

Source: http://infocenter.arm.com/help/topic/com.arm.doc.ddi0234b/BGEJCAFI.html

Current Program Status Register (CPSR)



Move	
Move	MOV{cond}{S} Rd, <Oprnd2>
Move NOT	MVN{cond}{S} Rd, <Oprnd2>
Arithmetic	
Add	ADD{cond}{S} Rd, Rn, <Oprnd2>
Add with carry	ADC{cond}{S} Rd, Rn, <Oprnd2>
Subtract	SUB{cond}{S} Rd, Rn, <Oprnd2>
Subtract with carry	SBC{cond}{S} Rd, Rn, <Oprnd2>
Sub reverse sub	RSB{cond}{S} Rd, Rn, <Oprnd2>
Sub rev. Sub. w carry	RSC{cond}{S} Rd, Rn, <Oprnd2>
Multiply	MUL{cond}{S} Rd, Rm, Rs
Multiply Accumulate	MLA{cond}{S} Rd, Rm, Rs, Rn
Multiply Unsigned Long	UMULL{cond}{S} RdLo, RdHi, Rm, Rs
Multiply Unsig. Accum Long	UMLAL{cond}{S} RdLo, RdHi, Rm, Rs
Multiply Signed Long	SMULL{cond}{S} RdLo, RdHi, Rm, Rs
Multiply signed accum long	SMLAL{cond}{S} RdLo, RdHi, Rm, Rs
Compare	CMP{cond} Rd, <Oprnd2>
Compare negative	CMN{cond} Rd, <Oprnd2>
Logical	
Test	TST{cond} Rn, <Oprnd2>
Test equivalence	TEQ{cond} Rn, <Oprnd2>
AND	AND{cond}{S} Rd, Rn, <Oprnd2>
EOR	EOR{cond}{S} Rd, Rn, <Oprnd2>
ORR	ORR{cond}{S} Rd, Rn, <Oprnd2>
Bit Clear	BIC{cond}{S} Rd, Rn, <Oprnd2>
Branch(Jump)	
Branch	B{cond} label
with link	BL{cond} label
Exch instr set	BX{cond} Rn
Coprocessors	
Data operations	CDP{cond} p<cpnum>,<op1>,CRd, CRn, CRm,<op2>
Movto reg coproc	MRC{cond}p<cpnum>,<op1>,Rd,CRn,CRm,<op2>
Movto coproc reg	MCR{cond} p<cpnum>,<op1>,Rd, CRn, CRm, <op2>
Load	LDC{cond} p<cpnum>, CRd, <a_mode5>
Store	STC{cond} p<cpnum>, CRd, <a_mode5>
Key to Table	
{!}	updates base register after the transfer if ! Present (Post -indexed alwaays updates)
<immed_8r>	A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits
+/-, ^	+ or - (+ may be ommited), ^ sets the S bit

Load	
Word	LDR{cond} Rd, <a_mode2>
Word user privilege	LDR{cond}T Rd, <a_mode2P>
Byte	LDR{cond}B Rd, <a_mode2>
Byte user privilege	LDR{cond}BT Rd, <a_mode2P>
Byte signed	LDR{cond}SB Rd, <a_mode3>
Halfword	LDR{cond}H Rd, <a_mode3>
Halfword signed	LDR{cond}SH Rd, <a_mode3>
Multiple	
Block data operations	
Increment Before	LDM{cond}IB Rd{!}, <reglist>{^}
Increment After	LDM{cond}IA Rd{!}, <reglist>{^}
Decrement Before	LDM{cond}DB Rd{!}, <reglist>{^}
Decrement After	LDM{cond}DA Rd{!}, <reglist>{^}
Stack operations	LDM{cond}<a_mode4L> Rd{!}, <reglist>
Stack restoreCPSR	LDM{cond}<a_mode4L> Rd{!}, <reglist+pc>^
User registers	LDM{cond}<a_mode4L> Rd, <reglist>^

Condition Field {cond}

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

Move	
Move	MOV{cond}{S} Rd, <Oprnd2>
Move NOT	MVN{cond}{S} Rd, <Oprnd2>
Move SPSR to Register	MRS{cond} Rd, SPSR
Move CPSR to Register	MRS{cond} Rd, CPSR
Move Register to SPSR	MSR{cond} SPSR{field}, Rm
Move Register to SPSR	MSR{cond} CPSR{field}, Rm
Move immediate to SPSR Flags	MSR{cond} SPSR_f, #32bit_Imm
Move immediate to CPST Flags	MSR{cond} CPSR_f, #32bit_Imm

Store	
Word	STR{cond} Rd, <a_mode2>
Word user privilege	STR{cond}T Rd, <a_mode2P>
Byte	STR{cond}B Rd, <a_mode2>
Byte user privilege	STR{cond}BT Rd, <a_mode2P>
Halfword	STR{cond}H Rd, <a_mode3>
Multiple	
Block data operations	
Increment Before	STM{cond}IB Rd{!}, <reglist>{^}
Increment After	STM{cond}IA Rd{!}, <reglist>{^}
Decrement Before	STM{cond}DB Rd{!}, <reglist>{^}
Decrement After	STM{cond}DA Rd{!}, <reglist>{^}
Stack operations	STM{cond}<a_mode4S> Rd{!}, <reglist>
User registers	STM{cond}<a_mode4S> Rd{!}, <reglist>^
SWAP	
Word	SWP{cond} Rd, Rm, [Rn]
Byte	SWP{cond}B Rd, Rm, [Rn]
Software Interrupt	
Software Interrupt	SWI 24bit_Imm

Key to Tables	
{cond}	Refer to Table Condition Field {cond}
<Oprnd2>	Refer to Table Oprnd2
{field}	Refer to Table Field
<reglist>	A comma-separated list of registers, enclosed in braces ({ and })
#32bit_Imm	A 32-bit constant, formed by right-rotating an 8-bit value by an even number of bits
<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/S>	Refer to Table Addressing Mode 4 (Load/Store)
<a_mode5>	Refer to Table Addressing Mode 5
S	Sets condition codes (optional)
B	Byte operation (optional)
H	Halfword operation (optional)
T	Forces address translation.