1.4.3. Thumb instruction summary

The Thumb instruction set formats are shown in Figure 1.6.

See the *ARM Architectural Reference Manual* for more information about the ARM instruction set formats.

Figure 1.6. Thumb instruction set formats

	Format	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Move shifted register	01	0	0	0	Op		Offset5					Rs			Rd		
Add and subtract	02	0	0	0	1	1	1	Op	Rn/ offset3			Rs			Rd		
Move, compare, add, and subtract immediate	03	0	0	1	Op		Rd			Offset8							
ALU operation	04	0	1	0	0	0	0	Op				Rs			Rd		
High register operations and branch exchange	05	0	1	0	0	0	1	Op		H1	H2	Rs/Hs			RdHd		
PC-relative load	06	0	1	0	0	1	Rd			Word8							
Load and store with relative offset	07	0	1	0	1	L	B	0	Ro			Rb			Rd		
Load and store sign-extended byte and halfword	08	0	1	0	1	H	S	1	Ro			Rb			Rd		
Load and store with immediate offset	09	0	1	1	B	L	Offset5					Rb			Rd		
Load and store halfword	10	1	0	0	0	L	Offset5					Rb			Rd		
SP-relative load and store	11	1	0	0	1	L	Rd			Word8							
Load address	12	1	0	1	0	SP	Rd			Word8							
Add offset to stack pointer	13	1	0	1	1	0	0	0	0	S	SWord7						
Push and pop registers	14	1	0	1	1	L	1	0	R	Rlist							
Multiple load and store	15	1	1	0	0	L	Rb			Rlist							
Conditional branch	16	1	1	0	1	Cond				Softset8							
Software interrupt	17	1	1	0	1	1	1	1	1	Value8							
Unconditional branch	18	1	1	1	0	0	Offset11										
Long branch with link	19	1	1	1	1	H	Offset										
	Format	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The Thumb instruction set summary is listed in Table 1.7.

Table 1.7. Thumb instruction set summary

Operation		Assembly syntax
Move	Immediate	`MOV Rd, #8bit_Imm`
	High to Low	`MOV Rd, Hs`

Operation		Assembly syntax
	Low to High	MOV Hd, Rs
	High to High	MOV Hd, Hs
Arithmetic	Add	ADD Rd, Rs, #3bit_Imm
	Add Low, and Low	ADD Rd, Rs, Rn
	Add High to Low	ADD Rd, Hs
	Add Low to High	ADD Hd, Rs
	Add High to High	ADD Hd, Hs
	Add Immediate	ADD Rd, #8bit_Imm
	Add Value to SP	ADD SP, #7bit_Imm ADD SP, #-7bit_Imm
	Add with carry	ADC Rd, Rs
	Subtract	SUB Rd, Rs, Rn SUB Rd, Rs, #3bit_Imm
	Subtract Immediate	SUB Rd, #8bit_Imm
	Subtract with carry	SBC Rd, Rs
	Negate	NEG Rd, Rs
	Multiply	MUL Rd, Rs
	Compare Low, and Low	CMP Rd, Rs
	Compare Low, and High	CMP Rd, Hs
	Compare High, and Low	CMP Hd, Rs
	Compare High, and High	CMP Hd, Hs
	Compare Negative	CMN Rd, Rs
	Compare Immediate	CMP Rd, #8bit_Imm
Logical	AND	AND Rd, Rs
	EOR	EOR Rd, Rs
	OR	ORR Rd, Rs
	Bit clear	BIC Rd, Rs
	Move NOT	MVN Rd, Rs
	Test bits	TST Rd, Rs
Shift/Rotate	Logical shift left	LSL Rd, Rs, #5bit_shift_imm LSL Rd, Rs

Operation		**Assembly syntax**
	Logical shift right	`LSR Rd, Rs, #5bit_shift_imm LSR Rd, Rs`
	Arithmetic shift right	`ASR Rd, Rs, #5bit_shift_imm ASR Rd, Rs`
	Rotate right	`ROR Rd, Rs`
Branch	Conditional	-
	• if Z set	`BEQ label`
	• if Z clear	`BNE label`
	• if C set	`BCS label`
	• if C clear	`BCC label`
	• if N set	`BMI label`
	• if N clear	`BPL label`
	• if V set	`BVS label`
	• if V clear	`BVC label`
	• if C set and Z clear	`BHI label`
	• if C clear and Z set	`BLS label`
	• if ((N set and V set) or (N clear and V clear))	`BGE label`
	• if ((N set and V clear) or if (N clear and V set))	`BLT label`
	• if (Z clear and ((N or V set) or (N or V clear)))	`BGT label`
	• if (Z set or ((N set and V clear) or (N clear and V set)))	`BLE label`
	Unconditional	`B label`
	Long branch with link	`BL label`
	Optional state change	-
	• to address held in Lo reg	`BX Rs`
	• to address held in Hi reg	`BX Hs`

Operation		Assembly syntax
Load	With immediate offset	-
	• word	`LDR Rd, [Rb, #7bit_offset]`
	• halfword	`LDRH Rd, [Rb, #6bit_offset]`
	• byte	`LDRB Rd, [Rb, #5bit_offset]`
	With register offset	-
	• word	`LDR Rd, [Rb, Ro]`
	• halfword	`LDRH Rd, [Rb, Ro]`
	• signed halfword	`LDRSH Rd, [Rb, Ro]`
	• byte	`LDRB Rd, [Rb, Ro]`
	• signed byte	`LDRSB Rd, [Rb, Ro]`
	PC-relative	`LDR Rd, [PC, #10bit_Offset]`
	SP-relative	`LDR Rd, [SP, #10bit_Offset]`
	Address	-
	• using PC	`ADD Rd, PC, #10bit_Offset`
	• using SP	ADD Rd, SP, #10bit_Offset
	Multiple	`LDMIA Rb!, <reglist>`
Store	With immediate offset	-
	• word	`STR Rd, [Rb, #7bit_offset]`
	• halfword	`STRH Rd, [Rb, #6bit_offset]`
	• byte	`STRB Rd, [Rb, #5bit_offset]`
	With register offset	-
	• word	`STR Rd, [Rb, Ro]`
	• halfword	`STRH Rd, [Rb, Ro]`
	• byte	`STRB Rd, [Rb, Ro]`

Operation		Assembly syntax
	SP-relative	`STR Rd, [SP, #10bit_offset]`
	Multiple	`STMIA Rb!, <reglist>`
Push/Pop	Push registers onto stack	`PUSH <reglist>`
	Push LR, and registers onto stack	`PUSH <reglist, LR>`
	Pop registers from stack	`POP <reglist>`
	Pop registers, and pc from stack	`POP <reglist, PC>`
Software Interrupt	-	`SWI 8bit_Imm`