MikroLeo Instruction Set

Encoain	g of M	ikroLeo	Instruc	ction Word (16-bits)		Mnemonic			
	ROMH (8-bit)		ROML (8-bit)					
			Decoder			ŏ	Instructions with all	Omenation	Affected
High Ni	High Nibble (HiNB) MICRO2_IN AMODE MOD		Address	MAddr	Operand/ LAddr	eп	possibilities	Operation	Flags
MTCDO2 TN			MICRO		Operana/ Exaai	٤	possibilities		rags
		MOD		L7.L4	L2-L0				
ь15	b14	b13:b12	b11:b8	b7:b4	b3:b0		157.444	1.00	
		0					LDI ACC,n	ACC ← n	ZF
		1	0	×	n	LDI	LDI RA,n	RA ← n	
		2					LDI RB,n	RB ← n	-
	×	3					LDI RC,n	RC ← n	
		0			n		NAND ACC,n	ACC ← ACC NAND n]
		1		×			NAND ACC, RA	ACC ← ACC NAND RA	
		2	1		×	NAND	NAND ACC, RB	ACC ← ACC NAND RB	ZF
	0	_		MAddr	LAddr			ACC ← ACC NAND RAM[RC:MAddr:LAddr]	1
	1	3		×	×		NAND ACC, RAM[Addr]	ACC ← ACC NAND RAM[RC:RB:RA]	1
	0	_	_	MAddr	LAddr			ACC ← RAM[RC:MAddr:LAddr]	
	1	0	2	X	×	LDW	LDW ACC,RAM[Addr]	ACC ← RAM[RC:RB:RA]	ZF
		1		^	^		LDA RA	ACC ← RA	
			2						
	×	2	3	×	×	LDA	LDA RB	ACC ← RB	ZF
		3					LDA RC	ACC ← RC	
		0]		n		OUTA n	OUTA ← n	1
		1		×	×		OUTA ACC	OUTA ← ACC]
		2	4	MAddr			OUTA RA	OUTA ← RA	_
	0	3			LAddr		OUTA RAM[Addr]	OUTA ←RAM[RC:MAddr:LAddr]	-
	1	3		×	×		OUTA RAM[Addr]	OUTA + RAM[RC:RB:RA]	
		0			n		OUTB n	OUTB ← n	
	×	1	5	×			OUTB ACC	OUTB ← ACC	
		2		MAddr	×	ООТВ		OUTB + RA	
	0				LAddr		COTERA	OUTB + RAM[RC:MAddr:LAddr]	
		3					OUTB RAM[Addr]		
	1	_		×	×			OUTB + RAM[RC:RB:RA]	- - -
		0 1 2	6		n		OUTC n	OUTC ← n	
	×			×	×		OUTC ACC	OUTC ← ACC	
					LAddr	OUTC	OUTC RA	OUTC ← RA	
	0	3		MAddr			OUTC RAM[Addr]	OUTC + RAM[RC:MAddr:LAddr]	
	1	3		×	×		OOTE KAM[Addi-]	OUTC + RAM[RC:RB:RA]	
0		1	7	×	×	LDR	LDR RA	RA ← ACC	
U		2					LDR RB	RB ← ACC	-
		3					LDR RC	RC + ACC	
	×	0			n		CMP ACC,n	ACC - n	
		1		×		1	CMP ACC,RA	ACC - RA	1
		2	8	^	×	CMP	CMP ACC,RB	ACC - RB	CF, ZF
		۷	°	88.0 4 4 4 4	I Adda		CMF ACC,RB		
	0	3		MAddr	LAddr		CMP ACC, RAM[Addr]	ACC - RAM[RC:MAddr:LAddr]	
	1			X	X			ACC - RAM[RC:RB:RA]	
		0	9 ×		n		OUTD n	OUTD ← n	
	×	1		× MAddr	×		OUTD ACC	OUTD + ACC	
		2					OUTD RA	OUTD ← RA	
	0	3			LAddr		OUTS DAMEATED	OUTD + RAM[RC:MAddr:LAddr]	1
	1	3	1				OUTD RAM[Addr]	OUTD + RAM[RC:RB:RA]	1
	0	_	Ah -	MAddr	LAddr	1		RAM[RC:MAddr:LAddr] ← ACC	
	1	0		×	×	STW	STW RAM[Addr], ACC	RAM[RC:RB:RA] + ACC	
	<u> </u>	0		^	n		SUB ACC,n	ACC + ACC - n	
	_	1		~	n n	1	SUB ACC,RA	ACC ← ACC - RA	1
	×		Bh	×	×	CLID			CE 35
		2	Bri		1.4.1.1	SUB	SUB ACC,RB	ACC + ACC - RB	CF, ZF
	0	3		MAddr	LAddr	↓	SUB ACC, RAM[Addr]	ACC ← ACC - RAM[RC:MAddr:LAddr]	_
	1			×	×			ACC ← ACC - RAM[RC:RB:RA]	
I		_	Ch	MAddr	LAddr	JPI	JPI RC:MA[7:4]:LA[3:0]	PC ← [RC:MAddr:LAddr]	
	0	\cap			X	O.T	0 KC.WA[1.4].LA[3.0]	PC ← [RC:RB:RA]	
	0	0	Cri	×	^			i o v [konkonkri]	
								-	1
	0	0	Dh	MAddr	LAddr	JPC	JPC MA[7:4]:LA[3:0]	If CF=1, PC ← [PCH:MAddr:LAddr]	-
	1 0 1	0	Dh	MAddr ×	LAddr ×			If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA]	_
	1 0 1 0			MAddr × MAddr	LAddr X LAddr	JPC JPZ	JPC MA[7:4]:LA[3:0] JPZ MA[7:4]:LA[3:0]	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr]	-
	1 0 1	0	Dh	MAddr ×	LAddr X LAddr X		JPZ MA[7:4]:LA[3:0]	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA]	-
	1 0 1 0	0 0	Dh	MAddr x MAddr x	LAddr X LAddr		JPZ MA[7:4]:LA[3:0] ADD ACC,n	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n	-
	1 0 1 0	0 0 0	Dh Eh	MAddr × MAddr	LAddr X LAddr X	JPZ	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA	-
	1 0 1 0 1	0 0	Dh	MAddr x MAddr x	LAddr X LAddr X n		JPZ MA[7:4]:LA[3:0] ADD ACC,n	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA ACC ← ACC + RB	cF, ZF
	1 0 1 0	0 0 0 1 2	Dh Eh	MAddr x MAddr x	LAddr × LAddr × n	JPZ	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA ADD ACC,RB	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA	CF, ZF
	1 0 1 0 1	0 0 0	Dh Eh	MAddr x MAddr x	LAddr X LAddr X n	JPZ	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA ACC ← ACC + RB	CF, ZF
	1 0 1 0 1 x	0 0 0 1 2	Dh Eh	MAddr x MAddr x x MAddr	LAddr X LAddr X n X LAddr	JPZ	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA ADD ACC,RB	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA ACC ← ACC + RB ACC ← ACC + RAM[RC:MAddr:LAddr]	CF, ZF
	1 0 1 0 1 x	0 0 0 1 2 3	Dh Eh	MAddr x MAddr x x MAddr x x	LAddr X LAddr X n X LAddr	JPZ ADD INA	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA ADD ACC,RB ADD ACC,RAM[Addr] INA	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA ACC ← ACC + RB ACC ← ACC + RAM[RC:MAddr:LAddr] ACC ← ACC + RAM[RC:RB:RA] ACC ← ACC + RAM[RC:RB:RA] ACC ← ACC + RAM[RC:RB:RA]	
1	1 0 1 0 1 x	0 0 0 1 2 3 0 1	Dh Eh	MAddr x MAddr x x MAddr	LAddr X LAddr X n X LAddr	ADD INA INB	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA ADD ACC,RB ADD ACC,RAM[Addr] INA INB	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA ACC ← ACC + RB ACC ← ACC + RAM[RC:MAddr:LAddr] ACC ← ACC + RAM[RC:RB:RA] ACC ← INA ACC ← INA	CF, ZF
1	1 0 1 0 1 x	0 0 0 1 2 3	Dh Eh	MAddr x MAddr x x MAddr x x	LAddr X LAddr X n X LAddr	JPZ ADD INA	JPZ MA[7:4]:LA[3:0] ADD ACC,n ADD ACC,RA ADD ACC,RB ADD ACC,RAM[Addr] INA	If CF=1, PC ← [PCH:MAddr:LAddr] If CF=1, PC ← [PCH:RB:RA] If ZF=1, PC ← [PCH:MAddr:LAddr] If ZF=1, PC ← [PCH:RB:RA] ACC ← ACC + n ACC ← ACC + RA ACC ← ACC + RB ACC ← ACC + RAM[RC:MAddr:LAddr] ACC ← ACC + RAM[RC:RB:RA] ACC ← ACC + RAM[RC:RB:RA] ACC ← ACC + RAM[RC:RB:RA]	

LA[3:0] \Rightarrow can be RA or Operand (LAddr), depends on the AMODE bit. MA[7:4] \Rightarrow can be RB or MAddr, depends on the AMODE bit.

Address = Addr => RC:MA[7:4]:LA[3:0]
If AMODE=0, Addr=RC:Maddr:LAddr

If AMODE=1, Addr=RC:RB:RA

x = don't care

 $\label{eq:MAddr} \textbf{MAddr} = \textbf{m, so the names are interchangeable}$ $\label{eq:Operand = n, so the names are interchangeable}$

AMODE = Addresing mode (b14) MOD = Modifier bits (b13:b12) MAddr = Medium Address (b7:b4) LAddr = Low Address (b3:b0)

MICRO = Instruction (b11:b8)

OPCODE = AMODE:MOD:MICRO