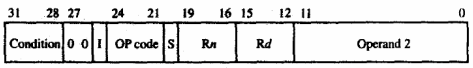
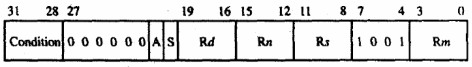
ARM

Vladimir Filipović
vladaf@matf.bg.ac.yu

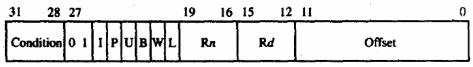
ARM



(a) Arithmetic, logic, compare, test, and move

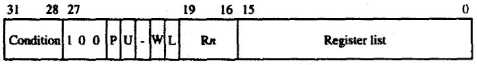


(b) Multiply and Multiply Accumulate

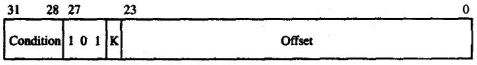


(c) Single word or byte transfer from/to memory

ARM



(d) Multiple word transfer from/to memory



(e) Branch and Branch with Link

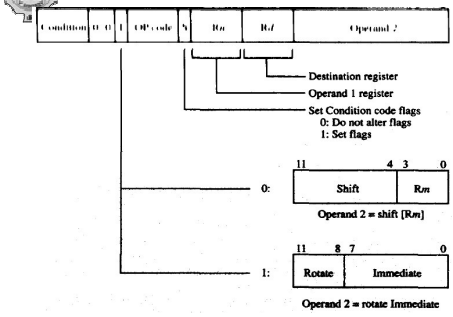
I Immediate
S Set
A Accumulate
P Pre/Post-index
U Up/Down
B Byte/Word
W Writeback
L Load/Store
K Link

Figure B.1 ARM instruction encoding formats.

ARM

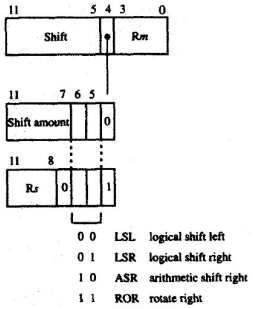
Condition code	Condition suffix	Condition name	Condition rule
0 0 0 0	EQ	Equal (zero)	$Z = 1$
0 0 0 1	NE	Not equal (nonzero)	$Z = 0$
0 0 1 0	CS/HS	Carry set/Unsigned higher or same	$C = 1$
0 0 1 1	CC/LO	Carry clear/Unsigned lower	$C = 0$
0 1 0 0	MI	Minus (negative)	$N = 1$
0 1 0 1	PL	Plus (positive or zero)	$N = 0$
0 1 1 0	VS	Overflow	$V = 1$
0 1 1 1	VC	No overflow	$V = 0$
1 0 0 0	HI	Unsigned higher	$\overline{C} \vee Z = 0$
1 0 0 1	LS	Unsigned lower or same	$\overline{C} \vee Z = 1$
1 0 1 0	GE	Signed greater than or equal	$N \oplus V = 0$
1 0 1 1	LT	Signed less than	$N \oplus V = 1$
1 1 0 0	GT	Signed greater than	$Z \vee (N \oplus V) = 0$
1 1 0 1	LE	Signed less than or equal	$Z \vee (N \oplus V) = 1$
1 1 1 0	AL	Always	
1 1 1 1		Not used	

ARM



Format ARM aritmetičkih i logičkih instrukcija, te instrukcija poređenja, testiranja i premeštanja

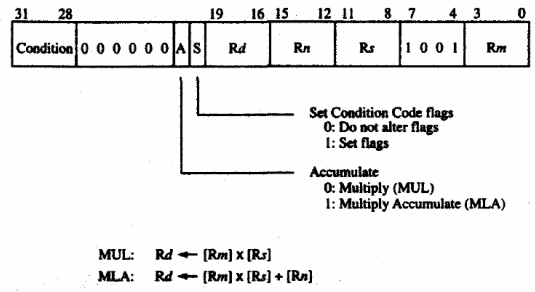
ARM

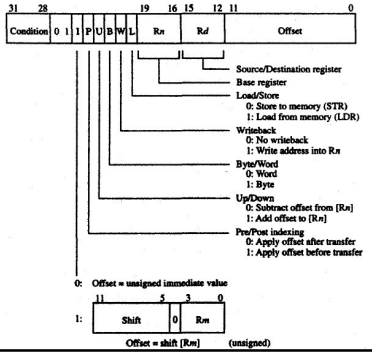


Format ARM aritmetičkih i logičkih pomeranja operanda koji se nalazi u registru R_m

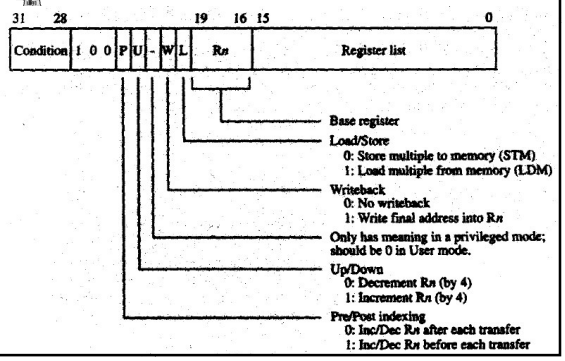
Mnemonic (Name)	OP code $b_{24}\ldots b_{21}$	Operation performed	CC flags affected if S = 1 N	Z	V	C
ADD (Add)	0 1 0 0	Rd ← [Rn] + Oper2	x	x	x	x
ADC (Add with carry)	0 1 0 1	Rd ← [Rn] + Oper2 + [C]	x	x	x	x
SUB (Subtract)	0 0 1 0	Rd ← [Rn] − Oper2	x	x	x	x
SBC (Subtract with carry)	0 1 1 0	Rd ← [Rn] − Oper2 + [C] − 1	x	x	x	x
RSB (Reverse subtract)	0 0 1 1	Rd ← Oper2 − [Rn]	x	x	x	x
RSC (Reverse subtract with carry)	0 1 1 1	Rd ← Oper2 − [Rn] + [C] − 1	x	x	x	x
MUL (Multiply)	(See Figure B.4)	Rd ← [Rm] × [Rs]	x	x		
MLA (Multiply accumulate)	(See Figure B.4)	Rd ← [Rm] × [Rs] + [Rn]	x	x		

Mnemonic (Name)	OP code $b_{24}\ldots b_{21}$	Operation performed	CC flags affected if S = 1 N	Z	V	C
AND (Logical AND)	0 0 0 0	Rd ← [Rn] ∧ Oper2	x	x		x
ORR (Logical OR)	1 1 0 0	Rd ← [Rn] ∨ Oper2	x	x		x
EOR (Exclusive-OR)	0 0 0 1	Rd ← [Rn] ⊕ Oper2	x	x		x
BIC (Bit clear)	1 1 1 0	Rd ← [Rn] ∧ ¬Oper2	x	x		x
CMP (Compare)	1 0 1 0	[Rn] − Oper2	x	x	x	x
CMN (Compare negative)	1 0 1 1	[Rn] + Oper2	x	x	x	x
TST (Bit test)	1 0 0 0	[Rn] ∧ Oper2	x	x		x
TEQ (Test equal)	1 0 0 1	[Rn] ⊕ Oper2	x	x		x
MOV (Move)	1 1 0 1	Rd ← Oper2	x	x		x
MVN (Move complement)	1 1 1 1	Rd ← ¬Oper2	x	x		x

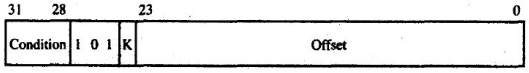




Mnemonic (Name)	Instruction bits B L	Operation performed
LDR (Load word)	0 1	Rd ← [EA]
LDRB (Load byte)	1 1	Rd ← [EA]
STR (Store word)	0 0	EA ← [Rd]
STRB (Store byte)	1 0	EA ← [Rd]



Mnemonic (Name)	Instruction bits P U L	Operation performed
LDMIA/LDMFD (Increment after/ Full descending)	0 1 1	$R_{low}, \ldots, R_{high} \leftarrow [[Rn]], [[Rn]+4], \ldots$
LDMIB/LDMED (Increment before/ Empty descending)	1 1 1	$R_{low}, \ldots, R_{high} \leftarrow [[Rn]+4], [[Rn]+8], \ldots$
LDMDA/LDMFA (Decrement after/ Full ascending)	0 0 1	$R_{high}, \ldots, R_{low} \leftarrow [[Rn]], [[Rn]-4], \ldots$
LDMDB/LDMEA (Decrement before/ Empty ascending)	1 0 1	$R_{high}, \ldots, R_{low} \leftarrow [[Rn]-4], [[Rn]-8], \ldots$
STMIA/STMEA (Increment after/ Empty ascending)	0 1 0	$[Rn], [Rn]+4, \ldots \leftarrow [R_{low}], \ldots, [R_{high}]$
STMIB/STMFA (Increment before/ Full ascending)	1 1 0	$[Rn]+4, [Rn]+8, \ldots \leftarrow [R_{low}], \ldots, [R_{high}]$
STMDA/STMED (Decrement after/ Empty descending)	0 0 0	$[Rn], [Rn]-4, \ldots \leftarrow [R_{high}], \ldots, [R_{low}]$
STMDB/STMFD (Decrement before/ Full descending)	1 0 0	$[Rn]-4, [Rn]-8, \ldots \leftarrow [R_{high}], \ldots, [R_{low}]$



K=0 :Branch (B)
K=1 :Branch with Link (BL); store return address in register R14

Mnemonic (Name)	Instruction formats	Operation performed
MRS (Copy status register)	User mode:	
	MRS Rd,CPSR	$Rd \leftarrow [CPSR]$
	Privileged mode:	
	MRS Rd,CPSR	$Rd \leftarrow [CPSR]$
	MRS Rd,SPSR	$Rd \leftarrow [SPSR\_mode]$
MSR (Write to status register)	User mode:	
	MSR CPSR,Rm	$CPSR_{31-28} \leftarrow [Rm]_{31-28}$
	MSR CPSR,imm32	$CPSR_{31-28} \leftarrow imm32_{31-28}$
	Privileged mode:	
	MSR CPSR,Rm	$CPSR \leftarrow [Rm]$
	MSR CPSR_flg,Rm	$CPSR_{31-28} \leftarrow [Rm]_{31-28}$
	MSR CPSR_flg,imm32	$CPSR_{31-28} \leftarrow imm32_{31-28}$
	MSR SPSR,Rm	$SPSR\_mode \leftarrow [Rm]$
	MSR SPSR_flg,Rm	$SPSR\_mode_{31-28} \leftarrow [Rm]_{31-28}$
	MSR SPSR_flg,imm32	$SPSR\_mode_{31-28} \leftarrow imm32_{31-28}$
SWI (Software interrupt)	SWI imm24	R14_svc ← updated[PC]; SPSR_svc ← [CPSR]; PC ← 0x08
SWP (Swap)	SWP Rd,Rm,[Rn]	$Rd \leftarrow [[Rn]]$; $[Rn] \leftarrow [Rm]$