ARM Assembly to Machine Language Translation Guide

Data Processing Instructions [op = 00]

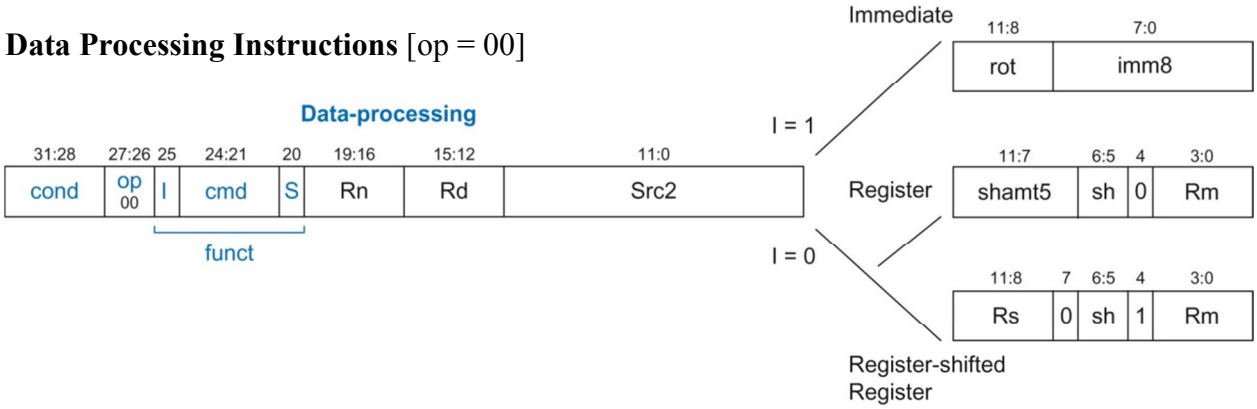


Table B.1 Data-processing instructions

cmd	Name	Description	Operation
0000	AND Rd, Rn, Src2	Bitwise AND	Rd ← Rn & Src2
0001	EOR Rd, Rn, Src2	Bitwise XOR	Rd ← Rn ^ Src2
0010	SUB Rd, Rn, Src2	Subtract	Rd ← Rn - Src2
0011	RSB Rd, Rn, Src2	Reverse Subtract	Rd ← Src2 - Rn
0100	ADD Rd, Rn, Src2	Add	Rd ← Rn+Src2
0101	ADC Rd, Rn, Src2	Add with Carry	Rd ← Rn+Src2+C
0110	SBC Rd, Rn, Src2	Subtract with Carry	Rd ← Rn - Src2 - $\overline{C}$
0111	RSC Rd, Rn, Src2	Reverse Sub w/ Carry	Rd ← Src2 - Rn - $\overline{C}$
1000 ($S = 1$)	TST Rd, Rn, Src2	Test	Set flags based on Rn & Src2
1001 ($S = 1$)	TEQ Rd, Rn, Src2	Test Equivalence	Set flags based on Rn ^ Src2
1010 ($S = 1$)	CMP Rn, Src2	Compare	Set flags based on Rn - Src2
1011 ($S = 1$)	CMN Rn, Src2	Compare Negative	Set flags based on Rn+Src2
1100	ORR Rd, Rn, Src2	Bitwise OR	Rd ← Rn | Src2
1101	Shifts:		
$I = 1$ OR ($\text{instr}_{11:4} = 0$)	MOV Rd, Src2	Move	Rd ← Src2
$I = 0$ AND ($sh = 00$; $\text{instr}_{11:4} \neq 0$)	LSL Rd, Rm, Rs/shamt5	Logical Shift Left	Rd ← Rm << Src2
$I = 0$ AND ($sh = 01$)	LSR Rd, Rm, Rs/shamt5	Logical Shift Right	Rd ← Rm >> Src2
$I = 0$ AND ($sh = 10$)	ASR Rd, Rm, Rs/shamt5	Arithmetic Shift Right	Rd ← Rm>>>Src2
$I = 0$ AND ($sh = 11$; $\text{instr}_{11:7,\,4} = 0$)	RRX Rd, Rm, Rs/shamt5	Rotate Right Extend	{Rd, C} ← {C, Rd}
$I = 0$ AND ($sh = 11$; $\text{instr}_{11:7} \neq 0$)	ROR Rd, Rm, Rs/shamt5	Rotate Right	Rd ← Rm ror Src2
1110	BIC Rd, Rn, Src2	Bitwise Clear	Rd ← Rn & ~Src2
1111	MVN Rd, Rm, Src2	Bitwise NOT	Rd ← ~Rm

NOP (no operation) is typically encoded as 0xE1A000, which is equivalent to MOV R0, R0.

Rm

~Src2

Table 6.3 Condition mnemonics

cond	Mnemonic	Name	CondEx
0000	EQ	Equal	Z
0001	NE	Not equal	$\overline{Z}$
0010	CS/HS	Carry set / unsigned higher or same	C
0011	CC/LO	Carry clear / unsigned lower	$\overline{C}$
0100	MI	Minus / negative	N
0101	PL	Plus / positive or zero	$\overline{N}$
0110	VS	Overflow / overflow set	V
0111	VC	No overflow / overflow clear	$\overline{V}$
1000	HI	Unsigned higher	$\overline{Z}C$
1001	LS	Unsigned lower or same	Z OR $\overline{C}$
1010	GE	Signed greater than or equal	$\overline{N \oplus V}$
1011	LT	Signed less than	$N \oplus V$
1100	GT	Signed greater than	$\overline{Z(N \oplus V)}$
1101	LE	Signed less than or equal	Z OR $(N \oplus V)$
1110	AL (or none)	Always / unconditional	Ignored

Table 6.8 *sh* field encodings

Instruction	sh	Operation
LSL	00_2	Logical shift left
LSR	01_2	Logical shift right
ASR	10_2	Arithmetic shift right
ROR	11_2	Rotate right

Table 6.9 Offset type control bits for memory instructions

Bit	Meaning: I	Meaning: U
0	Immediate offset in Src2	Subtract offset from base
1	Register offset in Src2	Add offset to base

Table 6.10 Index mode control bits for memory instructions

P	W	Index Mode
0	0	Post-index
0	1	Not supported
1	0	Offset
1	1	Pre-index

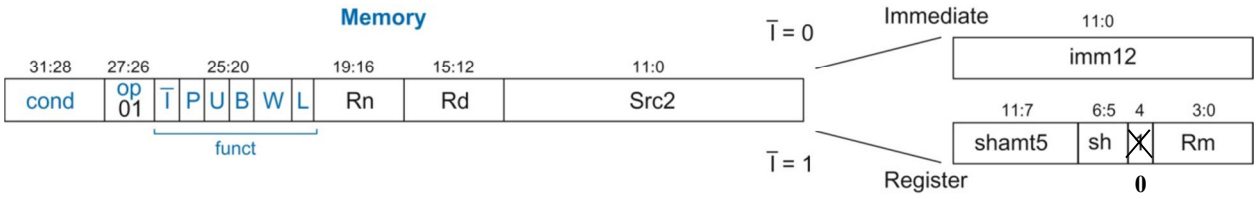
Table 6.11 Memory operation type control bits for memory instructions

L	B	Instruction
0	0	STR
0	1	STRB
1	0	LDR
1	1	LDRB

Table 6.4 ARM indexing modes

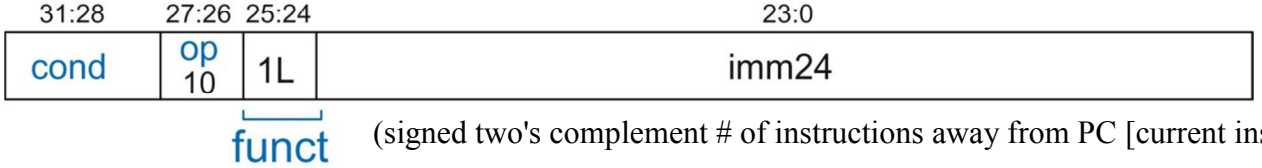
Mode	ARM Assembly	Address	Base Register
Offset	LDR R0, [R1, R2]	R1 + R2	Unchanged
Pre-index	LDR R0, [R1, R2]!	R1 + R2	R1 = R1 + R2
Post-index	LDR R0, [R1], R2	R1	R1 = R1 + R2

Memory Instructions [op = 01] ldr, str, ldrb, strb



r11	fp
r12	ip
r13	sp
r14	lr
r15	pc

Branch Instructions [op = 10] **Branch**



(signed two's complement # of instructions away from PC [current instruction + 2])

Table B.4 Branch instructions

L	Name	Description	Operation
0	B label	Branch	PC ← (PC+8)+imm24 << 2
1	BL label	Branch with Link	LR ← (PC+8) - 4; PC ← (PC+8)+imm24 << 2

When reading from PC (r15), PC = Current instruction executing + 8 bytes (+2 instructions away)