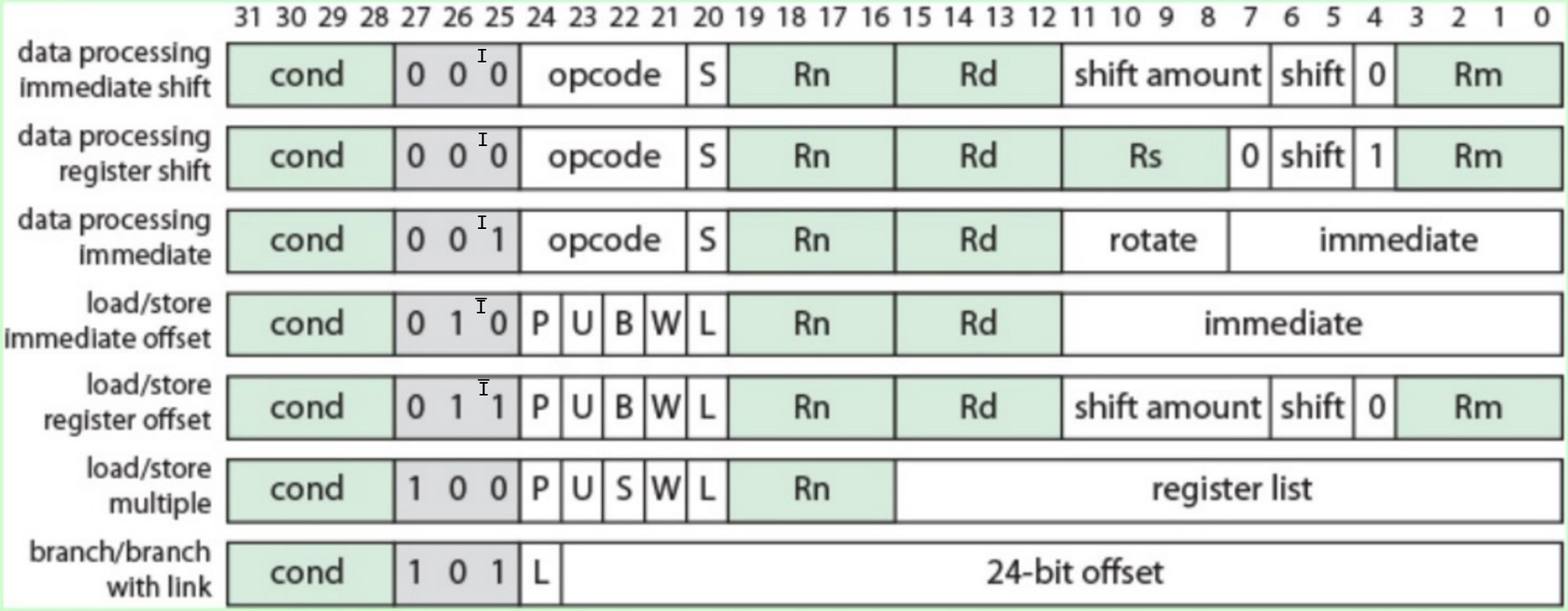
ARM Instruction Formats



- S = For data processing instructions, updates condition codes
- S = For load/store multiple instructions, execution restricted to supervisor mode
- P, U, W = distinguish between different types of addressing_mode
- B = Unsigned byte (B==1) or word (B==0) access
- L = For load/store instructions, Load (L==1) or Store (L==0)
- L = For branch instructions, is return address stored in link register

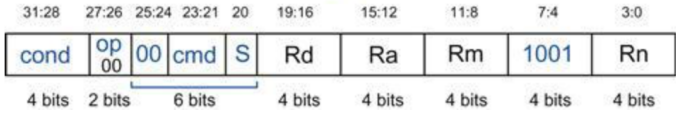
http://www.slideshare.net/an13ika/11-instruction-sets-addressing-modes

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 ← Rn ror Src2
1110	BIC Rd, Rn, Src2	Bitwise Clear	Rd ← Rn & ~Src2
1111	MVN Rd, Rn, Src2	Bitwise NOT	Rd ← ~Rn

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	$\overline{Z}(\overline{N \oplus V})$
1101	LE	Signed less than or equal	Z OR $(N \oplus V)$
1110	AL (or none)	Always / unconditional	Ignored

Multiply



Multiply instructions

cmd	Name	Description	Operation
000	MUL Rd, Rn, Rm	Multiply	Rd ← Rn × Rm (low 32 bits)
001	MLA Rd, Rn, Rm, Ra	Multiply Accumulate	Rd ← (Rn × Rm)+Ra (low 32 bits)
100	UMULL Rd, Rn, Rm, Ra	Unsigned Multiply Long	{Rd, Ra} ← Rn × Rm (all 64 bits, Rm/Rn unsigned)
101	UMLAL Rd, Rn, Rm, Ra	Unsigned Multiply Accumulate Long	{Rd, Ra} ← (Rn × Rm)+{Rd, Ra} (all 64 bits, Rm/Rn unsigned)
110	SMULL Rd, Rn, Rm, Ra	Signed Multiply Long	{Rd, Ra} ← Rn × Rm (all 64 bits, Rm/Rn signed)
111	SMLAL Rd, Rn, Rm, Ra	Signed Multiply Accumulate Long	{Rd, Ra} ← (Rn × Rm)+{Rd, Ra} (all 64 bits, Rm/Rn signed)

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

Meaning		
Bit $\overline{I}$		U
0	Immediate offset in Src2	Subtract offset from base
1	Register offset in Src2	Add offset to base

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

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

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

Immediate rotations and resulting 32-bit constant for *imm8* = 0xFF

rot	32-bit Constant
0000	0000 0000 0000 0000 0000 0000 1111 1111
0001	1100 0000 0000 0000 0000 0000 0011 1111
0010	1111 0000 0000 0000 0000 0000 0000 1111
…	…
1111	0000 0000 0000 0000 0000 0011 1111 1100