






Assembly instructions - ARM V7

Category	Instruction	Mnemonic	Meaning	Flags			
				N	Z	C	V
Arithmetic	Add	ADD{S}{cond} dest, op1, op2	dest = op1 + op2	±	±	±	±
	Subtract	SUB{S}{cond} dest, op1, op2	dest = op1 - op2	±	±	±	±
	Add with Carry	ADC{S}{cond} dest, op1, op2	dest = op1 + op2 + carry	±	±	±	±
	Subtract with Carry	SBC{S}{cond} dest, op1, op2	dest = op1 - op2 - 1 + carry	±	±	±	±
	Reverse Subtract	RSB{S}{cond} dest, op1, op2	dest = op2 - op1	±	±	±	±
	Reverse Subtract with Carry	RSC{S}{cond} dest, op1, op2	dest = op2 - op1 - 1 + carry	±	±	±	±
Logic	Bitwise And	AND{S}{cond} dest, op1, op2	dest = AND(op1, op2)	±	±	x	x
	Bitwise Exclusive Or	EOR{S}{cond} dest, op1, op2	dest = XOR(op1, op2)	±	±	x	x
	Bitwise Clear	BIC{S}{cond} dest, op1, op2	dest = AND(op1, NOT(op2))	±	±	x	x
	Bitwise Or	ORR{S}{cond} dest, op1, op2	dest = OR(op1, op2)	±	±	x	x
	Logical Shift Left	LSL{S}{cond} dest, op1, op2		±	±	±	x
	Logical Shift Right	LSR{S}{cond} dest, op1, op2		±	±	±	x
	Arithmetic Shift Right	ASR{S}{cond} dest, op1, op2		±	±	±	x
	Rotate Right	ROR{S}{cond} dest, op1, op2		±	±	±	x
	Rotate Right and Extend	RRX{S}{cond} dest, op1		±	±	±	x
Data transfer	Move	MOV{S}{cond} dest, op1	dest = op1	±	±	x	x
	Move Negated	MVN{S}{cond} dest, op1	dest = NOT(op1)	±	±	x	x
	Address Load	ADR{cond} dest, expression		x	x	x	x
	LDR Pseudo-Instruction	LDR{cond} dest, =expression		x	x	x	x
	Load Register	LDR{B}{cond} dest, [source {, OFFSET}]	dest = Mem[source + OFFSET]	x	x	x	x
	Store Register	STR{B}{cond} source, [dest {, OFFSET}]	Mem[dest + OFFSET] = source	x	x	x	x
Comparisons and jumps	Compare	CMP{cond} op1, op2	op1 - op2	±	±	±	±
	Compare Negated	CMN{cond} op1, op2	op1 + op2	±	±	±	±
	Test Bit(s) Set	TST{cond} op1, op2	AND(op1, op2)	±	±	x	x
	Test Equals	TEQ{cond} op1, op2	XOR(op1, op2)	±	±	x	x
	Branch	B{cond} target		x	x	x	x
	Branch with Link	BL{cond} target		x	x	x	x

± – Updated | x – Not affected

Directives	Declare Word(s) in Memory	name DCD value_1, value_2, ... value_N
	Declare Constant	name equ expression
	Declare Empty Word(s) in Memory	{name} FILL N, N must be a multiple of 4
	Stop Emulation	END{cond}

Notes:

- For all instructions that require **dest**, **op1**, & **op2**, **dest** and **op1** must be registers.
- expression** is a numerical constant or expression that evaluates to a 32-bit number. The operators +, - and * are allowed. A constant is a decimal number as a series of digits 0-9, a hexadecimal number prefixed with 0x or & or a binary number prefixed with 0b.
- For MOV / MVN, **dest** must be a register.
- For CMP / CMN / TST / TEQ, **op1** must be a register.
- {...} indicates optional code.
- {cond} refers to the **condition code**. (See the table "Condition code suffixes")
- {S} is the **set bit**. If this is present, the status bits (**Flags**) will be set.
- {, OFFSET} refers to the offset applied to the source address or destination address for load and store instructions respectively. It can be a register, a numerical expression, or a shifted register (like the flexible second operand discussed earlier).
- {B} refers to **byte mode**. By default, the LDR / STR instructions load a word (32 bits) from the memory at the given address. If B is used, the byte at the given address is loaded instead.
- The **target** for a branch instruction B must be a label on a line. This instruction will cause the program to "jump", i.e. branch, to this line of code.
- Branch with link BL is identical to branch, with the additional function that the link register is set to point to the next line of code before the branch is performed. This can be used to return from a subroutine.
- For EQU, **expression** can be any numerical expression

Condition code suffixes					
Suffix	Flags	Meaning	Suffix	Flags	Meaning
EQ	Z = 1	Equal	VC	V = 0	No overflow
NE	Z = 0	Not equal	HI	C = 1 and Z = 0	Higher, unsigned
CS or HS	C = 1	Higher or same, unsigned	LS	C = 0 or Z = 1	Lower or same, unsigned
CC or LO	C = 0	Lower, unsigned	GE	N = V	Greater than or equal, signed
MI	N = 1	Negative	LT	N != V	Less than, signed
PL	N = 0	Positive or zero	GT	Z = 0 and N = V	Greater than, signed
VS	V = 1	Overflow	LE	Z = 1 and N != V	Less than or equal, signed

The condition flags	
Name	Behavior
N	Set to 1 when the result of the operation was negative, cleared to 0 otherwise.
Z	Set to 1 when the result of the operation was zero, cleared to 0 otherwise.
C	Set to 1 when the operation resulted in a carry, cleared to 0 otherwise.
V	Set to 1 when the operation caused overflow, cleared to 0 otherwise.

Registers	
Name	Description
R0-R12	General-purpose registers. By convention, registers R0 to R3 are used to pass arguments to subroutines, and R0 is used to pass a result back to the callers. A subroutine that needs more than 4 inputs uses the stack for the additional inputs.
SP	<i>Stack Pointer</i>
LR	<i>Link Register</i>
PC	Program Counter