Instructions Summary

Category	Assembly Mnemonic	Description
Arithmetic, Logic		

Arithmetic	add Rd, K, Rd
	lea Rs, K, Rd
	add Rs, Rn, Rd
		Add

	addc Rs, Rn, Rd	Add with carry
	sub Rd, K, Rd sub Rs, Rn, Rd	Subtract
	subc Rs, Rn, Rd	Subtract with carry
	neg Rs, Rd	Negate
	zext Rs, Rd	Zero extend byte
	sext Rs, Rd	Sign extend byte
	sextw Rs, Rd	Sign extend word
Logic	and Rd, K, Rd and Rs, Rn, Rd	Logical And
	or Rs, Rn, Rd	Logical Or
	xor Rs, Rn, Rd	Exclusive logical Or
	not Rs, Rd	Logical Not
Shifts	asr Rs, Rd	Arithmetic shift right
	lsr Rs, Rd	Logical shift right
	lsl Rs, Rd	Logical shift left
Comparison	cmp Rd, K cmp Rd, Rs	Compare
Data moves		
Moves	mov K, Rd mov Rs, Rd	Move
	bswap Rs, Rd	Byte swap
Conditional moves	sel%cc Rs, Rn, Rd	Select
	set%cc Rd	Set
Memory access		
Memory load	ld.w [Rs, K], Rd ld.w [SP, K], Rd ld.w [Rn, Rs], Rd ld.w [&A], Rd	Load word from memory
	ld.sb [Rs, K], Rd ld.sb [Rn, Rs], Rd	Load sign extended byte from memory
	ld.zb [Rs, K], Rd ld.zb [SP, K], Rd ld.zb [Rn, Rs], Rd ld.zb [&A], Rd	Load zero extended byte from memory
Memory store	st.w Rd, [Rs, K] st.w Rd, [SP, K] st.w Rd, [Rn, Rs] st.w Rd, [&A]	Store word to memory
	st.b Rd, [Rs, K] st.b Rd, [SP, K] st.b Rd, [Rn, Rs] st.b Rd, [&A]	Store byte to memory

Category	Assembly Mnemonic	Description
Stack access	push Rd	Push to stack
	pop Rd	Pop from stack
Program memory	ld.w {Rs}, Rd	Program memory read
Branching and subroutines		
Branch instructions	jmp Label jmp Rd	Unconditional branch
	br%cc Label	Conditional branch
Subroutine instructions	call Label call Rd	Call to subroutine
	ret	Return from subroutine
Interrupts		
Interrupt instructions	dint	Disable interrupt
	eint	Enable interrupt
	reti	Return from interrupt
	halt	Halts processor

Prefixed instructions

The prefixed instructions are assembler emulated instructions that are made of core instructions preceded by a prefix instruction. The prefix instruction contains a 'p_imm' 11 bit immediate field that expands the functionality of core instructions. The prefix instruction extends the immediate field 'imm' of the next instruction by replacing it with the result of the logical expression: (p_imm << 5) | (imm & 0b11111), thus providing a full 16 bit immediate range to the prefixed instruction.

The following non exhaustive list shows several examples of prefix instruction transformations:

Core Instruction	Prefix	Prefixed Instruction	Description
Arithmetic, Logic			
add Rd, K, Rd	pfix_k	add Rd, #K, Rd	Add with long immediate. The 8 bit embedded immediate is replaced by a 16 bit one
and Rd, K, Rd	pfix_k	and Rd, #K, Rd	And with long immediate. The 8 bit embedded immediate is replaced by a 16 bit one
lea Rs, K, Rd	pfix_k	lea Rs, #K, Rd	Lea with long immediate. The 5 bit embedded immediate is replaced by a 16 bit one
Moves			
mov K, Rd	pfix_k	mov #K, Rd	Copy K into Rd. The 8 bit embedded immediate is replaced by a 16 bit one
Branching and subroutines			
br%cc Label	pfix_k	br%cc Label	Conditional branch. Branch instruction reach is extended from 9 to 16 bit long offsets
call Label	pfix_k	call &Label	Subroutine call. Call instruction reach is extended from 11 to 16 bit addresses
Memory			
ld.w [Rs, K], Rd	pfix_k	ld.w [Rs, #K], Rd	Load word with immediate offset. The 5 embedded immediate is replaced by a 16 bit one
ld.w [A], Rd	pfix_k	ld.w [&A], Rd	Load word with immediate absolute address. The 8 bit embedded immediate field is replaced by a 16 bit address