NS16ØØØ FAMILY INSTRUCTION SET SUMMARY

18 Aug 198Ø

Notations:

i = Integer length suffix: B = Byte
W = Word
D = Double Word

f = Floating Point length suffix:
F = Standard Floating
L = Long Floating

gen = General operand. Any addressing mode can be specified.

imm = Immediate operand. Either a 4-bit signed value encoded inside the Basic Opcode or an 8-bit value appended after the addressing extensions.

disp = Displacement (addressing constant): 8, 16 or 32 bits. All three lengths legal.

reg = Any General Purpose Register: RØ-R7.

areg = Any Dedicated Address Register: SP, SB, FP, MOD, INT, PSR, UPSR.

freg = Any Floating Point Register: FØ-F7.

mreg = Any Memory Management Status/Control Register.

1, 2, 3 = Number of bytes in Basic Instruction.

* = Privileged.

x = Some forms or options are privileged.

Moves

2	MOVi	gen,gen	Move a value.
2	MOVQi	imm,gen	Extend and move a 4-bit constant.
3	MOVMi	gen,gen,disp	Move Multiple: disp bytes.
3	ZEii	gen,gen	Move with zero extension.
3	SEii	gen,gen	Move with sign extension.
2	ADDR	gen,gen	Move Effective Address.

Integer Arithmetic

2	ADDi	gen,gen	Add.
2	ADDQi	imm,gen	Add 4-bit constant.
2	ADDCi	gen,gen	Add with carry.
2	SUBi	gen,gen	Subtract.
2	SUBCi	gen,gen	Subtract with carry (borrow).
3	NEGi	gen,gen	Negate (2's complement).
3	ABSi	gen,gen	Take absolute value.
3	MULi	gen,gen	Multiply.
3	DIVi	gen,gen	Divide, round down.
3	REMi	gen,gen	Remainder from DIV. (Modulus)
3	DIVZi	gen,gen	Divide, round toward zero.
3	REMZi	gen,gen	Remainder from DIVZ.
3	MEIi	gen,gen	Multiply to Extended Integer.
3	DEIi	gen,gen	Divide Extended Integer.

Integer Comparison

2	CMPi	gen,gen	Compare.
2	CMPQi	imm,gen	Compare to 4-bit constant.
3	CMPMi	gen,gen,disp	Compare Multiple: disp bytes.

Logical and Boolean

2	ANDi	gen,gen	Logical AND.
2	ORi	gen,gen	Logical OR.
2	BICi	gen,gen	Clear selected bits.
2	XORi	gen,gen	Logical Exclusive OR.
3	COMi	gen,gen	Complement all bits.
3	NOTi	gen,gen	Boolean complement: LSB only.
2	Scondi	gen	Save condition code (cond) as a Boolean variable.

Shifts

3	LSHi	gen,gen	Logical Shift, left or right.
3	ASHi	gen,gen	Arithmetic Shift, left or right.
3	ROTi	gen,gen	Rotate, left or right.

Bits

2	TBITi	gen,gen	Test bit.
3	SBITi	gen,gen	Test and set bit.
3	SBITIi	gen,gen	Test and set bit, interlocked.
3	CBITi	gen,gen	Test and clear bit.
3	CBITIi	gen,gen	Test and clear bit, interlocked.
3	IBITi	gen,gen	Test and invert bit.
3	FFSi	gen,gen	Find first set bit.

Bit Fields

Bit fields are values in memory which are not aligned to byte boundaries. Examples are PACKED arrays and records used in Pascal. "Extract" instructions read and align a bit field. "Insert" instructions write a bit field from an aligned source.

3	EXTi	reg, gen, gen, disp (offset,base,dest,length)	Extract bit field (array oriented).
3	INSi	reg, gen , gen, disp (offset,source,base,length)	Insert bit field (array oriented).
3	EXTSi	gen, gen,imm (base,dest,offset&length)	Extract bit field (short form).
3	INSSi	gen , gen, imm (source,base,offset&length)	Insert bit field (short form).
3	CVTP	reg,gen,gen	Convert to Bit Field Pointer.

Arrays

3	CHECKi	reg,gen,gen	Index bounds check.
3	INDEXi	reg,gen,gen	Recursive indexing step for multiple-dimensioned arrays.

Strings

String instructions are the only ones which assign specific functions to the General Purpose Registers:

R4	Comparison Value
R3	Translation Table Pointer
R2	String 2 Pointer
R1	String 1 Pointer
RØ	Limit Count

Options on all string instructions are:

BACKWARD:	Decrement string pointers after each step rather than incrementing.
UNTIL_MATCH:	End instruction if String 1 entry matches R4.
WHILE_MATCH:	End instruction if String 1 entry does not match R4.

All string instructions end when RØ decrements to zero.

3	MOVSi	options	Move String 1 to String 2.
3	MOVSTR	options	Move string, translating.
3	CMPSi	options	Compare String 1 to String 2.
3	CMPSTR	options	Compare, translating String 1.
3	SKPSi	options	Skip over String 1 entries.
3	SKPSTR	options	Skip, translating for UNTIL/WHILE.

Packed Decimal (BCD)

3	ADDPi	gen,gen	Add Packed.
3	SUBPi	gen,gen	Subtract Packed.

Jumps and Linkage

2	JMP	gen	Jump.
1	BR	disp	Branch (PC Relative).
2	CASEi	gen	Multiway branch.
1	Bcond	disp	Conditional branch.
2	ACBi	imm,gen,disp	Add 4-bit constant, compare and conditionally branch.
2	JSR	gen	Jump to subroutine.
1	BSR	disp	Branch to subroutine.
1	CXP	disp	Call external procedure.
2	CXPD	gen	Call external procedure using descriptor.
3	LXPD	gen,disp	Load external procedure descriptor.
1	SVC		Supervisor Call.
1	FLAG		Flag Trap.
1	BPT		Breakpoint Trap.
1	ENTER	<reg list>,disp	Save registers and allocate stack frame (Enter Procedure).
1	EXIT	<reg list>	Restore registers and reclaim stack frame (Exit Procedure).
1	RET	disp	Return from subroutine.
1	RXP	disp	Return from external procedure call.
1*	RETT	disp	Return from trap.
1*	RETI		Return from interrupt.

CPU Register Manipulation

1	SAVE	<reg list>	Save General Purpose Registers.
1	RESTORE	<reg list>	Restore General Purpose Registers.
2x	LPRi	areg,gen	Load Dedicated Address Register.
2x	SPRi	areg,gen	Store Dedicated Address Register.
2	ADJSPi	gen	Adjust Stack Pointer.
2x	BISPSRi	gen	Set selected bits in PSR.
2x	BICPSRi	gen	Clear selected bits in PSR.
3*	SETCFG	imm	Set Configuration Register.

Floating Point

3	MOVf	gen,gen	Move a Floating Point value.
3	MOVLF	gen,gen	Move and shorten a Long value to Standard.
3	MOVFL	gen,gen	Move and lengthen a Standard value to Long.
3	MOVif	gen,gen	Convert any integer to Standard or Long Floating.
3	ROUNDfi	gen,gen	Convert to integer by rounding.
3	TRUNCfi	gen,gen	Convert to integer by truncating, toward zero.
3	FLOORfi	gen,gen	Convert to largest integer less than or equal to value.
3	ADDf	gen,gen	Add.
3	SUBf	gen,gen	Subtract.
3	MULf	gen,gen	Multiply.
3	DIVf	gen,gen	Divide.
3	CMPf	gen,gen	Compare.
3	NEGf	gen,gen	Negate.
3	MAGf	gen,gen	Take absolute value.
3	FRACf	gen,gen	Take remainder from TRUNC.
3	INTf	gen,gen	Take integer portion without converting to integer format.
3	POLYf	freg,gen,gen	Polynomial evaluation step.
3	POLYFL	freg,gen,gen	POLY, extending precision.
3	DOTf	freg,gen,gen	Dot Product step.
3	DOTFL	freg,gen,gen	DOT, extending precision.
3	LFSR	gen	Load FSR.
3	SFSR	gen	Store FSR.
3	SVFREG	<freg list>	Save Floating Point Registers. FSR may be included in freg list.
3	RSFREG	<freg list>	Restore Floating Point Registers. FSR may be included in freg list.

Memory Management

3*	LMR	mreg,gen	Load Memory Management Register.
3*	SMR	mreg,gen	Store Memory Management Register.
3*	RDVAL	gen	Validate address for reading.
3*	WRVAL	gen	Validate address for writing.
3*	MOVSUi	gen,gen	Move a value from Supervisor Space to User Space.
3*	MOVUSi	gen,gen	Move a value from User Space to Supervisor Space.

Miscellaneous

1	NOP		No Operation.
1	WAIT		Wait for interrupt.
1	DIA		Diagnose. Respond to hardware breakpoint.

NS16ØØØ FAMILY ADDRESSING MODE SUMMARY

5 Aug 198Ø

These addressing modes may be used for any general operand (indicated as "gen" in the Instruction Set Summary).

Register	One of the eight General Purpose Registers. If the operand is a Floating Point operand of a Floating Point instruction, then these eight modes refer to the Floating Point Registers instead. Notation: Rn or Fn.
Register Relative	The Effective Address is calculated by adding an 8-bit, 16-bit or 32-bit displacement to the contents of the specified General Purpose Register. Notation: disp(Rn) .
Immediate	An 8- , 16-, 32-, or 64-bit operand is fetched from the instruction. Notation: value .
Absolute	The memory address of the operand is contained in the instruction. Notation: @disp .
Top of Stack	If the operand is to be read by the instruction, it is popped from the stack. If written, it is pushed onto the stack. If only its effective address is used, the stack pointer remains unchanged. Notation: TOS .
Memory Space	Refers to operands by adding a displacement to one of the four memory space pointers. Notations:

disp(PC)	Program Memory	(PC Relative)
disp(SB)	Static Memory	(SB Relative)
disp(SP)	Stack Memory	(SP Relative)
disp(FP)	Frame Memory	(FP Relative)