MAC-2 Instruction Repertoire				
OpCode Binary	**OpCode Hex**	**Assembly Mnemonic**	**Instruction**	**Meaning or Action**
0000xxxxxxxxxxx	0xxx	lodd	Load direct	ac:=m[x]
0001xxxxxxxxxxx	1xxx	stod	Store direct	m[x]:=ac
0010xxxxxxxxxxx	2xxx	addd	Add direct	ac:=ac+m[x]
0011xxxxxxxxxxx	3xxx	subd	Subtract direct	ac:=ac−m[x]
0100xxxxxxxxxxx	4xxx	jpos	Jump if positive	**if** ac≥0 **then** pc:=x
0101xxxxxxxxxxx	5xxx	jzer	Jump if zero	**if** ac=0 **then** pc:=x
0110xxxxxxxxxxx	6xxx	jump	Jump	pc:=x
0111xxxxxxxxxxx	7xxx	loco	Load constant	ac:=x (0≤x≤4095)
1000xxxxxxxxxxx	8xxx	lodl	Load local	ac:=m[x+sp]
1001xxxxxxxxxxx	9xxx	stol	Store local	m[x+sp]:=ac
1010xxxxxxxxxxx	axxx	addl	Add local	ac:=ac+m[x+sp]
1011xxxxxxxxxxx	bxxx	subl	Subtract local	ac:=ac−m[x+sp]
1100xxxxxxxxxxx	cxxx	jneg	Jump if negative	**if** ac<0 **then** pc:=x
1101xxxxxxxxxxx	dxxx	jnze	Jump if nonzero	**if** ac≠0 **then** pc:=x
1110xxxxxxxxxxx	exxx	call	Call procedure	sp:=sp−1;m[sp]:=pc;pc:=x
1111000000000000	f000	pshi	Push indirect	sp:=sp−1;m[sp]:=m[ac]
1111001000000000	f200	popi	Pop indirect	m[ac]:=m[sp];sp:=sp+1
1111010000000000	f400	push	Push onto stack	sp:=sp−1;m[sp]:=ac
1111011000000000	f600	pop	Pop from stack	ac:=m[sp];sp:=sp+1
1111100000000000	f800	retn	Return	pc:=m[sp];sp:=sp+1
1111101000000000	fa00	swap	Swap ac, sp	tmp:=ac;ac:=sp;sp:=tmp
11111100yyyyyyyy	fcyy	insp	Increment sp	sp:=sp+y (0≤y≤255)
11111110yyyyyyyy	feyy	desp	Decrement sp	sp:=sp−y (0≤y≤255)
1111111100000000	ff00	ftac	Copy f to ac	ac:=f
1111111101000000	ff40	actf	Copy ac to f	f:=ac
111111111000zzzz	ff8z	racr	Rotate ac z-bits right	ac_i:=$ac_{i+z \bmod 16}$ (0≤z≤15)
1111111111000000	ffc0	andf	AND f with ac	ac:=ac.AND.f
1111111111111111	ffff	halt	Halt machine	stops fetch, decode, execute cycles

xxxxxxxxxxx is a 12-bit machine address (or constant); in column 2 it is called xxx and in column 5 it is called x. yyyyyyyy is an 8-bit constant; in column 2 it is called yy and in column 5 it is called y. zzzz is a 4-bit constant; in column 2 it is called z and in column 5 it is called z.