

## Table of Contents

Programming Model .....	6
Configurations.....	6
Instruction Prefixes .....	6
FAR.....	6
OUTER .....	6
Additional Instructions.....	8
JMP FAR .....	8
JSR FAR .....	8
RTF .....	8
Differences from the 6809 .....	8
Control Registers .....	8
Checkpoint Register.....	9
Hardware:.....	9
Push / Pull Post-byte .....	9
6309 Instructions Not supported.....	9
Size.....	9
Instruction Set Description .....	10
The Index Post-byte .....	10
ABX – Add B Accumulator to X.....	11
ADCA – Add with Carry to Accumulator A .....	12
ADCB – Add with Carry to Accumulator B.....	13
ADCD – Add with Carry to Accumulator D .....	14
ADDA – Add to Accumulator A .....	17
ADDB – Add to Accumulator B.....	18
ADDD – Add to Accumulator D .....	19
ANDA – Bitwise ‘And’ to Accumulator A .....	25
ANDB – Bitwise ‘And’ to Accumulator B.....	26
ANDCC – Bitwise ‘And’ to Condition Code Reg.....	27
ANDD – Bitwise ‘And’ to Accumulator D .....	28
ASL – Arithmetic Shift Left Memory.....	30
ASLA – Arithmetic Shift Left Accumulator A.....	31

ASLB – Arithmetic Shift Left Accumulator B .....	32
ASLD – Arithmetic Shift Left Accumulator D .....	33
ASR – Arithmetic Shift Right Memory .....	34
ASRA – Arithmetic Shift Right Accumulator A .....	35
ASRB – Arithmetic Shift Right Accumulator B .....	36
ASRD – Arithmetic Shift Right Accumulator D .....	37
BCC – Branch if Carry Clear .....	37
BCS – Branch if Carry Set .....	38
BEQ – Branch if Equal .....	38
BGE – Branch if Greater or Equal .....	39
BGT – Branch if Greater Than .....	39
BHI – Branch if Higher .....	40
BHS – Branch if Higher or Same .....	40
BITA – Bitwise ‘And’ to Accumulator A .....	41
BITB – Bitwise ‘And’ to Accumulator B .....	42
BITD – Bitwise ‘And’ to Accumulator D .....	43
BLE – Branch if Less or Equal .....	44
BLO – Branch if Lower .....	45
BLS – Branch if Lower or the Same .....	45
BLT – Branch if Less Than .....	46
BMI – Branch if Minus .....	47
BNE – Branch if Not Equal .....	48
BPL – Branch if Plus .....	48
BRA – Branch Always .....	49
BRN – Branch Never .....	49
BSR – Branch To Subroutine .....	50
BVC – Branch if Overflow Clear .....	50
BVS – Branch if Overflow Set .....	51
CLR – Clear Memory .....	52
CLRA – Clear Accumulator A .....	53
CLRB – Clear Accumulator B .....	54
CLRD – Clear Accumulator D .....	55
CMPA – Compare to Accumulator A .....	59

CMPB – Compare to Accumulator B .....	60
CMPD – Compare to Accumulator D.....	61
COM – Complement Memory .....	70
COMA – Complement Accumulator A .....	71
COMB – Complement Accumulator B.....	71
COMD – Complement Accumulator D .....	72
CWAI – Wait For Interrupt.....	74
DAA – Decimal Adjust after Addition .....	74
DEC – Decrement Memory .....	75
DECA – Decrement Accumulator A .....	76
DECB – Decrement Accumulator B .....	77
DECD – Decrement Accumulator D .....	77
EORA – Bitwise Exclusive ‘Or’ to Accumulator A .....	78
EORB – Bitwise Exclusive ‘Or’ to Accumulator B.....	83
EORD – Bitwise Exclusive ‘Or’ to Accumulator D .....	84
EXG – Exchange Registers.....	86
INC – Increment Memory.....	87
INCA – Increment Accumulator A.....	88
INCB – Increment Accumulator B .....	89
INCD – Increment Accumulator D.....	89
JMP – Unconditional Jump.....	90
JSR –Jump to Subroutine.....	92
LBCC – Long Branch if Carry Clear .....	93
LBCS – Long Branch if Carry Set.....	94
LBEQ – Long Branch if Equal .....	94
LBGE – Long Branch if Greater or Equal .....	95
LBGT – Long Branch if Greater Than.....	95
LBHI – Branch if Higher .....	96
LBHS – Long Branch if Higher or Same.....	96
LBLE – Branch if Less or Equal.....	97
LBLO – Long Branch if Lower .....	98
LBLS – Long Branch if Lower or the Same .....	98
LBLT – Long Branch if Less Than.....	99

LBMI – Long Branch if Minus .....	100
LBNE – Long Branch if Not Equal .....	101
LBPL – Long Branch if Plus.....	101
LBRA – Long Branch Always.....	102
LBRN – Long Branch Never .....	102
LBSR – Long Branch To Subroutine.....	103
LBVC – Long Branch if Overflow Clear.....	103
LBVS – Long Branch if Overflow Set.....	104
LDA – Load Accumulator A .....	105
LDB – Load Accumulator B .....	106
LDD – Load Accumulator D .....	107
LEAS – Load Effective Address Into S .....	108
LEAU – Load Effective Address Into U .....	116
LEAX – Load Effective Address Into X.....	116
LEAY – Load Effective Address Into Y .....	116
LSL – Logical Shift Left Memory .....	117
LSLA – Logical Shift Left Accumulator A .....	118
LSLB – Logical Shift Left Accumulator B.....	119
LSLD – Logical Shift Left Accumulator D .....	120
LSR – Logical Shift Right Memory.....	121
LSRA – Logical Shift Right Accumulator A.....	122
LSRB – Logical Shift Right Accumulator B .....	123
LSRD – Logical Shift Right Accumulator D .....	124
MUL – Multiply.....	124
NEG – Negate Memory .....	125
NEGA – Negate Accumulator A .....	126
NEGB – Negate Accumulator B.....	126
NEGD – Negate Accumulator D .....	127
NOP – No Operation.....	127
ORA – Bitwise ‘Or’ to Accumulator A .....	128
ORB – Bitwise ‘Or’ to Accumulator B.....	129
ORCC – Bitwise ‘Or’ to Condition Code Reg.....	130
ORD – Bitwise ‘Or’ to Accumulator D .....	131

PSHS – Push onto Stack .....	132
PSHU – Push onto User Stack .....	133
PULS – Pull from Stack.....	134
PULU – Pull from User Stack.....	135
ROL – Rotate Left Memory .....	136
ROLA – Rotate Left Accumulator A .....	137
ROLB – Rotate Left Accumulator B .....	138
ROLD – Rotate Left Accumulator D .....	139
ROR – Rotate Right Memory .....	140
RORA – Rotate Right Accumulator A .....	141
RORB – Rotate Right Accumulator B.....	142
RORD – Rotate Right Accumulator D .....	143
RTF – Return From Far Subroutine.....	143
RTI – Return From Interrupt.....	145
RTS – Return From Subroutine .....	146
SBCA – Subtract with Carry from Accumulator A .....	148
SBCB – Subtract with Carry from Accumulator B.....	149
SBCD – Subtract with Carry from Accumulator D .....	150
SEX – Sign Extend .....	151
STA – Store Accumulator A.....	152
STB – Store Accumulator B .....	153
STD – Store Accumulator D.....	153
STS – Store Stack Pointer .....	154
STU – Store User Stack Pointer.....	154
STX – Store X Register .....	155
STY – Store Y Register .....	155
SUBA – Subtract from Accumulator A .....	156
SUBB – Subtract from Accumulator B.....	157
SUBD – Subtract from Accumulator D .....	158
SWI – Software Interrupt.....	159
SWI2 – Software Interrupt.....	160
SWI3 – Software Interrupt.....	161
SYNC – Halt and Wait for Interrupt.....	162

TFR – Transfer Registers .....	163
TST – Test Memory .....	164
TSTA – Test Accumulator A .....	165
TSTB – Test Accumulator B .....	165
TSTD – Test Accumulator D .....	166

## Programming Model

35	24	23	0
	X – Index Register		
	Y – Index Register		
	U – User stack pointer		
0000	S – Hardware stack pointer		
PC			
	A	B	
	DPR	0	

A,B registers concatenate to form D register

## Configurations

The rf6809 core may be configured to use 12-bit bytes which increases the address range to 36-bits. The rf6809 core may also be configured to support many instructions compatible with the 6309 processor.

## Instruction Prefixes

rf6809 makes use of instruction prefixes to extend the addressing modes available. There are two prefixes FAR, and OUTER, which indicate to use a far address or outer indexing.

## FAR

FAR when applied to extended addressing indicates to use a full 24-bit/triple byte address rather than a 16 bit one.

When the FAR prefix is applied to indirect addressing the prefix indicates that the indirect address is 24-bit. This allows the use of a 24-bit indirect address to reach anywhere in memory.

Opcode: 0x15

## OUTER

The OUTER prefix indicates that the index register is applied after retrieving an indirect address. Normally the index register is used in the calculation of the indirect address.

When configured for 12-bit bytes the OUTER prefix is not used as there are sufficient bits in the index post-byte to encode outer indexing mode.

Opcode: 0x1B

## Additional Instructions

**JMP FAR** – performs a jump using a 24-bit extended address.

Opcode: 0x8F

**JSR FAR** – performs a jump to subroutine using a 24-bit extended address. The full 24-bit program counter is stored on the stack.

Opcode: 0xCF

**RTF** – performs a far return from subroutine by loading a full 24-bit program counter from the stack.

Opcode: 0x38

Indirect addresses must reside within the first 64k bank of memory.

## Differences from the 6809

The program counter is a full 24-bit register. The JMP and JSR instructions modify only the low order 16 bits of the program counter. To modify the full 24-bits use the JMP FAR and JSR FAR instructions. A return from a far subroutine may be done using the RTF instruction.

During interrupt processing the entire 24-bit program counter is stacked. The RTI instruction also loads the entire 24-bit program counter.

If 6309 instructions are enabled then the E, F registers are pushed onto the stack for interrupts except for the FIRQ. The RTI instruction will also reload the E, F registers.

## Control Registers

There are several control registers mapped into the address space.

Address	Access	Register Usage
FF..FE0	RO	Core ID – used to identify core in multi-core application. Reflects the value of the coreid_i input.
FF..FE1	WO	Checkpoint register. If checkpointing is enabled this register must be written within one second, or an NMI will occur.
FF..FE4/5	RO	high order bits of millisecond count
FF.FF6/7	RO	low order bits of millisecond count

The millisecond count register contains a count of the number of milliseconds since the last reset.



## Checkpoint Register

The core may be configured to include a checkpoint register and timer. When checkpointing is present an NMI will be generated if the checkpoint register is not written to within one second.

## Hardware:

This is a softcore implementation of a 6809 compatible processor. As such no attempt was made to duplicate the 6809's bus cycle activity. Instructions may not execute in the same number of clock cycles as the 6809. Instructions in some circumstances execute in fewer clock cycles.

### Push / Pull Post-byte

When 6309 instructions are enabled, and the core is configured for 12-bit bytes the push and pull instructions may include pushing and pulling of accumulators E and F. The push / pull order is outlined below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC	higher memory address

Push / pull post-byte (12-bit bytes)

11	10	9	8	7	6	5	4	3	2	1	0
~	~	F	E	PC	U or S	Y	X	DP	B	A	CCR

### 6309 Instructions Not supported

PSHSW, PULSW, PSHUW, PULUW

TFM

Divide instructions

Memory bit manipulation instructions.

The Q register and associated instructions.

### Size

12-bit bytes, 6309 support: 8000 LUTs 5 block rams, 1 DSP

12-bit bytes, no 6309 support 6500 LUTs, 5 block rams, 1 DSP.

## Instruction Set Description for 12-bit Bytes

### The Index Post-byte

The indexed addressing mode specification field is twelve bits in size.

Bits rr specifies one of the index registers, XR, YR, SP, or UP

Bits dddddddd specifies a nine-bit displacement.

The 'i' bit indicates one level of indirection is added for the data fetch.

The 'o' bit indicates that indexing is applied after indirection.

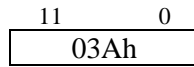
Ndx Pattern		
0rrddddddddd	EA = ,R + 9 bit offset	
1rrri00000000	EA = ,R+	
1rrri00000001	EA = ,R++	
1rrri00000010	EA = ,R-	
1rrri00000011	EA = ,--R	
1rrri0000100	EA = ,R + 0 offset	
1rrri0000101	EA = ,R + ACCB offset	
1rrri0000110	EA = ,R + ACCA offset	
1rrri0001000	EA = ,R + 12 bit offset	
1rrri0001001	EA = ,R + 24 bit offset	
1rrri0001010	EA = ,R + 36 bit offset	
1rrri0001011	EA = ,R + D offset	
1rrri0001100	EA = ,PC + 12 bit offset	
1rrri0001101	EA = ,PC + 24 bit offset	
1rrri0001110	EA = ,PC + 36 bit offset	
1rrri0001111	EA = [,Address]	

# ABX – Add B Accumulator to X

## Description

The B accumulator is added to the X register.

## Instruction Format: INH



## Flags Affected:

E	F	H	I	N	Z	V	C

# ADCA – Add with Carry to Accumulator A

## Description

The source operand is added to accumulator A including a carry.

### Instruction Format: IMM

23	12	11	0
Immed12		089h	

### Instruction Format: DP

23	12	11	0
Offset12		099h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A9h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B9h	

### Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# ADCB – Add with Carry to Accumulator B

## Description

### Instruction Format: IMM

23	12	11	0
Immed12		0C9h	

### Instruction Format: DP

23	12	11	0
Offset12		0D9h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E9h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F9h	

### Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# ADCD – Add with Carry to Accumulator D

## Description

The source operand is added to accumulator D including a carry.

\*This instruction is available only if 6309 instruction supported is configured.

## Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		189h	

## Instruction Format: DP

23	12	11	0
Offset12		199h	

## Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1A9h		

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B9h	

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# ADCR – Add with Carry Register to Register

## Description

Add register to register with carry.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH

23	20	19	16	15	12	11	0
~		r0		r1			131h

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

## Flags Affected:

**N** set equal to the most significant bit of the result

**Z** set if result value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit from the most significant bit, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑





# ADDA – Add to Accumulator A

## Description

The source operand is added to accumulator A. The carry is not included in the addition but is generated as a result flag.

### Instruction Format: IMM

23	12	11	0
Immed12		08Bh	

### Instruction Format: DP

23	12	11	0
Offset12		09Bh	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0ABh

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0BBh	

### Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# ADDB – Add to Accumulator B

## Description

The source operand is added to accumulator B. The carry is not included in the addition but is generated as a result flag.

## Instruction Format: IMM

23	12	11	0
Immed12		0CBh	

## Instruction Format: DP

23	12	11	0
Offset12		0DBh	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0EBh

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0FBh	

## Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# ADDD – Add to Accumulator D

## Description

The source operand is added to accumulator D. The carry is not included in the addition but is generated as a result flag.

## Instruction Format: IMM

35	24	23	12	11	0
Immed Lo			Immed Hi		0C3h

## Instruction Format: DP

23	12	11	0
Offset12		0D3h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E3h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo			Address Hi		0F3h

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# ADDE – Add to Accumulator E

## Description

The source operand is added to accumulator E. The carry is not included in the addition but is generated as a result flag.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: IMM

23	12	11	0
Immed12		28Bh	

## Instruction Format: DP

23	12	11	0
Offset12		29Bh	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2ABh

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2BBh	

## Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# ADDF – Add to Accumulator F

## Description

The source operand is added to accumulator F. The carry is not included in the addition but is generated as a result flag.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: IMM

23	12	11	0
Immed12		2CBh	

## Instruction Format: DP

23	12	11	0
Offset12		2DBh	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2EBh

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2FBh	

## Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# ADDR – Add Register to Register

## Description

Add register to register.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH

23	20	19	16	15	12	11	0
~		r0		r1			131h

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

## Flags Affected:

**N** set equal to the most significant bit of the result

**Z** set if result value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit from the most significant bit, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# ADDW – Add to Accumulator W

## Description

The source operand is added to accumulator W. The carry is not included in the addition but is generated as a result flag.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		18Bh	

## Instruction Format: DP

23	12	11	0
Offset12		19Bh	

## Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1ABh		

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1BBh	

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑





# ANDA – Bitwise ‘And’ to Accumulator A

## Description

### Instruction Format: IMM

23	12	11	0
Immed12		084h	

### Instruction Format: DP

23	12	11	0
Offset12		094h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A4h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B4h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# ANDB – Bitwise ‘And’ to Accumulator B

## Description

### Instruction Format: IMM

23	12	11	0
Immed12		0C4h	

### Instruction Format: DP

23	12	11	0
Offset12		0D4h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E4h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F4h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## ANDCC – Bitwise ‘And’ to Condition Code Reg

### Description

This instruction can be used to clear bits in the condition code register. A common use is to clear the interrupt mask bits.

### Instruction Format: INH

23	12	11	0
Immed12		01Ch	

### Flags Affected:

Flags for which the immediate constant has a zero bit will be cleared, other flags will not be affected.

E	F	H	I	N	Z	V	C

## ANDD – Bitwise ‘And’ to Accumulator D

### Description

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		184h	

### Instruction Format: DP

23	12	11	0
Offset12		194h	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1A4h		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B4h	

### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# ANDR – Bitwise ‘And’ Register to Register

## Description

And register to register.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH

23 20	19 16	15 12	11	0
~	r0	r1	134h	

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

## Flags Affected:

**N** set equal to the most significant bit of the result

**Z** set if result value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## ASL – Arithmetic Shift Left Memory

### Description

Memory is read, bits are shifted to the left by one bit, then the result is written back to memory. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

### Instruction Format: DP

23	12	11	0
Offset12		008h	

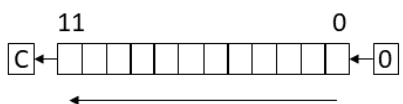
### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	068h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		078h	

### Operation:



### Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the result

**Z** set if result value is zero, otherwise cleared

**V** set to the exclusive or of bits 10 and 11

**C** set to the original value of bit 11

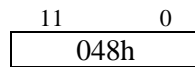
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# ASLA – Arithmetic Shift Left Accumulator A

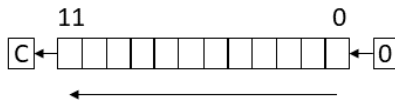
## Description

Bits in the accumulator A are shifted once to the left. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 11, otherwise cleared

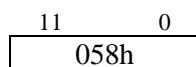
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

## ASLB – Arithmetic Shift Left Accumulator B

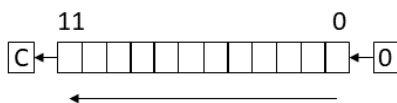
### Description

Bits in the accumulator B are shifted once to the left. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

### Instruction Format: INH



### Operation:



### Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

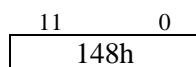


## ASLD – Arithmetic Shift Left Accumulator D

### Description

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 22 and 23

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# ASR – Arithmetic Shift Right Memory

## Description

Memory is read, bits are shifted to the left by one bit, then the result is written back to memory. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

## Instruction Format: DP

23	12	11	0
Offset12		007h	

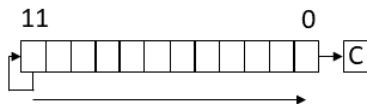
## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	067h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		077h	

## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the result

**Z** set if result value is zero, otherwise cleared

**C** set to the original value of bit 0

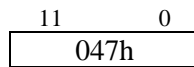
E	F	H	I	N	Z	V	C
		?		↓	↓		↓

# ASRA – Arithmetic Shift Right Accumulator A

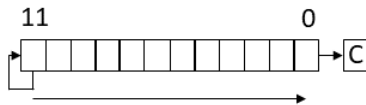
## Description

Bits in the accumulator A are shifted once to the right. The sign bit is shifted into the most significant bit and the least significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**C** set if there is a carry out of bit 0, otherwise cleared

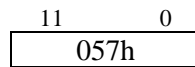
E	F	H	I	N	Z	V	C
		?		↓	↓		↓

# ASRB – Arithmetic Shift Right Accumulator B

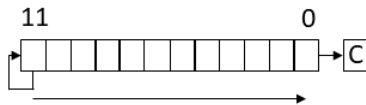
## Description

Bits in the accumulator B are shifted once to the right. The sign bit is shifted into the most significant bit and the least significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

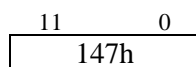
E	F	H	I	N	Z	V	C
		?		↑	↑		↑

## ASRD – Arithmetic Shift Right Accumulator D

### Description

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**C** set if there is a carry out of bit 0, otherwise cleared

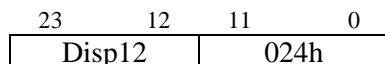
E	F	H	I	N	Z	V	C
				↑	↑		↑

## BCC – Branch if Carry Clear

### Description

BCC performs a PC relative branch using a 12-bit sign extended displacement if the carry flag bit is clear in the condition codes register.

### Instruction Format: REL



### Flags Affected:

E	F	H	I	N	Z	V	C

## BCS – Branch if Carry Set

### Description

BCC performs a PC relative branch using a 12-bit sign extended displacement if the carry flag bit is set in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	025h		

### Flags Affected:

E	F	H	I	N	Z	V	C

## BEQ – Branch if Equal

### Description

BEQ performs a PC relative branch using a 12-bit sign extended displacement if the zero-flag bit is set in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	027h		

### Flags Affected:

E	F	H	I	N	Z	V	C

## BGE – Branch if Greater or Equal

### Description

BGE performs a PC relative branch using a 12-bit sign extended displacement if the negative-flag bit and overflow flag bit are both clear, or are both set in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	02Ch		

### Operation:

if ((cc.n = 1 and cc.v = 1) or (cc.n = 0 and cc.v = 0))

PC = PC + sign extend(displ2)

### Flags Affected:

E	F	H	I	N	Z	V	C

## BGT – Branch if Greater Than

### Description

BGT performs a PC relative branch using a 12-bit sign extended displacement if the negative-flag bit and overflow flag bit are both clear, or are both set and the zero-flag bit is clear in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	02Eh		

### Operation:

if ((cc.n = 1 and cc.v = 1) or (cc.n = 0 and cc.v = 0)) and cc.z = 0)

PC = PC + sign extend(displ2)

### Flags Affected:

E	F	H	I	N	Z	V	C

## BHI – Branch if Higher

### Description

BHI performs a PC relative branch using a 12-bit sign extended displacement if the zero-flag bit and carry flag bit are both clear in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	022h		

### Operation:

if (cc.z = 0 and cc.c = 0)

PC = PC + sign extend(displ2)

### Flags Affected:

E	F	H	I	N	Z	V	C

## BHS – Branch if Higher or Same

### Description

BHS performs a PC relative branch using a 12-bit sign extended displacement if the carry flag bit is clear in the condition codes register.

This is an alternate mnemonic for the [BCC](#) instruction.

### Instruction Format: REL

23	12	11	0
Disp12	024h		

### Operation:

if (cc.c = 0)

PC = PC + sign extend(displ2)

### Flags Affected:

E	F	H	I	N	Z	V	C



# BITA – Bitwise ‘And’ to Accumulator A

## Description

This instruction works in the same manner as the [ANDA](#) instruction except that the result is discard and accumulator A is not updated. Only the result status flags are updated.

### Instruction Format: IMM

23	12	11	0
Immed12		085h	

### Instruction Format: DP

23	12	11	0
Offset12		095h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A5h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B5h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# BITB – Bitwise ‘And’ to Accumulator B

## Description

This instruction works in the same manner as the [ANDB](#) instruction except that the result is discard and accumulator B is not updated. Only the result status flags are updated.

### Instruction Format: IMM

23	12	11	0
Immed12		0C5h	

### Instruction Format: DP

23	12	11	0
Offset12		0D5h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E5h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F5h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## BITD – Bitwise ‘And’ to Accumulator D

### Description

This instruction works in the same manner as the [ANDD](#) instruction except that the result is discarded and accumulator D is not updated. Only the result status flags are updated.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: IMM

23	12	11	0
Immed12		185h	

### Instruction Format: DP

23	12	11	0
Offset12		195h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	1A5h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B5h	

### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# BLE – Branch if Less or Equal

## Description

BLE performs a PC relative branch using a 12-bit sign extended displacement if the negative-flag bit and overflow flag bit are different or the zero-flag bit is set in the condition codes register.

## Instruction Format: REL

23	12	11	0
Disp12	02Fh		

## Operation:

if ((cc.n <> cc.v) or (cc.z))

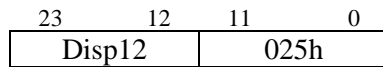
PC = PC + sign extend(disp12)

## Flags Affected:

E	F	H	I	N	Z	V	C

This is an alternate mnemonic for the [BCS](#) instruction.

### Instruction Format: REL

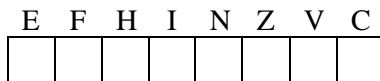


### Operation:

```
if (cc.c)
```

$$PC = PC + \text{sign extend}(\text{disp12})$$

### Flags Affected:

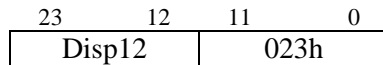


## BLS – Branch if Lower or the Same

## Description

BLS performs a PC relative branch using a 12-bit sign extended displacement if the carry-flag bit is set or the zero-flag bit is set in the condition codes register.

### Instruction Format: REL

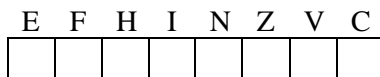


### Operation:

if (cc.c or cc.z)

$$PC = PC + \text{sign extend}(\text{disp12})$$

### Flags Affected:



## BLT – Branch if Less Than

### Description

BLT performs a PC relative branch using a 12-bit sign extended displacement if the negative-flag bit is not equal to the overflow-flag bit in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12		02Dh	

### Operation:

if (cc.n <> cc.v)

PC = PC + sign extend(displ2)

### Flags Affected:

E	F	H	I	N	Z	V	C

## BMI – Branch if Minus

### Description

BMI performs a PC relative branch using a 12-bit sign extended displacement if the negative-flag bit is set in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	02Bh		

### Operation:

if (cc.n)

$PC = PC + \text{sign extend}(\text{disp12})$

### Flags Affected:

E	F	H	I	N	Z	V	C





## BRA – Branch Always

### Description

BRA always performs a PC relative branch using a 12-bit sign extended displacement.

### Instruction Format: REL

23	12	11	0
Disp12		020h	

### Operation:

$$PC = PC + \text{sign extend}(\text{disp12})$$

### Flags Affected:

E	F	H	I	N	Z	V	C

## BRN – Branch Never

### Description

BRA never performs a PC relative branch using a 12-bit sign extended displacement. It is effectively a two-byte NOP instruction. The displacement may contain any useful value.

### Instruction Format: REL

23	12	11	0
Disp12		021h	

### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C

## BSR – Branch To Subroutine

### Description

BSR performs a PC relative branch using a 12-bit sign extended displacement after pushing the address of the next instruction on the stack.

### Instruction Format: REL

23	12	11	0
Disp12	08Dh		

### Operation:

$SP = SP - 2$

$Memory[SP] = PC$

$PC = PC + \text{sign extend}(\text{disp12})$

### Flags Affected:

E	F	H	I	N	Z	V	C

## BVC – Branch if Overflow Clear

### Description

BVC performs a PC relative branch using a 12-bit sign extended displacement if the overflow flag bit is clear in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12	028h		

### Flags Affected:

E	F	H	I	N	Z	V	C

## BVS – Branch if Overflow Set

### Description

BCC performs a PC relative branch using a 12-bit sign extended displacement if the overflow flag bit is set in the condition codes register.

### Instruction Format: REL

23	12	11	0
Disp12		029h	

### Flags Affected:

E	F	H	I	N	Z	V	C

# CLR – Clear Memory

## Description

Zero is written to memory.

## Instruction Format: DP

23	12	11	0
Offset12		00Fh	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	06Fh

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		07Fh	

## Operation:

## Flags Affected:

**N** clear

**Z** set

**V** clear

**C** clear

E	F	H	I	N	Z	V	C
				0	1	0	0

# CLRA – Clear Accumulator A

## Description

A zero is loaded into accumulator A.

## Instruction Format: INH

11	0
04Fh	

## Operation:

Acca = 0

## Flags Affected:

N cleared

Z is set

V is cleared

C is cleared

E	F	H	I	N	Z	V	C
				0	1	0	0

## CLRB – Clear Accumulator B

### Description

A zero is loaded into accumulator B.

### Instruction Format: INH

11	0
05Fh	

### Operation:

Accb = 0

### Flags Affected:

**N** cleared

**Z** is set

**V** is cleared

**C** is cleared

E	F	H	I	N	Z	V	C
				0	1	0	0

## CLRD – Clear Accumulator D

### Description

A zero is loaded into accumulator D.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

11	0
14Fh	

### Operation:

Accd = 0

### Flags Affected:

**N** cleared

**Z** is set

**V** is cleared

**C** is cleared

E	F	H	I	N	Z	V	C
				0	1	0	0

## CLRE – Clear Accumulator E

### Description

A zero is loaded into accumulator E.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

11	0
24Fh	

### Operation:

Acca = 0

### Flags Affected:

**N** cleared

**Z** is set

**V** is cleared

**C** is cleared

E	F	H	I	N	Z	V	C
				0	1	0	0



## CLRF – Clear Accumulator F

### Description

A zero is loaded into accumulator F.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

11	0
25Fh	

### Operation:

$\text{Accf} = 0$

### Flags Affected:

**N** cleared

**Z** is set

**V** is cleared

**C** is cleared

E	F	H	I	N	Z	V	C
				0	1	0	0

# CLR W – Clear Accumulator W

## Description

A zero is loaded into accumulator W.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH

11	0
15Fh	

## Operation:

Accw = 0

## Flags Affected:

**N** cleared

**Z** is set

**V** is cleared

**C** is cleared

E	F	H	I	N	Z	V	C
				0	1	0	0

# CMPA – Compare to Accumulator A

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

23	12	11	0
Immed12		081h	

### Instruction Format: DP

23	12	11	0
Offset12		091h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A1h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B1h	

### Flags Affected:

**H** the state of this bit is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# CMPB – Compare to Accumulator B

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

23	12	11	0
Immed12		0C1h	

### Instruction Format: DP

23	12	11	0
Offset12		0D1h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E1h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F1h	

### Flags Affected:

**H** the state of this bit is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# CMPD – Compare to Accumulator D

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		183h	

### Instruction Format: DP

23	12	11	0
Offset12		193h	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1A3h		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B3h	

### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# CMPE – Compare to Accumulator E

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: IMM

23	12	11	0
Immed12		281h	

## Instruction Format: DP

23	12	11	0
Offset12		291h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2A1h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2B1h	

## Flags Affected:

**H** the state of this bit is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# CMPF – Compare to Accumulator F

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: IMM

23	12	11	0
Immed12		2C1h	

## Instruction Format: DP

23	12	11	0
Offset12		2D1h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2E1h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2F1h	

## Flags Affected:

**H** the state of this bit is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# CMPR – Compare Register to Register

## Description

Compare two registers.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH

23	20	19	16	15	12	11	0
~		r0		r1			137h

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

## Flags Affected:

**N** set equal to the most significant bit of the result

**Z** set if result value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of the most significant bit, otherwise cleared

E	F	H	I	N	Z	V	C
				↕	↕	↕	↕



# CMPS – Compare to Stack Pointer

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo			Immed Hi		18Ch

### Instruction Format: DP

23	12	11	0
Offset12		19Ch	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	1ACh

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo			Address Hi		1BCh

### Flags Affected:

**N** set equal to bit 23 of the stack pointer

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# CMPU – Compare to User Stack Pointer

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		183h	

### Instruction Format: DP

23	12	11	0
Offset12		193h	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1A3h		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B3h	

### Flags Affected:

**N** set equal to bit 23 of the user stack pointer

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# CMPW – Compare to Accumulator W

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo			Immed Hi		081h

### Instruction Format: DP

23	12	11	0
Offset12		091h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A1h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo			Address Hi		0B1h

### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# CMPX – Compare to X Index Register

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo			Immed Hi		08Ch

### Instruction Format: DP

23	12	11	0
Offset12		09Ch	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0ACh

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo			Address Hi		0BCh

### Flags Affected:

**N** set equal to bit 23 of the index register

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# CMPY – Compare to Y Index Register

## Description

This instruction performs a subtract operation and discards the result. The result status flags are updated.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		18Ch	

### Instruction Format: DP

23	12	11	0
Offset12		19Ch	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1ACh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1BCh	

### Flags Affected:

**N** set equal to bit 23 of the index register

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# COM – Complement Memory

## Description

Memory is read, complemented then written.

## Instruction Format: DP

23	12	11	0
Offset12		003h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	063h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		073h	

## Operation:

## Flags Affected:

N clear

Z set

V clear

C clear

E	F	H	I	N	Z	V	C
				↑	↑	0	1

# COMA – Complement Accumulator A

## Description

The ones complement of accumulator A is loaded into accumulator A.

## Instruction Format: INH

11	0
043h	

## Operation:

$Acca = \sim Acca$

## Flags Affected:

**N** is set to bit 11 of the result

**Z** is set if the result is zero

**V** is cleared

**C** is set

E	F	H	I	N	Z	V	C
				↑	↑	0	1

# COMB – Complement Accumulator B

## Description

The ones complement of accumulator B is loaded into accumulator B.

## Instruction Format: INH

11	0
053h	

## Operation:

$Accb = \sim Accb$

## Flags Affected:

**N** is set to bit 11 of the result

**Z** is set if the result is zero

**V** is cleared

**C** is set

E	F	H	I	N	Z	V	C
				↑	↑	0	1

## COMD – Complement Accumulator D

### Description

The ones complement of accumulator D is loaded into accumulator D.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

11	0
143h	

### Operation:

$$\text{Accd} = \sim \text{Accd}$$

### Flags Affected:

**N** is set to bit 23 of the result

**Z** is set if the result is zero

**V** is cleared

**C** is set

E	F	H	I	N	Z	V	C
				↑	↑	0	1

## COME – Complement Accumulator E

### Description

The ones complement of accumulator E is loaded into accumulator E.

### Instruction Format: INH

11	0
243h	

### Operation:

$$\text{Acce} = \sim \text{Acce}$$

### Flags Affected:

**N** is set to bit 11 of the result

**Z** is set if the result is zero

**V** is cleared

**C** is set

E	F	H	I	N	Z	V	C
				↑	↑	0	1



## COMF – Complement Accumulator F

### Description

The ones complement of accumulator F is loaded into accumulator F.

### Instruction Format: INH

11	0
253h	

### Operation:

$$\text{Accf} = \sim\text{Accf}$$

### Flags Affected:

**N** is set to bit 11 of the result

**Z** is set if the result is zero

**V** is cleared

**C** is set

E	F	H	I	N	Z	V	C
				↑	↑	0	1

## COMW – Complement Accumulator W

### Description

The ones complement of accumulator W is loaded into accumulator W.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

11	0
153h	

### Operation:

$$\text{Accw} = \sim\text{Accw}$$

### Flags Affected:

**N** is set to bit 23 of the result

**Z** is set if the result is zero

**V** is cleared

**C** is set

E	F	H	I	N	Z	V	C
				↑	↑	0	1

## CWAI – Wait For Interrupt

### Description

This instruction waits for an interrupt to occur and may be used to clear bits in the condition code register. The condition code register is bitwise anded with an immediate value. The E bit in the condition code register is set and the entire machine state is stored on the stack.

### Instruction Format: INH

23	12	11	0
Immed12		03Ch	

### Flags Affected:

Flags for which the immediate constant has a zero bit will be cleared, other flags will not be affected.

E	F	H	I	N	Z	V	C

## DAA – Decimal Adjust after Addition

### Description

The value in accumulator A is adjusted after an addition to be consistent with a BCD number.

### Instruction Format: INH

11	0
019h	

### Operation:

Acca = 0

### Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero, otherwise cleared

**V** is undefined

**C** is set if there is a carry out from bit 11

E	F	H	I	N	Z	V	C
				↑	↑	?	↑

# DEC – Decrement Memory

## Description

Memory is read, decremented and written.

## Instruction Format: DP

23	12	11	0
Offset12		00Ah	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	06Ah

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		07Ah	

## Operation:

## Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800

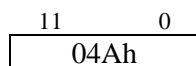
E	F	H	I	N	Z	V	C
				↑	↑	↑	

# DECA – Decrement Accumulator A

## Description

Accumulator A is decremented by one.

## Instruction Format: INH



## Operation:

$Acca = Acca - 1$

## Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800

E	F	H	I	N	Z	V	C
				↕	↕	↕	

## DECB – Decrement Accumulator B

### Description

Accumulator B is decremented by one.

### Instruction Format: INH

11	0
05Ah	

### Operation:

$$\text{Accb} = \text{Accb} - 1$$

### Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800

E	F	H	I	N	Z	V	C
				↑	↑	↑	

## DECD – Decrement Accumulator D

### Description

Accumulator D is decremented by one.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

11	0
14Ah	

### Operation:

$$\text{Accd} = \text{Accd} - 1$$

### Flags Affected:

**N** is set to the value of bit 23 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800000

E	F	H	I	N	Z	V	C
				↑	↑	↑	

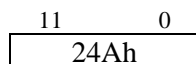
# DECE – Decrement Accumulator E

## Description

Accumulator E is decremented by one.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH



## Operation:

$$Acce = Acce - 1$$

## Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800

E	F	H	I	N	Z	V	C
				↑	↑	↑	

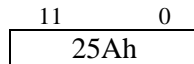
## DECF – Decrement Accumulator F

### Description

Accumulator F is decremented by one.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Operation:

$$\text{Accf} = \text{Accf} - 1$$

### Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800

E	F	H	I	N	Z	V	C
				↑	↑	↑	

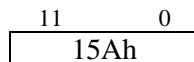
## DECW – Decrement Accumulator W

### Description

Accumulator W is decremented by one.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Operation:

$$\text{Accd} = \text{Accd} - 1$$

### Flags Affected:

**N** is set to the value of bit 23 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$800000

E	F	H	I	N	Z	V	C
				↑	↑	↑	

## DIVD – Divide Accumulator D by Memory

### Description

Divide 24-bit accumulator D by a 12-bit value from memory. Both values are treated as signed values. If overflow occurs and the result will not fit into 12-bits the overflow flag is set.

**Clock Cycles:** approximately 28

### Instruction Format: IMM

23	12	11	0
Immed12		28Dh	

### Instruction Format: DP

23	12	11	0
Offset12		29Dh	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2ADh

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2BDh	

### Flags Affected:

**N** set equal to bit 11 of the result in accumulator B

**Z** set if accumulator B value is zero, otherwise cleared

**V** set if an overflow occurred, otherwise cleared

**C** set if the quotient in accumulator B is odd, otherwise cleared if even

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑



## DIVQ – Divide Accumulator D by Memory

### Description

Divide 48-bit accumulator Q by a 24-bit value from memory. Both values are treated as signed values. If overflow occurs and the result will not fit into 24-bits the overflow flag is set.

**Clock Cycles:** approximately 56

### Instruction Format: IMM

35	24	23	12	11	0
Immed lo		Immed hi		28Eh	

### Instruction Format: DP

23	12	11	0
Offset12		29Eh	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	2AEh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2BEh	

### Flags Affected:

**N** set equal to bit 11 of the result in accumulator B

**Z** set if accumulator B value is zero, otherwise cleared

**V** set if an overflow occurred, otherwise cleared

**C** set if the quotient in accumulator B is odd, otherwise cleared if even

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

## EORA – Bitwise Exclusive ‘Or’ to Accumulator A

### Description

#### Instruction Format: IMM

23	12	11	0
Immed12		088h	

#### Instruction Format: DP

23	12	11	0
Offset12		098h	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A8h

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B8h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# EORB – Bitwise Exclusive ‘Or’ to Accumulator B

## Description

### Instruction Format: IMM

23	12	11	0
Immed12		0C8h	

### Instruction Format: DP

23	12	11	0
Offset12		0D8h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E8h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F8h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## EORD – Bitwise Exclusive ‘Or’ to Accumulator D

### Description

#### Instruction Format: IMM

23	12	11	0
Immed12		188h	

#### Instruction Format: DP

23	12	11	0
Offset12		198h	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	1A8h

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B8h	

### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## EORR – Bitwise Exclusive ‘or’ Register to Register

### Description

Exclusive or register to register.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH

23	20	19	16	15	12	11	0
~		r0		r1			136h

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

### Flags Affected:

**N** set equal to the most significant bit of the result

**Z** set if result value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# EXG – Exchange Registers

## Description

Exchange two registers.

## Instruction Format: INH

23	20	19	16	15	12	11	0
~		r0		r1			01Eh

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

## Flags Affected:

E	F	H	I	N	Z	V	C

# INC – Increment Memory

## Description

Memory is read, incremented and written.

## Instruction Format: DP

23	12	11	0
Offset12			00Ch

## Instruction Format: NDX

23	12	11	0
As needed			Ndx12
			06Ch

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		07Ch	

## Operation:

## Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FF

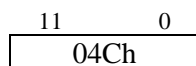
E	F	H	I	N	Z	V	C
				↑	↑	↑	

# INCA – Increment Accumulator A

## Description

Accumulator A is incremented by one.

## Instruction Format: INH



## Operation:

$Acca = Acca + 1$

## Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FF

E	F	H	I	N	Z	V	C
				↕	↕	↕	

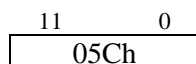


## INCB – Increment Accumulator B

### Description

Accumulator B is incremented by one.

### Instruction Format: INH



### Operation:

$$\text{Accb} = \text{Accb} + 1$$

### Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FF

E	F	H	I	N	Z	V	C
				↑	↑	↑	

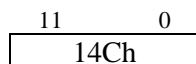
## INCD – Increment Accumulator D

### Description

Accumulator D is incremented by one.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Operation:

$$\text{Accd} = \text{Accd} + 1$$

### Flags Affected:

**N** is set to the value of bit 23 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FFFFFFF

E	F	H	I	N	Z	V	C
				↑	↑	↑	

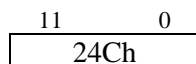
# INCE – Increment Accumulator E

## Description

Accumulator E is incremented by one.

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH



## Operation:

$$\text{Acce} = \text{Acce} + 1$$

## Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FF

E	F	H	I	N	Z	V	C
				↑	↑	↑	

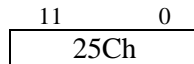
## INCF – Increment Accumulator F

### Description

Accumulator F is incremented by one.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Operation:

$$\text{Accf} = \text{Accf} + 1$$

### Flags Affected:

**N** is set to the value of bit 11 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FF

E	F	H	I	N	Z	V	C
				↑	↑	↑	

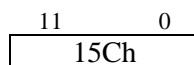
## INCW – Increment Accumulator W

### Description

Accumulator W is incremented by one.

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Operation:

$$\text{Accw} = \text{Accw} + 1$$

### Flags Affected:

**N** is set to the value of bit 23 of the result

**Z** is set if the result is zero.

**V** is set if the original value was \$7FFFFFFF

E	F	H	I	N	Z	V	C
				↑	↑	↑	

## JMP – Unconditional Jump

### Description

Load the program counter with the source operand.

#### Instruction Format: DP

23	12	11	0
Offset12		00Eh	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	06Eh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		07Eh	

#### Flags Affected:

E	F	H	I	N	Z	V	C

## JSR – Jump to Subroutine

### Description

Push the address of the next instruction on the stack, then perform a jump operation.

#### Instruction Format: DP

23	12	11	0
Offset12		09Dh	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0ADh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0BDh	

#### Flags Affected:

E	F	H	I	N	Z	V	C

# LBCC – Long Branch if Carry Clear

## Description

LBCC performs a PC relative branch using a 24-bit sign extended displacement if the carry flag bit is clear in the condition codes register.

## Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		124h	

## Flags Affected:

E	F	H	I	N	Z	V	C

## LBCS – Long Branch if Carry Set

### Description

LBCS performs a PC relative branch using a 24-bit sign extended displacement if the carry flag bit is set in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		125h	

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBEQ – Long Branch if Equal

### Description

LBEQ performs a PC relative branch using a 24-bit sign extended displacement if the zero-flag bit is set in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		127h	

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBGE – Long Branch if Greater or Equal

### Description

LBGE performs a PC relative branch using a 24-bit sign extended displacement if the negative-flag bit and overflow flag bit are both clear, or are both set in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		12Ch	

### Operation:

if ((cc.n = 1 and cc.v = 1) or (cc.n = 0 and cc.v = 0))

PC = PC + sign extend(dis24)

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBGT – Long Branch if Greater Than

### Description

LBGT performs a PC relative branch using a 24-bit sign extended displacement if the negative-flag bit and overflow flag bit are both clear, or are both set and the zero-flag bit is clear in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		12Eh	

### Operation:

if ((cc.n = 1 and cc.v = 1) or (cc.n = 0 and cc.v = 0)) and cc.z = 0)

PC = PC + sign extend(dis24)

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBHI – Branch if Higher

### Description

LBHI performs a PC relative branch using a 24-bit sign extended displacement if the zero-flag bit and carry flag bit are both clear in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo	Disp hi	122h			

### Operation:

if (cc.z = 0 and cc.c = 0)

PC = PC + sign extend(dis24)

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBHS – Long Branch if Higher or Same

### Description

LBHS performs a PC relative branch using a 24-bit sign extended displacement if the carry flag bit is clear in the condition codes register.

This is an alternate mnemonic for the [BCC](#) instruction.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo	Disp hi	124h			

### Operation:

if (cc.c = 0)

PC = PC + sign extend(dis24)

### Flags Affected:

E	F	H	I	N	Z	V	C



# LBLE – Branch if Less or Equal

## Description

LBLE performs a PC relative branch using a 24-bit sign extended displacement if the negative-flag bit and overflow flag bit are different or the zero-flag bit is set in the condition codes register.

## Instruction Format: REL

35	24	23	12	11	0
Disp lo	Disp hi	12Fh			

## Operation:

if ((cc.n <> cc.v) or (cc.z))

PC = PC + sign extend(dis24)

## Flags Affected:

E	F	H	I	N	Z	V	C

## LBLO – Long Branch if Lower

### Description

LBLO performs a PC relative branch using a 24-bit sign extended displacement if the carry-flag bit is set in the condition codes register.

This is an alternate mnemonic for the [LBCS](#) instruction.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		125h	

### Operation:

if (cc.c)

$PC = PC + \text{sign extend}(\text{disp24})$

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBSL – Long Branch if Lower or the Same

### Description

LBSL performs a PC relative branch using a 24-bit sign extended displacement if the carry-flag bit is set or the zero-flag bit is set in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		123h	

### Operation:

if (cc.c or cc.z)

$PC = PC + \text{sign extend}(\text{disp24})$

### Flags Affected:

E	F	H	I	N	Z	V	C

# LBLT – Long Branch if Less Than

## Description

LBLT performs a PC relative branch using a 24-bit sign extended displacement if the negative-flag bit is not equal to the overflow-flag bit in the condition codes register.

## Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		12Dh	

## Operation:

if (cc.n <> cc.v)

PC = PC + sign extend(dis24)

## Flags Affected:

E	F	H	I	N	Z	V	C

# LBMI – Long Branch if Minus

## Description

LBMI performs a PC relative branch using a 24-bit sign extended displacement if the negative-flag bit is set in the condition codes register.

## Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		12Bh	

## Operation:

if (cc.n)

$PC = PC + \text{sign extend}(\text{disp24})$

## Flags Affected:

E	F	H	I	N	Z	V	C

## LBNE – Long Branch if Not Equal

### Description

LBEQ performs a PC relative branch using a 24-bit sign extended displacement if the zero-flag bit is clear in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		126h	

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBPL – Long Branch if Plus

### Description

LBPL performs a PC relative branch using a 24-bit sign extended displacement if the negative-flag bit is clear in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		12Ah	

### Operation:

if (cc.n = 0)

$PC = PC + \text{sign extend}(\text{disp24})$

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBRA – Long Branch Always

### Description

LBRA always performs a PC relative branch using a 24-bit sign extended displacement.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		016h	

### Operation:

$$PC = PC + \text{sign extend}(\text{disp24})$$

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBRN – Long Branch Never

### Description

LBRN never performs a PC relative branch using a 24-bit sign extended displacement. It is effectively a three-byte NOP instruction. The displacement may contain any useful value.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		121h	

### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBSR – Long Branch To Subroutine

### Description

LBSR performs a PC relative branch using a 24-bit sign extended displacement after pushing the address of the next instruction on the stack.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		017h	

### Operation:

$SP = SP - 2$

$Memory[SP] = PC$

$PC = PC + \text{sign extend}(\text{disp24})$

### Flags Affected:

E	F	H	I	N	Z	V	C

## LBVC – Long Branch if Overflow Clear

### Description

LBVC performs a PC relative branch using a 24-bit sign extended displacement if the overflow flag bit is clear in the condition codes register.

### Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		128h	

### Flags Affected:

E	F	H	I	N	Z	V	C

# LBVS – Long Branch if Overflow Set

## Description

LBVS performs a PC relative branch using a 24-bit sign extended displacement if the overflow flag bit is set in the condition codes register.

## Instruction Format: REL

35	24	23	12	11	0
Disp lo		Disp hi		129h	

## Flags Affected:

E	F	H	I	N	Z	V	C



# LDA – Load Accumulator A

## Description

The source operand is loaded into accumulator A.

### Instruction Format: IMM

23	12	11	0
Immed12		086h	

### Instruction Format: DP

23	12	11	0
Offset12		096h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A6h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B6h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDB – Load Accumulator B

## Description

The source operand is loaded into accumulator B.

### Instruction Format: IMM

23	12	11	0
Immed12		0C6h	

### Instruction Format: DP

23	12	11	0
Offset12		0D6h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E6h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F6h	

## Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDD – Load Accumulator D

## Description

The source operand is loaded into accumulator B.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		0CCh	

### Instruction Format: DP

23	12	11	0
Offset12		0DCh	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	0ECh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0FCh	

## Flags Affected:

**N** set equal to bit 23 of the accumulator (bit 11 of accumulator A)

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDE – Load Accumulator E

## Description

The source operand is loaded into accumulator E.

## Instruction Format: IMM

23	12	11	0
Immed12		286h	

## Instruction Format: DP

23	12	11	0
Offset12		296h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2A6h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2B6h	

## Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDF – Load Accumulator F

## Description

The source operand is loaded into accumulator F.

### Instruction Format: IMM

23	12	11	0
Immed12		2C6h	

### Instruction Format: DP

23	12	11	0
Offset12		2D6h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	2E6h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		2F6h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDS – Load Stack Pointer

## Description

The source operand is loaded into the stack pointer.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		1CEh	

### Instruction Format: DP

23	12	11	0
Offset12		1DEh	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1EEh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1FEh	

### Flags Affected:

**N** set equal to bit 23 of the stack pointer

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDU – Load User Stack Pointer

## Description

The source operand is loaded into the user stack pointer.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		0CEh	

### Instruction Format: DP

23	12	11	0
Offset12		0DEh	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	0EEh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0FEh	

### Flags Affected:

**N** set equal to bit 23 of the user stack pointer

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDW – Load Accumulator W

## Description

The source operand is loaded into accumulator W.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		186h	

### Instruction Format: DP

23	12	11	0
Offset12		196h	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1A6h		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B6h	

## Flags Affected:

**N** set equal to bit 23 of the accumulator (bit 11 of accumulator E)

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	



# LDX – Load X Index Register

## Description

The source operand is loaded into the X index register.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		08Eh	

### Instruction Format: DP

23	12	11	0
Offset12		09Eh	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	0AEh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0BEh	

## Flags Affected:

**N** set equal to bit 23 of the index register

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LDY – Load Y Index Register

## Description

The source operand is loaded into the Y index register.

### Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		18Eh	

### Instruction Format: DP

23	12	11	0
Offset12		19Eh	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1AEh		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1BEh	

### Flags Affected:

**N** set equal to bit 23 of the index register

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# LEAS – Load Effective Address Into S

## Description

The address of the source operand is loaded into the stack pointer.

## Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	032h		

## Flags Affected:

E	F	H	I	N	Z	V	C

## LEAU – Load Effective Address Into U

### Description

The address of the source operand is loaded into the user stack pointer.

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12		033h	

### Flags Affected:

E	F	H	I	N	Z	V	C

## LEAX – Load Effective Address Into X

### Description

The address of the source operand is loaded into the stack pointer.

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12		030h	

### Flags Affected:

E	F	H	I	N	Z	V	C
					↑		

## LEAY – Load Effective Address Into Y

### Description

The address of the source operand is loaded into the stack pointer.

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12		031h	

### Flags Affected:

E	F	H	I	N	Z	V	C
					↑		

# LSL – Logical Shift Left Memory

## Description

Memory is read, bits are shifted to the left by one bit, then the result is written back to memory. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

## Instruction Format: DP

23	12	11	0
Offset12		008h	

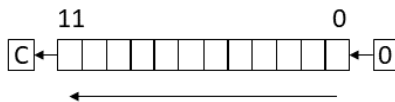
## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	068h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		078h	

## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the result

**Z** set if result value is zero, otherwise cleared

**V** set to the exclusive or of bits 10 and 11

**C** set to the original value of bit 11

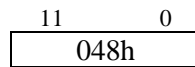
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# LSLA – Logical Shift Left Accumulator A

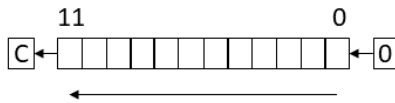
## Description

Bits in the accumulator A are shifted once to the left. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 11, otherwise cleared

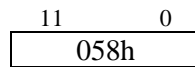
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# LSLB – Logical Shift Left Accumulator B

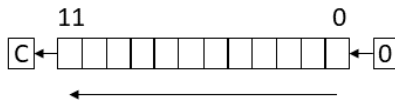
## Description

Bits in the accumulator B are shifted once to the left. A zero is shifted into the least significant bit and the most significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 11, otherwise cleared

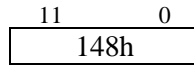
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# LSLD – Logical Shift Left Accumulator D

## Description

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH



## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 22 and 23

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑



# LSR – Logical Shift Right Memory

## Description

Memory is read, bits are shifted to the right by one bit, then the result is written back to memory. A zero is shifted into the most significant bit and the least significant bit is captured in the carry result flag.

## Instruction Format: DP

23	12	11	0
Offset12		004h	

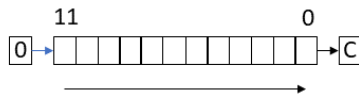
## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	064h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		074h	

## Operation:



## Flags Affected:

**N** set equal to bit 11 of the result  
**Z** set if result value is zero, otherwise cleared  
**C** set to the original value of bit 0

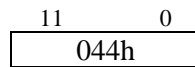
E	F	H	I	N	Z	V	C
				↑	↑		↑

# LSRA – Logical Shift Right Accumulator A

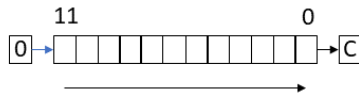
## Description

Bits in the accumulator A are shifted once to the right. A zero is shifted into the most significant bit and the least significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**C** set if there is a carry out of bit 0, otherwise cleared

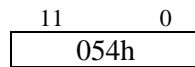
E	F	H	I	N	Z	V	C
				↑	↑		↑

# LSRB – Logical Shift Right Accumulator B

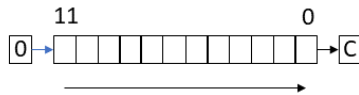
## Description

Bits in the accumulator B are shifted once to the right. A zero is shifted into the most significant bit and the least significant bit is captured in the carry result flag.

## Instruction Format: INH



## Operation:



## Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**C** set if there is a carry out of bit 0, otherwise cleared

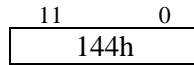
E	F	H	I	N	Z	V	C
				↑	↑		↑

# LSRD – Logical Shift Right Accumulator D

## Description

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH



## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**C** set if there is a carry out of bit 0, otherwise cleared

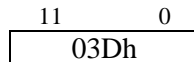
E	F	H	I	N	Z	V	C
				↑	↑		↑

# MUL – Multiply

## Description

Accumulators A and B are multiplied, and the resulting product is placed in D.

## Instruction Format: INH



## Operation:

$$\text{Accd} = \text{Acca} * \text{Accb}$$

## Flags Affected:

**Z** is set if the result is zero, otherwise cleared

**C** is set to the new value of bit 11 of accumulator B

E	F	H	I	N	Z	V	C
					↑		↑

# NEG – Negate Memory

## Description

Memory is read, negated, then written.

## Instruction Format: DP

23	12	11	0
Offset12		000h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	060h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		070h	

## Operation:

## Flags Affected:

**N** set equal to bit 11 of memory

**Z** set if value is zero, otherwise cleared

**V** set if the original value is \$800

**C** cleared if the original value was zero

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

## NEGA – Negate Accumulator A

### Description

Accumulator A is negated.

### Instruction Format: INH

11	0
040h	

### Operation:

acca = -acca

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if the original value is \$800

**C** cleared if the original value was zero

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

## NEGB – Negate Accumulator B

### Description

Accumulator B is negated.

### Instruction Format: INH

11	0
050h	

### Operation:

accb = -accb

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if the original value is \$800

**C** cleared if the original value was zero

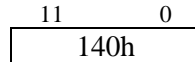
E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# NEGD – Negate Accumulator D

## Description

Accumulator D is negated.

## Instruction Format: INH



## Operation:

accd = -accd

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if the original value is \$800000

**C** cleared if the original value was zero

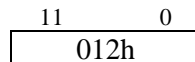
E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# NOP – No Operation

## Description

This instruction does not perform any operation.

## Instruction Format: INH



## Operation:

none

## Flags Affected:

E	F	H	I	N	Z	V	C

# ORA – Bitwise ‘Or’ to Accumulator A

## Description

### Instruction Format: IMM

23	12	11	0
Immed12		08Ah	

### Instruction Format: DP

23	12	11	0
Offset12		09Ah	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0AAh

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0BAh	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	



# ORB – Bitwise ‘Or’ to Accumulator B

## Description

### Instruction Format: IMM

23	12	11	0
Immed12		0CAh	

### Instruction Format: DP

23	12	11	0
Offset12		0DAh	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0EAh

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0FAh	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# ORCC – Bitwise ‘Or’ to Condition Code Reg

## Description

This instruction can be used to set bits in the condition code register. A common use is to set the interrupt mask bits.

## Instruction Format: INH

23	12	11	0
Immed12		01Ah	

## Flags Affected:

Flags for which the immediate constant has a one bit will be set, other flags will not be affected.

E	F	H	I	N	Z	V	C

## ORD – Bitwise ‘Or’ to Accumulator D

### Description

#### Instruction Format: IMM

23	12	11	0
Immed12		18Ah	

#### Instruction Format: DP

23	12	11	0
Offset12		19Ah	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	1AAh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1BAh	

### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# PSHS – Push onto Stack

## Description

This instruction is used to store registers to the stack.

## Instruction Format: INH

23	12	11	0
Post-byte		034h	

Registers are pushed from higher memory addresses to lower memory addresses in the order outlined below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC	higher memory address

Push / pull post-byte (12-bit bytes)

11	10	9	8	7	6	5	4	3	2	1	0
~	~	F	E	PC	U or S	Y	X	DP	B	A	CCR

## Flags Affected:

Flags for which the immediate constant has a one bit will be set, other flags will not be affected.

E	F	H	I	N	Z	V	C

# PSHU – Push onto User Stack

## Description

This instruction is used to store registers to the user stack.

## Instruction Format: INH

23	12	11	0
Post-byte		036h	

Registers are pushed from higher memory addresses to lower memory addresses in the order outlined below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC	higher memory address

Push / pull post-byte (12-bit bytes)

11	10	9	8	7	6	5	4	3	2	1	0
~	~	F	E	PC	U or S	Y	X	DP	B	A	CCR

## Flags Affected:

Flags for which the immediate constant has a one bit will be set, other flags will not be affected.

E	F	H	I	N	Z	V	C

# PULS – Pull from Stack

## Description

This instruction is used to load registers from the stack.

## Instruction Format: INH

23	12	11	0
Post-byte		035h	

Registers are pulled from lower memory addresses to higher memory addresses as in the order outlined below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC	higher memory address

Push / pull post-byte (12-bit bytes)

11	10	9	8	7	6	5	4	3	2	1	0
~	~	F	E	PC	U or S	Y	X	DP	B	A	CCR

## Flags Affected:

Flags for which the immediate constant has a one bit will be set, other flags will not be affected.

E	F	H	I	N	Z	V	C

# PULU – Pull from User Stack

## Description

This instruction is used to load registers from the user stack.

## Instruction Format: INH

23	12	11	0
Post-byte		037h	

Registers are pulled from lower memory addresses to higher memory addresses as in the order outlined below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC	higher memory address

Push / pull post-byte (12-bit bytes)

11	10	9	8	7	6	5	4	3	2	1	0
~	~	F	E	PC	U or S	Y	X	DP	B	A	CCR

## Flags Affected:

Flags for which the immediate constant has a one bit will be set, other flags will not be affected.

E	F	H	I	N	Z	V	C

# ROL – Rotate Left Memory

## Description

Memory is read, bits are shifted to the left by one bit, then the result is written back to memory. The most significant bit is captured in the carry result flag. The original carry bit is shifted into the least significant memory bit.

## Instruction Format: DP

23	12	11	0
Offset12		009h	

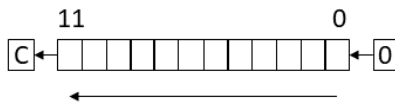
## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	069h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		079h	

## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the result

**Z** set if result value is zero, otherwise cleared

**V** set to the exclusive or of bits 10 and 11

**C** set to the original value of bit 11

E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

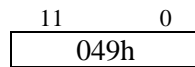


# ROLA – Rotate Left Accumulator A

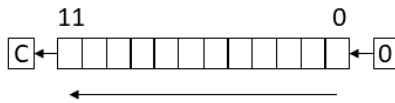
## Description

Bits in the accumulator A are shifted once to the left. The most significant bit is shifted into the carry and carry shifted into the least significant bit.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 11, otherwise cleared

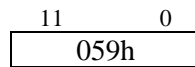
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# ROLB – Rotate Left Accumulator B

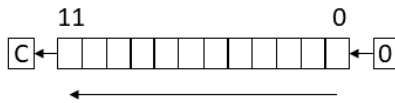
## Description

Bits in the accumulator B are shifted once to the left. The most significant bit is shifted into the carry and carry shifted into the least significant bit.

## Instruction Format: INH



## Operation:



## Flags Affected:

**H** setting is undefined

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 11, otherwise cleared

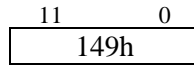
E	F	H	I	N	Z	V	C
		?		↑	↑	↑	↑

# ROLD – Rotate Left Accumulator D

## Description

- This instruction is available only if 6309 instruction support is configured.

## Instruction Format: INH



## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 22 and 23

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# ROR – Rotate Right Memory

## Description

Memory is read, bits are shifted to the right by one bit, then the result is written back to memory. The least significant bit is captured in the carry result flag. The original carry bit is shifted into the most significant memory bit.

## Instruction Format: DP

23	12	11	0
Offset12		006h	

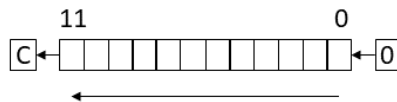
## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	066h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		076h	

## Operation:



## Flags Affected:

**N** set equal to bit 11 of the result  
**Z** set if result value is zero, otherwise cleared  
**V** set to the exclusive or of bits 10 and 11  
**C** set to the original value of bit 0

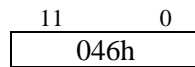
E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# RORA – Rotate Right Accumulator A

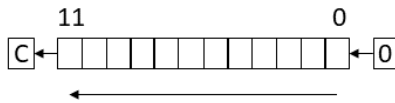
## Description

Bits in the accumulator A are shifted once to the right. The least significant bit is shifted into the carry and carry shifted into the most significant bit.

## Instruction Format: INH



## Operation:



## Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 10 and 11

**C** set if there is a carry out of bit 0, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

Bits in the accumulator B are shifted once to the right. The least significant bit is shifted into the carry and carry shifted into the most significant bit.

11	0
056h	

**N** set equal to bit 11 of the accumulator  
**Z** set if accumulator value is zero, otherwise cleared  
**V** set to the exclusive or of accumulator bits 10 and 11  
**C** set if there is a carry out of bit 0, otherwise cleared

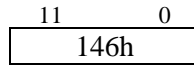
E	F	H	I	N	Z	V	C
				↕	↕	↕	↕

## RORD – Rotate Right Accumulator D

### Description

- This instruction is available only if 6309 instruction support is configured.

### Instruction Format: INH



### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set to the exclusive or of accumulator bits 22 and 23

**C** set if there is a carry out of bit 0, otherwise cleared

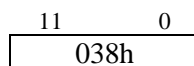
E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

## RTF – Return From Far Subroutine

### Description

RTF returns from a far subroutine by loading the program bank and program counter from stack. Note that often the program counter may be pulled from the stack at the same time as other registers using the PULS instruction.

### Instruction Format: INH



### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C





# RTI – Return From Interrupt

## Description

RTI restores the state of the machine from the stack and is used at the end of an interrupt processing routine to return to the interrupted code.

If 6309 instruction support is enabled and the entire machine state was stacked, then the E, F registers will be restored from the stack.

Registers are restored from lower to higher memory addresses as outlined in the table below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC Bank	higher memory address
PC	

## Instruction Format: INH

11	0
03Bh	

## Operation:

## Flags Affected:

E	F	H	I	N	Z	V	C

## RTS – Return From Subroutine

### Description

RTS returns from a subroutine by loading the program counter from stack. Note that often the program counter may be pulled from the stack at the same time as other registers using the PULS instruction.

### Instruction Format: INH

11	0
039h	

### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C



# SBCA – Subtract with Carry from Accumulator A

## Description

The source operand is subtracted from accumulator A including a carry.

### Instruction Format: IMM

23	12	11	0
Immed12		082h	

### Instruction Format: DP

23	12	11	0
Offset12		092h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A2h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B2h	

### Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# SBCB – Subtract with Carry from Accumulator B

## Description

The source operand is subtracted from accumulator B including a carry.

### Instruction Format: IMM

23	12	11	0
Immed12		0C2h	

### Instruction Format: DP

23	12	11	0
Offset12		0D2h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E2h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F2h	

### Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# SBCD – Subtract with Carry from Accumulator D

## Description

The source operand is subtracted from accumulator D including a carry.

\*This instruction is available only if 6309 instruction supported is configured.

## Instruction Format: IMM

35	24	23	12	11	0
Immed Lo		Immed Hi		182h	

## Instruction Format: DP

23	12	11	0
Offset12		192h	

## Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	1A2h		

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1B2h	

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 23, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

## SEX – Sign Extend

### Description

Sign-extend the value from accumulator B into accumulator A.

### Instruction Format: INH

11	0
01Dh	

### Operation:

### Flags Affected:

**N** set equal to bit 11 of the accumulator B

**Z** set if accumulator value is zero, otherwise cleared

E	F	H	I	N	Z	V	C
				↑			

# STA – Store Accumulator A

## Description

### Instruction Format: DP

23	12	11	0
Offset12		097h	

### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A7h

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B7h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	



## STB – Store Accumulator B

### Description

#### Instruction Format: DP

23	12	11	0
Offset12			0D7h

#### Instruction Format: NDX

23	12	11	0
As needed			Ndx12
			0E7h

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F7h	

#### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## STD – Store Accumulator D

### Description

#### Instruction Format: DP

23	12	11	0
Offset12			0DDh

#### Instruction Format: NDX

23	12	11	0
As needed			Ndx12
			0EDh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0FDh	

#### Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## STS – Store Stack Pointer

### Description

#### Instruction Format: DP

23	12	11	0
Offset12		1DFh	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	1EFh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1FFh	

#### Flags Affected:

**N** set equal to bit 23 of the stack pointer

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## STU – Store User Stack Pointer

### Description

#### Instruction Format: DP

23	12	11	0
Offset12		0DFh	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0EFh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0FFh	

#### Flags Affected:

**N** set equal to bit 23 of the stack pointer

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## STX – Store X Register

### Description

#### Instruction Format: DP

23	12	11	0
Offset12		09Fh	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0AFh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0BFh	

#### Flags Affected:

**N** set equal to bit 23 of the X register

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## STY – Store Y Register

### Description

#### Instruction Format: DP

23	12	11	0
Offset12		19Fh	

#### Instruction Format: NDX

23	12	11	0
As needed		Ndx12	1AFh

#### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		1BFh	

#### Flags Affected:

**N** set equal to bit 23 of the Y register

**Z** set if accumulator value is zero, otherwise cleared

**V** always cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# SUBA – Subtract from Accumulator A

## Description

The source operand is subtracted from accumulator A. Carry is not included in the subtraction but is still generated as a result flag.

## Instruction Format: IMM

23	12	11	0
Immed12		080h	

## Instruction Format: DP

23	12	11	0
Offset12		090h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0A0h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B0h	

## Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

# SUBB – Subtract from Accumulator B

## Description

The source operand is subtracted from accumulator B. Carry is not included in the subtraction but is still generated as a result flag.

## Instruction Format: IMM

23	12	11	0
Immed12		0C0h	

## Instruction Format: DP

23	12	11	0
Offset12		0D0h	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	0E0h

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0F0h	

## Flags Affected:

**H** set if there is a carry out of bit 4, otherwise cleared

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
		↑		↑	↑	↑	↑

## SUBD – Subtract from Accumulator D

### Description

The source operand is subtracted from accumulator D. Carry is not included in the subtraction but is still generated as a result flag.

### Instruction Format: IMM

35	24	23	12	11	0
Immed24				083h	

### Instruction Format: DP

23	12	11	0
Offset12		093h	

### Instruction Format: NDX

	23	12	11	0
As needed	Ndx12	0A3h		

### Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		0B3h	

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** set if overflow occurred, otherwise cleared

**C** set if there is a carry out of bit 11, otherwise cleared

E	F	H	I	N	Z	V	C
				↑	↑	↑	↑

# SWI – Software Interrupt

## Description

SWI stores the entire state of the machine onto the stack then vectors to the SWI processing routine. Interrupts are masked by the SWI instruction.

If 6309 instruction support is enabled and the entire machine state was stacked, then the E, F registers will be restored from the stack.

Registers are restored from lower to higher memory addresses as outlined in the table below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC Bank	higher memory address
PC	

## Instruction Format: INH

11	0
03Fh	

## Operation:

## Flags Affected:

E	F	H	I	N	Z	V	C
	1		1				

## SWI2 – Software Interrupt

### Description

SWI stores the entire state of the machine onto the stack then vectors to the SWI2 processing routine.

If 6309 instruction support is enabled and the entire machine state was stacked, then the E, F registers will be restored from the stack.

Registers are restored from lower to higher memory addresses as outlined in the table below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC Bank	higher memory address
PC	

### Instruction Format: INH

11	0
13Fh	

### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C



## SWI3 – Software Interrupt

### Description

SWI stores the entire state of the machine onto the stack then vectors to the SWI3 processing routine.

If 6309 instruction support is enabled and the entire machine state was stacked, then the E, F registers will be restored from the stack.

Registers are restored from lower to higher memory addresses as outlined in the table below.

CCR	Lower memory address
A	
B	
E	
F	
DP	
X	
Y	
U or S	
PC Bank	higher memory address
PC	

### Instruction Format: INH

11	0
13Fh	

### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C

## SYNC – Halt and Wait for Interrupt

### Description

SYNC activates the address and data bus tristate controls while waiting for an interrupt.

### Instruction Format: INH

11	0
013h	

### Operation:

### Flags Affected:

E	F	H	I	N	Z	V	C

# TFR – Transfer Registers

## Description

Transfer register to register.

If a 24-bit register is transferred to a twelve-bit one the twelve-bit register is set to the lower order 12-bits of the 24-bit register.

If accumulator A or B is transferred to a 24-bit register, the most significant 12-bits of the destination are set to \$FFF, while the low order byte of the destination is set to the value of the accumulator register.

For other twelve-bit registers (CC or DP) the twelve-bit register value is copied to both the upper and lower bytes of the 24-bit register.

Transfers involving the PC use only the two low order bytes of the PC.

Transfer Type	Register	
24 to 12	Any	Low order 12-bits from source copied to destination
12 to 24	A, B, E, or F	Lower order 12-bits set to accumulator, high order bits set to \$FFF
12 to 24	CCR, DP	Source copied to both high and low order bytes of destination.

## Instruction Format: INH

23 20	19 16	15 12	11	0
~	r0	r1	01Fh	

r0/r1		r0/r1	
0	D	8	A
1	X	9	B
2	Y	10	CC
3	U	11	DP
4	S	12	0
5	PC	13	0
6	W	14	E
7	resv	15	F

## Flags Affected:

No flags are affected unless the transfer is into the CCR register.

E	F	H	I	N	Z	V	C

# TST – Test Memory

## Description

Memory is tested against the value zero.

## Instruction Format: DP

23	12	11	0
Offset12		00Dh	

## Instruction Format: NDX

23	12	11	0
As needed		Ndx12	06Dh

## Instruction Format: EXT

35	24	23	12	11	0
Address Lo		Address Hi		07Dh	

## Operation:

## Flags Affected:

**N** set equal to bit 11 of memory

**Z** set if value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## TSTA – Test Accumulator A

### Description

Accumulator A is tested against the value zero.

### Instruction Format: INH

11	0
04Dh	

### Operation:

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## TSTB – Test Accumulator B

### Description

Accumulator B is tested against the value zero.

### Instruction Format: INH

11	0
05Dh	

### Operation:

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

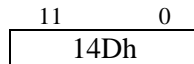
# TSTD – Test Accumulator D

## Description

Accumulator D is tested against the value zero.

\*This instruction is available only if 6309 instruction supported is configured.

## Instruction Format: INH



## Operation:

acc<sub>D</sub> = -acc<sub>D</sub>

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## TSTE – Test Accumulator E

### Description

Accumulator E is tested against the value zero.

\*This instruction is available only if 6309 instruction supported is configured.

### Instruction Format: INH

11	0
24Dh	

### Operation:

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

## TSTF – Test Accumulator F

### Description

Accumulator F is tested against the value zero.

\*This instruction is available only if 6309 instruction supported is configured.

### Instruction Format: INH

11	0
25Dh	

### Operation:

### Flags Affected:

**N** set equal to bit 11 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	

# TSTW – Test Accumulator W

## Description

Accumulator W is tested against the value zero.

\*This instruction is available only if 6309 instruction supported is configured.

## Instruction Format: INH

11	0
15Dh	

## Operation:

## Flags Affected:

**N** set equal to bit 23 of the accumulator

**Z** set if accumulator value is zero, otherwise cleared

**V** cleared

E	F	H	I	N	Z	V	C
				↑	↑	0	



