Hex	Range	Opcode encoding	BYTE 2	BYTE 3	Description	Mnemonic	Usage	Microcode	ASM	Tested	
00	0	NA	NA	NA	Start reserved	NA	NA	☑	☑	☐	
01	01	NA	NA	NA	Out signal on	ON	ON	☑	☑	☐	
02	02	NA	NA	NA	Out signal off	OFF	OFF	☑	☑	☐	
03	03	NA	NA	NA	Halt	HALT	HALT	☑	☑	☐	
04	04	NA	Hi Address Byte	Lo Address Byte	Jump to subroutine	JSR	JSR XXXX	☑	☑	☐	
05	05	NA	NA	NA	Return from subroutine	RET	RET	☑	☑	☐	
06	06	NA	LO nibble = src register		Jump to subroutine using register	JSRUR	JSRVR Rn	☐	☐	☐	???
07	07		LO nibble = src register	NA	Push Register to stack	PUSHR	PUSHR Rn	☑	☐	☐	
08	08		LO nibble = src register	NA	Pop Register from stack	POPR		☑	☐	☐	
09	09	NA	NA	NA	PUSH accumulator onto stack	PUSH	PUSH	☑	☑	☐	
0A	0A	NA	NA	NA	POP stack into accumulator	POP	POP	☑	☑	☐	
0B	0B	NA	NA	NA	Move accumulator into tmp register	MVAT	MVAT	☑	☑	☐	
0C	0C	NA	NA	NA	Move tmp register into accumulator	MVTA	MVTA	☑	☑	☐	
0D	0D	NA	operand	NA	Load tmp register imm	LDTI	LDTI XX	☑	☑	☐	
0E	0E	NA	operand	NA	Load accumulator imm	LDAI	LDAI XX	☑	☑	☐	
0F	0F	NA	Hi nibble = src register, LO nibble = dst register	NA	MOVE REG REG NOT DONE	MOVRR	MOVRR Rs,Rd	☐	☐	☐	?? done no?
10	10-17	Bottom 3 bits = Reg	operand	NA	Load register low imm (byte)	MVIB	MVIB Rn, XX	☑	☑	☐	
18	18-1F	Bottom 3 bits = Reg	operand	NA	Load register imm (word)	MVIW	MVIW Rn,XXXX	☑	☑	☐	
20	20-27	Bottom 3 bits = Reg	NA	NA	Move register low into accumulator	MVRLA	MVRLA Rn	☑	☑	☐	
28	28-2F	Bottom 3 bits = Reg	NA	NA	Move register hi into accumulator	MVRHA	MVRHA Rn	☑	☑	☐	
30	30-37	Bottom 3 bits = Reg	NA	NA	Move accumulator into register low	MVARL	MVARL Rn	☑	☑	☐	
38	38-3F	Bottom 3 bits = Reg	NA	NA	Move accumulator and register hi	MVARH	MVARH Rn	☑	☑	☐	
40	40-47	Bottom 3 bits = Reg	NA	NA	Load accumulator via register	LDAVR	LDAVR Rn	☑	☑	☐	
48	48-4F	Bottom 3 bits = Reg	NA	NA	Store accumulator via register	STAVR	STAVR Rn	☑	☑	☐	
50	50-57	Bottom 3 bits = Reg	NA	NA	Increment register	INCR	INCR Rn	☑	☑	☐	
58	58-5F	Bottom 3 bits = Reg	NA	NA	Decrement register	DECR	DECR Rn	☑	☑	☐	
60	60-6F	Bottom 4 bits = output IOADDR	NA	NA	Output accumulator	OUTA	OUTA IOADDR	☑	☑	☐	
70	70-7F	Bottom 4 bits = output IOADDR	operand	NA	Output imm	OUTI	OUTI IOADDR, XX	☑	☑	☐	
80	80-8F	Bottom 4 bits = output IOADDR	NA	NA	Output via reg	OUTVR	OUTVR IOADDR,Rn	☐	☑	☐	??? same as ldavr , outa
90	90-9F	Bottom 4 bits = output IOADDR	NA	NA	Input into accumulator	INP	INP IOADDR	☑	☑	☐	
A0	A0	NA	Hi Address Byte	Lo Address Byte	Branch	BR	BR ADDR	☑	☑	☐	
A1	A1	NA	Hi Address Byte	Lo Address Byte	Branch if accumulator is zero	BRZ	BRZ ADDR	☑	☑	☐	
A2	A2	NA	Hi Address Byte	Lo Address Byte	Branch if accumulator is not zero	BRNZ	BRNZ ADDR	☑	☑	☐	
A3	A3	NA	Hi Address Byte	Lo Address Byte	Branch if in line is hi	BRINH	BRINH ADDR	☑	☑	☐	
A4	A4	NA	Hi Address Byte	Lo Address Byte	Branch if in line is low	BRINL	BRINL ADDR	☑	☑	☐	
A5	A5					BRNC?		☐	☐	☐	
A6	A6	NA	Hi Address Byte	Lo Address Byte	Branch if carry is hi	BRC		☑	☑	☐	
A7	A7	NA	Hi Address Byte	Lo Address Byte	Branch is accumulator is less than tmp register	BRLT	BRLT ADDR	☑	☑	☐	
A8	A8	NA	Hi Address Byte	Lo Address Byte	Branch is accumulator is equal to tmp register	BREQ	BREQ ADDR	☑	☑	☐	
A9	A9	NA	Hi Address Byte	Lo Address Byte	Branch is accumulator is greater than tmp register	BRGT	BRGT ADDR	☑	☑	☐	
AA	AA	NA	Hi Address Byte	Lo Address Byte	Branch is accumulator is not equal to tmp register	BRNEQ	BRNEQ ADDR	☑	☑	☐	
AB	AB	NA	Hi Address Byte	Lo Address Byte	Branch if data bus is zero	BR16Z	BR16Z ADDR	☑	☐	☐	Should this be register based ? 3 byte ins?
AC	AC	NA	Hi Address Byte	Lo Address Byte	Branch if data bus is not zero	BR16NZ	BR16NZ ADDR	☑	☐	☐	
AD	AD							☐	☐	☐	
AE	AE							☐	☐	☐	
AF	AF	NA	Hi Address Byte	Lo Address Byte	Branch if hardware/no branch if emulator	BRDEV	BRDEV ADDR	☑	☑	☐	
B0	B0	NA	operand	NA	Add imm to accumulator	ADDI	ADDI XX	☑	☑	☐	
B1	B1	NA	operand	NA	Subtract Imm from accumulator	SUBI	SUBI XX	☑	☑	☐	
B2	B2	NA	operand	NA	Or accumulator with imm	ORI	ORI XX	☑	☑	☐	
B3	B3	NA	operand	NA	And accumulator with imm	ANDI	ANDI XX	☑	☑	☐	
B4	B4	NA	operand	NA	Xor accumulator and imm	XORI	XORI XX	☑	☑	☐	
B5	B5	NA	NA	NA	Invert accumulator	INVA	INVA	☑	☑	☐	
B6	B6	NA	NA	NA	Shift accumulator left (MSB->LSB)	SHL	SHL	☑	☑	☐	
B7	B7	NA	NA	NA	Shift accumulator right (LSB->MSB)	SHR	SHR	☑	☑	☐	
B8	B8	NA	NA	NA	Add tmp register to accumulator	ADDT	ADDT	☑	☑	☐	
B9	B9	NA	NA	NA	Subtract tmp register from accumulator	SUBT	SUBT	☑	☑	☐	
BA	BA	NA	NA	NA	OR accumulator with tmp register	ORT	ORT	☑	☑	☐	
BB	BB	NA	NA	NA	And accumulator with tmp register	ANDT	ANDT	☑	☑	☐	
BC	BC	NA	NA	NA	Xor accumulator with tmp register	XORT	XORT	☑	☑	☐	
BD	BD	NA			Ring shift left accumulator	RSHL	RSHL	☑	☑	☐	
BE	BE	NA			Ring shift right accumulator	RSHR	RSHR	☑	☑	☐	
BF	BF	NA			Propagate MSB right	PSHR	PSHR	☑	☑	☐	
C0	C0-C7	Bottom 3 bits = Reg	NA	NA	Load tmp register via register	LDTVR	LDTVR Rn	☐	☑	☐	2 byte?

C8	C8-CF	Bottom 3 bits = Reg	NA	NA	Store tmp register via register	STTVR	STTVR Rn	☐	☑	☐	2 byte?
D0	D0-D7	Bottom 3 bits = Reg	operand	NA	Load memory imm via register	LDIVR	LDIVR Rn,xx	☑	☑	☐	
D8	D8-DF	Bottom 3 bits = Reg	NA	NA	Branch via register	BRVR	BRVR Rn	☑	☑	☐	
E0	E0					CSHL		☑	☑	☐	
E1	E1					CSHR		☑	☑	☐	
E2	E2					ADDIC		☑	☐	☐	
E3	E3					ADDTC		☑	☐	☐	
E4	E4		Hi Address Byte	Lo Address Byte	Load accumulator from memory	LDA	LDA ADDR	☐	☐	☐	Moved NOV 25
E5	E5		Hi Address Byte	Lo Address Byte	Store accumulator to memory	STA	STA ADDR	☐	☐	☐	Moved NOV 25
E6	E6		Hi Address Byte	Lo Address Byte	Load tmp from memory	LDT	LDT ADDR	☐	☐	☐	Moved NOV 25
E7	E7		Hi Address Byte	Lo Address Byte	Store tmp to memory	STT	STT ADDR	☐	☐	☐	Moved NOV 25
E8	E8-EF	Bottom 3 bits = Reg	Hi Address Byte	Lo Address Byte	Store register to memory	STR	STR Rn, ADDR	☐	☐	☐	Moved NOV 25
F0	F0-F7	Bottom 3 bits = Reg	Hi Address Byte	Lo Address Byte	Load register from memory	LDR	LDR Rn, ADDR	☐	☐	☐	
F8						subic		☐	☐	☐	
F9						subtc		☐	☐	☐	
FA								☐	☐	☐	
FB	FB					INTE	INTE	☐	☐	☐	
FC	FC					INTD	INTD	☐	☐	☐	
FD	FD					IRET	IRET	☐	☐	☐	
FE	FE		Hi Address Byte	Lo Address Byte	Load INTERRUPT address reg	IADDR	IADDR ADDR	☐	☐	☐	
FF	FF				INTERRUPT handler			☐	☐	☐	
		Maybe for above src and dst in operand		eb&f0							
					movevrrr1r2						
					movevrr1r2inc						
					movevrrr1r1dec						