Registers

Register	Description
R0	16 bit, General Purpose
R1	16 bit, General Purpose
R2	16 bit, General Purpose
R3	16 bit, General Purpose
R4	16 bit, General Purpose
R5	16 bit, General Purpose
R6	16 bit, General Purpose
R7	16 bit, General Purpose
R8	16 bit, General Purpose
SP	16 bit, Stack Pointer
PC	16 bit, Program Pointer
SR I V S C Z	Status Register
ASR C Z	Arithmetic Status Register
All general purpose registers are 16 bit long. ALU operations are 16 bit. 8 bit operations are not natively supported except for extended loads and truncated stores.	

Condition Codes Summary

Encoding	Machine Name	Alt Names	SR Flags	Description
000	eq	z	Z	Equal than. Zero
001	ne	nz	!Z	Not equal. Not zero
010	uge	hs, c	C	Unsigned greater than or equal. Carry
011	ult	lo, nc	!C	Unsigned less than. Not carry
100	lt	-	S != V	Signed less than
101	ge	-	S == V	Signed greater than or equal
110	ugt	hi	C && !Z	Unsigned greater than
111	gt	-	(S == V) && !Z	Signed greater than
-	ule	ls	!C || Z	Unsigned less than or equal Implemented as the opposite of ugt
-	le	-	(S != V) || Z	Signed less than or equal Implemented as the opposite of gt

Opcode Summary, by opcode number

Pattern	Encoding																Description
T1	1	1	1	o	aaaa aaaa aaaa												Relative Call/Jump
T2	1	1	0	1	cc			a aaaa aaaa									Conditional branch
T6	1	1	0	0	Rd			1	op		kk kkkk						And, Or immediate
T9	1	1	0	0	o	xx		0	kkkk kkkk								Add/Subtract offset to SP
T7	1	0	op		Rd			Rn			kk kkkk						Load/store with immediate offset
T8	0	1	op		Rd			1	kkkk kkkk								Move, Compare, Add, Subtract immediate
T9	0	1	op		Rd			0	kkkk kkkk								SP relative load/store
T5	0	0	1	op				Rn			Rs			Rd			Three register ALU operation, Load/store with register offset
T4	0	0	0	1	cc			Rn			Rs			Rd			Conditional select
T3	0	0	0	0	cc			1	1	x	x	x	x	Rd			Conditional set
T11	0	0	0	0	op			1	0	op		x	x	Rd			Push/Pop, Move SP Register, Add SP Register, Branch/Call Indirect, Move immediate, Load/store with absolute address, ALU operation
T12	0	0	0	0	op			0	1	x	x	x	x	x	x	x	Zero Operand Instructions
T13	0	0	0	0	op			0	0	x	Rs			Rd			Two register ALU operation Two register Move, Compare, ALU operation
-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NOP instruction, emulated through 'mov r0, r0'

Opcode Summary, by instruction pattern

Pattern	Encoding																Description
T1	1	1	1	o	aaaa aaaa aaaa												Relative Call/Jump
T2	1	1	0	1	cc			a aaaa aaaa									Conditional branch
T3	0	0	0	0	cc			1	1	x	x	x	x	Rd			Conditional set
T4	0	0	0	1	cc			Rn			Rs			Rd			Conditional select
T5	0	0	1	op				Rn			Rs			Rd			Three register ALU operation, Load/store with register offset
T6	1	1	0	0	Rd			1	op		kk kkkk						And, Or immediate
T7	1	0	op		Rd			Rn			kk kkkk						Load/store with immediate offset
T8	0	1	op		Rd			1	kkkk kkkk								Move, Compare, Add, Subtract immediate
T9	0	1	op		Rd			0	kkkk kkkk								SP relative load/store
T10	1	1	0	0	o	xx		0	kkkk kkkk								Add/Subtract offset to SP
T11	0	0	0	0	op			1	0	op		x	x	Rd			Push/Pop, Move SP Register, Add SP Register, Branch/Call Indirect, Move immediate, Load/store with absolute address, ALU operation
T12	0	0	0	0	op			0	1	x	x	x	x	x	x	x	Zero Operand Instructions
T13	0	0	0	0	op			0	0	x	Rs			Rd			Two register ALU operation Two register Move, Compare, ALU operation

Instructions Summary

	Encoding	Machine Name	Assembly Instruction	Description
Relative Call/Jump				
T1	0	jmprel	jmp Label	PC relative unconditional branch to Label
	1	callrel	call Label	PC relative subroutine call to Label
Conditional branch				
T2	%cc	b%cc	b%cc Label	Branch PC relative to Label if %cc matches SR flags, otherwise proceed with the next instruction
Conditional set				
T3	%cc 0	set%cc	set%cc Rd	Conditional set. Move 1 to Rd if %cc matches SR flags, otherwise move 0 to Rd
	%cc 1	nset%cc	nset%cc Rd	Negated conditional set. Move -1 to Rd if %cc matches SR flags, otherwise move 0 to Rd
Conditional select				
T4	%cc	sel%cc	sel%cc Rn, Rs, Rd	Conditional select. Copy Rn to Rd if %cc matches SR flags, otherwise copy Rs to Rd
Three register ALU operation				
T5	0000	addrrr	add Rn, Rs, Rd	Rd = Rn+Rs, update ASR
	0001	adcrrr	addc Rn, Rs, Rd	Rd = Rn+(Rs+C), update ASR
	0010	subrrr	sub Rn, Rs, Rd	Rd = Rn-Rs, update ASR
	0011	subcrrr	subc Rn, Rs, Rd	Rd = Rn-(Rs+!C), update ASR
	0100	orrrr	or Rn, Rs, Rd	Rd = Rn | Rs, update ASR
	0101	andrrr	and Rn, Rs, Rd	Rd = Rn & Rs, update ASR
	0110	xorrrr	xor Rn, Rs, Rd	Rd = Rn ^ Rs, update ASR
	0111	-	-	Reserved
Load/store with register offset				
T5	100x	mov16nr	ld.w [Rn, Rs], Rd	Load contents of word aligned memory address Rn+Rs into Rd
	1010	mov8znr	ld.zb [Rn, Rs], Rd	Load zero-extended contents of byte memory address Rn+Rs into Rd
	1011	mov8snr	ld.sb [Rn, Rs], Rd	Load sign-extended contents of byte memory address Rn+Rs into Rd
	110x	mov16rn	st.w Rd, [Rn, Rs]	Store Rd in word aligned memory address Rn+Rs
	111x	mov8rn	st.b Rd, [Rn, Rs]	Store byte truncated Rd in byte memory address Rn+Rs
And, Or Immediate				
T6	00	andkr	and Rd, K, Rd	Logical AND zero-extended K with Rd and store result in Rd, update ASR
	01	orkr	or Rd, K, Rd	Logical OR zero-extended K with Rd and store result in Rd, update ASR
	-	-	-	Reserved
	-	-	-	Reserved
Load/store with immediate offset				
T7	00	mov16mr	ld.w [Rn, K], Rd	Load contents of word aligned memory address Rn+zext(K) into Rd.
	01	movs8mr	ld.sb [Rn, K], Rd	Load sign-extended contents of byte memory address Rn+zext(K) into Rd
	10	mov16rm	st.w Rd, [Rn, K]	Store Rd in word aligned memory address Rn+zext(K)

	Encoding	Machine Name	Assembly Instruction	Description
	11	mov8rm	st.b Rd, [Rn, K]	Store byte truncated Rd in byte memory address Rn+zext(K)
Move, Compare, Add, Subtract immediate				
T8	00	movkr	mov K, Rd	Copy sign-extended K into Rd
	01	cmpkr	cmp Rd, K	Compare Rd with sign-extended K and update SR flags
	10	addkr	add Rd, K, Rd	Add zero-extended K to Rd and store result in Rd, update ASR
	11	subkr	sub Rd, K, Rd	Subtract zero-extended K from Rd and store in Rd, update ASR
SP relative load/store				
T9	00	mov16qr	ld.w [SP, K], Rd	Load the contents of stack memory address SP+zext(K) into Rd
	01	movs8qr	ld.sb [SP, K], Rd	Load the contents of byte memory address SP+zext(K) into Rd
	10	mov16rq	st.w Rd, [SP, K]	Store Rd in stack memory address SP+zext(K)
	11	mov8rq	st.b Rd, [SP, K]	Store the lower byte of Rd in byte memory address SP+zext(K)
Add/Subtract offset to SP				
T10	0	addks	add SP, K, SP	Add zero-extended K to SP, update ASR
	1	subks	sub SP, K, SP	Subtract zero-extended K from SP, update ASR
Push/Pop, move SP Register, add SP Register, Branch/Call indirect				
T11	000 00	push	push Rd	Decrement SP and store Rd onto the stack
	001 00	pop	pop Rd	Load Rd from the stack and increment SP
	010 00	movsr	mov SP, Rd	Copy SP into Rd
	011 00	movrs	mov Rd, SP	Copy Rd into SP
	100 00	addrs	add SP, Rd, SP	SP = SP+Rd, update ASR
	101 00	subrs	sub SP, Rd, SP	SP = SP-Rd, update ASR
	110 00	addsr	add SP, Rd, Rd	Rd = SP+Rd, update ASR
	111 00	subsr	sub SP, Rd, Rd	Rd = SP-Rd, update ASR
Branch/Call indirect				
T11	000 01	jmpreg	jmp Rd	Jump to Rd
	001 01	callreg	call Rd	Subroutine call to Rd
	010 01	-	-	
	011 01	-	-	
	100 01	-	-	
	101 01	-	-	
	110 01	-	-	
	111 01	movSr	mov SR, Rd	Move Status Register
One Register ALU Operation				
T11	000 10	lsr	lsr Rd	1 bit shift right of Rd, update ASR
	001 10	lsl	lsl Rd	1 bit shift left of Rd, update ASR
	010 10	asr	asr Rd	1 bit signed shift right of Rd, update ASR
	011 10	lsrc	lsrc Rd	1 bit shift right of Rd through carry, update ASR
	100 10	lslc	lslc Rd	1 bit shift left of Rd through carry, update ASR
	101 10	neg	neg Rd	Negate Rd

	Encoding	Machine Name	Assembly Instruction	Description
	110 10	not	not Rd	Complement Rd
	111 10	-	-	-
Move Immediate, Load/store with absolute address				
T11	000 11	movKr	mov.w K, Rd	Copy K into Rd (K is in the next instruction word)
	-	K		
	001 11	mov16ar	ld.w [A], Rd	Load contents of word aligned memory address A into Rd (A is in the next instruction word)
	-	A		
	010 11	mov8zar	ld.zb [A], Rd	Load zero-extended contents of byte memory address A into Rd (A is in the next instruction word)
	-	A		
	011 11	mov8sar	ld.sb [A], Rd	Load sign-extended contents of byte memory address A into Rd (A is in the next instruction word)
	-	A		
	100 11	mov16ra	st.w Rd, [A]	Store Rd in word aligned memory address A (A is in the next instruction word)
	-	A		
	101 11	mov8ra	st.b Rd, [A]	Store lower byte of Rd in byte memory address A (A is in the next instruction word)
	-	A		
	110 11	-	-	-
	111 11	-	-	-
Zero Operand Instructions				
T12	000	ret	ret	Return from subroutine
	001	reti	reti	Return from interrupt
	010	dint	dint	Disable interrups
	011	eint	eint	Enable interrupts
	100			
	101			
	110			
	111			
Two register Move, Compare, ALU operation				
T13	000	movrr	mov Rs, Rd	Copy Rs to Rd
	001	cmprr	cmp Rs, Rd	Compare Rd with Rs and update SR flags
	010	zext	zext Rs, Rd	Move zero-extended Rs low byte to Rd
	011	sext	sext Rs, Rd	Move sign-extended Rs low byte to Rd
	100	swapb	swapb Rs, Rd	Move the swapped bytes of Rs to Rd
	101	sextw	sextw Rs, Rd	Sets Rd to all ones if Rs is negative, or zero otherwise
	110	-	-	-
	111	mov16pr	ld {Rs}, Rd	Load Program Memory