Conjunto de instrucciones ARM Thumb (1/2)

Operación		Ensamblador	Acción	Actualiza	Notas
Mover	Inmediato 8 bits	`mov Rd, #Inm8`	Rd ← Inm8	N Z	Rango Inm8: 0–255.
	Lo a Lo	`mov Rd, Rm`	Rd ← Rm	N Z	
	Hi a Lo, Lo a Hi, Hi a Hi	`mov Rd, Rm`	Rd ← Rm		
Sumar	Inmediato 3 bits	`add Rd, Rn, #Inm3`	Rd ← Rn + Inm3	N Z C V	Rango Inm3: 0–7.
	Inmediato 8 bits	`add Rd, Rd, #Inm8`	Rd ← Rd + Inm8	N Z C V	Rango Inm8: 0–255.
	Lo a Lo	`add Rd, Rn, Rm`	Rd ← Rn + Rm	N Z C V	
	Hi a Lo, Lo a Hi, Hi a Hi	`add Rd, Rd, Rm`	Rd ← Rd + Rm		
	Valor a SP	`add SP, SP, #Inm`	SP ← SP + Inm		Rango Inm: 0–508 (alineado a palabra).
	Crear dirección desde SP	`add Rd, SP, #Inm`	Rd ← SP + Inm		Rango Inm: 0–1020 (alineado a palabra).
Restar	Inmediato 3 bits	`sub Rd, Rn, #Inm3`	Rd ← Rn − Inm3	N Z C V	Rango Inm3: 0–7.
	Inmediato 8 bits	`sub Rd, Rd, #Inm8`	Rd ← Rd − Inm8	N Z C V	Rango Inm8: 0–255.
	Lo a Lo	`sub Rd, Rn, Rm`	Rd ← Rn − Rm	N Z C V	
	Valor de SP	`sub SP, SP, #Inm`	SP ← SP − Inm		Rango Inm: 0–508 (alineado a palabra).
	Negar	`neg Rd, Rn`	Rd ← −Rn	N Z C V	
Multiplicar	Multiplica	`mul Rd, Rm, Rd`	Rd ← Rm ∗ Rd	N Z	
Comparar	Compara	`cmp Rn, Rm`	Act. flags según Rn − Rm	N Z C V	Cualquier registro a cualquier registro.
	Compara con negado	`cmn Rn, Rm`	Act. flags según Rn + Rm	N Z C V	
	Inmediato	`cmp Rn, #Inm8`	Act. flags según Rn − Inm8	N Z C V	Rango Inm8: 0–255.
Lógicas	AND	`and Rd, Rd, Rm`	Rd ← Rd AND Rm	N Z	
	AND NOT (borrar bits)	`bic Rd, Rd, Rm`	Rd ← Rd AND NOT Rm	N Z	
	OR	`orr Rd, Rd, Rm`	Rd ← Rd OR Rm	N Z	
	XOR (or exclusiva)	`eor Rd, Rd, Rm`	Rd ← Rd XOR Rm	N Z	
	NOT	`mvn Rd, Rm`	Rd ← NOT Rm	N Z	
	Comprueba bits	`tst Rn, Rm`	Act. flags según Rn AND Rm	N Z	
Desplazar	Lógico a la izquierda	`lsl Rd, Rm, #Shift`	Rd ← Rm << Shift	N Z C	Rango Shift: 0–31
		`lsl Rd, Rd, Rs`	Rd ← Rd << $[Rs]_{7:0}$	N Z C	
	Lógico a la derecha	`lsr Rd, Rm, #Shift`	Rd ← Rm $>>_l$ Shift	N Z C	Rango Shift: 0–31
		`lsr Rd, Rd, Rs`	Rd ← Rd $>>_l$ $[Rs]_{7:0}$	N Z C	
	Aritmético a la derecha	`asr Rd, Rm, #Shift`	Rd ← Rm $>>_a$ Shift	N Z C	Rango Shift: 0–31
		`asr Rd, Rd, Rs`	Rd ← Rd $>>_a$ $[Rs]_{7:0}$	N Z C	
	Rotación a la derecha	`ror Rd, Rd, Rs`	Rd ← Rd ROR $[Rs]_{7:0}$	N Z C	

Conjunto de instrucciones ARM Thumb (2/2)

Operación		Ensamblador	Acción	Notas
Cargar	Con desp. inm., palabra	`ldr Rd, [Rn, #Inm]`	Rd ← [Rn + Inm]	Rango Inm: 0–124, múltiplos de 4.
	media palabra	`ldrh Rd, [Rn, #Inm]`	Rd ← ZeroExtend($[Rn + Inm]_{15:0}$)	Bits 31:16 a 0. Rango Inm: 0–62, pares.
	byte	`ldrb Rd, [Rn, #Inm]`	Rd ← ZeroExtend($[Rn + Inm]_{7:0}$)	Bits 31:8 a 0. Rango Inm: 0–31.
	Con desp. en registro, palabra	`ldr Rd, [Rn, Rm]`	Rd ← [Rn + Rm]	
	media palabra	`ldrh Rd, [Rn, Rm]`	Rd ← ZeroExtend($[Rn + Rm]_{15:0}$)	Bits 31:16 a 0.
	media palabra con signo	`ldrsh Rd, [Rn, Rm]`	Rd ← SignExtend($[Rn + Rm]_{15:0}$)	Bits 31:16 igual al bit 15.
	byte	`ldrb Rd, [Rn, Rm]`	Rd ← ZeroExtend($[Rn + Rm]_{7:0}$)	Bits 31:8 a 0.
	byte con signo	`ldrsb Rd, [Rn, Rm]`	Rd ← SignExtend($[Rn + Rm]_{7:0}$)	Bits 31:8 igual al bit 7.
	Relativo al PC	`ldr Rd, [PC, #Inm]`	Rd ← [PC + Inm]	Rango Inm: 0–1020, múltiplos de 4.
	Relativo al SP	`ldr Rd, [SP, #Inm]`	Rd ← [SP + Inm]	Rango Inm: 0–1020, múltiplos de 4.
Almacenar	Con desp. inm., palabra	`str Rd, [Rn, #Inm]`	[Rn + Inm] ← Rd	Rango Inm: 0–124, múltiplos de 4.
	media palabra	`strh Rd, [Rn, #Inm]`	$[Rn + Inm]_{15:0} \leftarrow Rd_{15:0}$	$Rd_{31:16}$ se ignora. Rango Inm: 0–62, pares.
	byte	`strb Rd, [Rn, #Inm]`	$[Rn + Inm]_{7:0} \leftarrow Rd_{7:0}$	$Rd_{31:8}$ se ignora. Rango Inm: 0–31.
	Con desp. en registro, palabra	`str Rd, [Rn, Rm]`	[Rn +Rm] ← Rd	
	media palabra	`strh Rd, [Rn, Rm]`	$[Rn + Rm]_{15:0} \leftarrow Rd_{15:0}$	$Rd_{31:16}$ se ignora.
	byte	`strb Rd, [Rn, Rm]`	$[Rn + Rm]_{7:0} \leftarrow Rd_{7:0}$	$Rd_{31:8}$ se ignora.
	Relativo al SP	`str Rd, [SP, #Inm]`	[SP + Inm] ← Rd	Rango Inm: 0–1020, múltiplos de 4.
Apilar	Apilar	`push <loreglist>`	Apila registros en la pila	
	Apilar y enlazar	`push <loreglist+LR>`	Apila LR y registros en la pila	
Desapilar	Desapilar	`pop <loreglist>`	Desapila registros de la pila	
	Desapilar y retorno	`pop <loreglist+PC>`	Desapila registros y salta a la dirección cargada en el PC	
Saltar	Salto condicional	`b{cond} <label>`	Si {cond}, PC ← label (rango salto: −252 a +258 bytes de la instrucción actual).	
	Salto incondicional	`b <label>`	PC ← label (rango salto: ±2 KiB de la instrucción actual).	
	Salto largo y enlaza	`bl <label>`	LR ← dirección de la siguiente instrucción, PC ← label (Instrucción de 32 bits. Rango salto: ±4 MiB de la instrucción actual).	
Extender	Con signo, media a palabra	`sxth Rd, Rm`	$Rd_{31:0} \leftarrow$ SignExtend($Rm_{15:0}$)	
	Con signo, byte a palabra	`sxtb Rd, Rm`	$Rd_{31:0} \leftarrow$ SignExtend($Rm_{7:0}$)	
	Sin signo, media a palabra	`uxth Rd, Rm`	$Rd_{31:0} \leftarrow$ ZeroExtend($Rm_{15:0}$)	
	Sin signo, byte a palabra	`uxtb Rd, Rm`	$Rd_{31:0} \leftarrow$ ZeroExtend($Rm_{7:0}$)	

{cond}	EQ	Igual	NE	Distinto	MI	Negativo	PL	Positivo	VS	Desbordamiento
	HI	Mayor sin signo	CS	Mayor o igual sin signo	CC	Menor sin signo	LS	Menor o igual sin signo	VC	No desbordamiento
	GT	Mayor que	GE	Mayor o igual	LT	Menor que	LE	Menor o igual que		