付録 B. 命令一覧表

■ 本文中の記号の説明

1. サイズ欄

B	オペランドサイズがバイト (8ビット)
W	オペランドサイズがワード (16ビット)
L	オペランドサイズがロング (32ビット)

2. ニモニック欄

R	8/16/32ビットのカレントバンクレジスタを含む8本の汎用レジスタ 8ビットの場合: W, A, B, C, D, E, H, L 16ビットの場合: WA, BC, DE, HL, IX, IY, IZ, SP 32ビットの場合: XWA, XBC, XDE, XHL, XIX, XIY, XIZ, XSP
r	8/16/32ビットの汎用レジスタ
cr	8/16/32ビットのCPUの全コントロールレジスタ DMAS0~3, DMAD0~3, DMAC0~3, DMAM0~3, INTNEST
A	Aレジスタ(8ビット)
F	フラグレジスタ(8ビット)
F'	裏フラグレジスタ(8ビット)
SR	ステータスレジスタ(16ビット)
PC	プログラムカウンタ(32ビット)
(mem)	8/16/32ビットのメモリ内のデータ
mem	実効アドレス値
<W>	オペランドサイズがワードのとき "W" の記述が必要であることを示します。
[ ]	カッコ内のオペランドの記述が、省略できることを示します。
#	8/16/32ビットの即値データ
#3	3 ビットの即値データ; 0~7 or 1~8 ……短縮コード用
#4	4 ビットの即値データ; 0~15 or 1~16
d8	8 ビットのディスプレースメント; −80H~+7FH
d16	16ビットのディスプレースメント; −8000H~+7FFFH
cc	コンディションコード
(#8)	ダイレクトアドレッシング ; (00H) ~ (0FFH) …256バイト空間
(#16)	64KBエリアアドレッシング ; (0000H) ~ (0FFFFH)
$	その命令が置かれている先頭アドレス

3. コード欄

Z	オペランドサイズを表す指定コードです。 バイト (8ビット) = 0 ワード (16ビット) = 2 ロング (32ビット) = 4
ZZ	オペランドサイズを表す指定コードです。 バイト (8ビット) = 00H ワード (16ビット) = 10H ロング (32ビット) = 20H

4. フラグ欄 (SZHVNC)

–	フラグは変化しません。。
*	命令の実行によって、フラグは変化します。
0	フラグは “0” にクリアされます。
1	フラグは “1” にセットされます。
P	命令の実行によって、フラグは変化します。(パリティフラグとして働きます)
V	命令の実行によって、フラグは変化します。(オーバフローフラグとして働きます)
X	フラグには、不定値がセットされます。

5. 命令長欄

命令長を、バイト単位で示しています。

+#	イミディエートデータ長を加えてください。
+M	アドレッシングコード長を加えてください。
+#M	イミディエートデータ長 + アドレッシングコード長を加えてください。

6. ステート欄

命令の実行処理時間を、ステート単位で、8ビット, 16ビット, 32ビット処理の場合を順に示しています。

1ステート = 100 ns @ 20 MHz発振時
1ステート = 80 ns @ 25 MHz発振時

■ 900/H命令一覧表 (1/10)

(1) 転送

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
LD	BWL	LD R,r	C8+zz+r :88+R	R ← r	------	2	2. 2. 2
	BWL	LD r,R	C8+zz+r :98+R	r ← R	------	2	2. 2. 2
	BWL	LD r,#3	C8+zz+r :A8+#3	r ← #3	------	2	2. 2. 2
	BWL	LD R,#	20+zz+R :#	R ← #	------	1+#	2. 3. 5
	BWL	LD r,#	C8+zz+r :03:#	r ← #	------	2+#	3. 4. 6
	BWL	LD R,(mem)	80+zz+mem:20+R	R ← (mem)	------	2+M	4. 4. 6
	BWL	LD (mem),R	B0+mem :40+zz+R	(mem) ← R	------	2+M	4. 4. 6
	BW-	LD<W> (#8) ,#	08+z :#8:#	(#8) ← #	------	2+#	5. 6. -
	BW-	LD<W> (mem),#	B0+mem :00+z:#	(mem) ← #	------	2+M#	5. 6. -
	BW-	LD<W> (#16),(mem)	80+zz+mem:19:#16	(#16) ← (mem)	------	4+M	8. 8. -
	BW-	LD<W> (mem),(#16)	B0+mem :14+z:#16	(mem) ← (#16)	------	4+M	8. 8. -
PUSH	B--	PUSH F	18	(-XSP) ← F	------	1	3. -. -
	B--	PUSH A	14	(-XSP) ← A	------	1	3. -. -
	-WL	PUSH R	18+zz+R	(-XSP) ← R	------	1	-. 3. 5
	BWL	PUSH r	C8+zz+r :04	(-XSP) ← r	------	2	4. 4. 6
	BW-	PUSH<W> #	09+z :#	(-XSP) ← #	------	1+#	4. 5. -
	BW-	PUSH<W> (mem)	80+zz+mem:04	(-XSP) ← (mem)	------	2+M	6. 6. -
POP	B--	POP F	19	F ← (XSP+)	******	1	4. -. -
	B--	POP A	15	A ← (XSP+)	------	1	4. -. -
	-WL	POP R	38+zz+R	R ← (XSP+)	------	1	-. 4. 6
	BWL	POP r	C8+zz+r :05	r ← (XSP+)	------	2	5. 5. 7
	BW-	POP<W> (mem)	B0+mem :04+z	(mem) ← (XSP+)	------	2+M	7. 7. -
LDA	-WL	LDA R,mem	B0+mem :10+zz+R	R ← mem	------	2+M	-. 4. 4
LDAR	-WL	LDAR R,$+4+d16	F3:13:d16:20+zz+R	R ← PC+d16	------	5	-. 7. 7

(2) 交換

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
EX	B--	EX F,F'	16	F ↔ F'	******	1	2. -. -
	BW-	EX R,r	C8+zz+r :B8+R	R ↔ r	------	2	3. 3. -
	BW-	EX (mem),R	80+zz+mem:30+R	(mem) ↔ R	------	2+M	6. 6. -
MIRR	-W-	MIRR r	D8+r :16	r<0:MSB>←r<MSB:0>	------	2	-. 3. -

■ 900/H命令一覧表 (2/10)

(3) ブロック転送/ブロックサーチ

命令群	サイズ	ニモニック	コード (16進)	機　能	SZHVNC	命令長	ステート
LDxx	BW-	LDI<W> [(XDE+),(XHL+)]	83+zz :10	(XDE+) ← (XHL+) BC ← BC-1	--0①0-	2	8. 8. -
	BW-	LDI<W> (XIX+),(XIY+)	85+zz :10	(XIX+) ← (XIY+) BC ← BC-1	--0①0-	2	8. 8. -
	BW-	LDIR<W> [(XDE+),(XHL+)]	83+zz :11	repeat (XDE+) ← (XHL+) BC ← BC-1 until BC=0	--000-	2	7n+1
	BW-	LDIR<W> (XIX+),(XIY+)	85+zz :11	repeat (XIX+) ← (XIY+) BC ← BC-1 until BC=0	--000-	2	7n+1
	BW-	LDD<W> [(XDE-),(XHL-)]	83+zz :12	(XDE-) ← (XHL-) BC ← BC-1	--0①0-	2	8. 8. -
	BW-	LDD<W> (XIX-),(XIY-)	85+zz :12	(XIX-) ← (XIY-) BC ← BC-1	--0①0-	2	8. 8. -
	BW-	LDDR<W> [(XDE-),(XHL-)]	83+zz :13	repeat (XDE-) ← (XHL-) BC ← BC-1 until BC=0	--000-	2	7n+1
	BW-	LDDR<W> (XIX-),(XIY-)	85+zz :13	repeat (XIX-) ← (XIY-) BC ← BC-1 until BC=0	--000-	2	7n+1
CPxx	BW-	CPI [A/WA,(R+)]	80+zz+R :14	A/WA - (R+) BC ← BC-1	*②*①1-	2	6. 6. -
	BW-	CPIR [A/WA,(R+)]	80+zz+R :15	repeat A/WA - (R+) BC ← BC-1 until A/WA=(R) or BC=0	*②*①1-	2	6n+1
	BW-	CPD [A/WA,(R-)]	80+zz+R :16	A/WA - (R-) BC ← BC-1	*②*①1-	2	6. 6. -
	BW-	CPDR [A/WA,(R-)]	80+zz+R :17	repeat A/WA - (R-) BC ← BC-1 until A/WA=(R) or BC=0	*②*①1-	2	6n+1

(注1) ①： 命令実行後BC＝0ならP/Vフラグは“0”に、それ以外は“1”にセットされます。
②： A/WA＝(R)ならZフラグは“1”に、それ以外は“0”にセットされます。

(注2) CPI, CPIR, CPD, CPDR命令でオペランドを省略したときは、“A, (XHL+/−)”を指定したことになります。

■ 900/H命令一覧表 (3/10)

(4) 算術演算

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
ADD	BWL	ADD R,r	C8+zz+r :80+R	R ← R + r	***V0*	2	2. 2. 2
	BWL	ADD r,#	C8+zz+r :C8:#	r ← r + #	***V0*	2+#	3. 4. 6
	BWL	ADD R,(mem)	80+zz+mem:80+R	R ← R + (mem)	***V0*	2+M	4. 4. 6
	BWL	ADD (mem),R	80+zz+mem:88+R	(mem) ← (mem) + R	***V0*	2+M	6. 6. 10
	BW-	ADD<W> (mem),#	80+zz+mem:38:#	(mem) ← (mem) + #	***V0*	2+M#	7. 8. -
ADC	BWL	ADC R,r	C8+zz+r :90+R	R ← R + r + CY	***V0*	2	2. 2. 2
	BWL	ADC r,#	C8+zz+r :C9:#	r ← r + # + CY	***V0*	2+#	3. 4. 6
	BWL	ADC R,(mem)	80+zz+mem:90+R	R ← R+(mem)+CY	***V0*	2+M	4. 4. 6
	BWL	ADC (mem),R	80+zz+mem:98+R	(mem) ←(mem)+R+CY	***V0*	2+M	6. 6. 10
	BW-	ADC<W> (mem),#	80+zz+mem:39:#	(mem) ←(mem)+#+CY	***V0*	2+M#	7. 8. -
SUB	BWL	SUB R,r	C8+zz+r :A0+R	R ← R - r	***V1*	2	2. 2. 2
	BWL	SUB r,#	C8+zz+r :CA:#	r ← r - #	***V1*	2+#	3. 4. 6
	BWL	SUB R,(mem)	80+zz+mem:A0+R	R ← R - (mem)	***V1*	2+M	4. 4. 6
	BWL	SUB (mem),R	80+zz+mem:A8+R	(mem) ← (mem) - R	***V1*	2+M	6. 6. 10
	BW-	SUB<W> (mem),#	80+zz+mem:3A:#	(mem) ← (mem) - #	***V1*	2+M#	7. 8. -
SBC	BWL	SBC R,r	C8+zz+r :B0+R	R ← R - r - CY	***V1*	2	2. 2. 2
	BWL	SBC r,#	C8+zz+r :CB:#	r ← r - # - CY	***V1*	2+#	3. 4. 6
	BWL	SBC R,(mem)	80+zz+mem:B0+R	R ←R-(mem)-CY	***V1*	2+M	4. 4. 6
	BWL	SBC (mem),R	80+zz+mem:B8+R	(mem) ←(mem)-R-CY	***V1*	2+M	6. 6. 10
	BW-	SBC<W> (mem),#	80+zz+mem:3B:#	(mem) ←(mem)-#-CY	***V1*	2+M#	7. 8. -
CP	BWL	CP R,r	C8+zz+r :F0+R	R - r	***V1*	2	2. 2. 2
	BW-	CP r,#3	C8+zz+r :D8+#3	r - #3	***V1*	2	2. 2. -
	BWL	CP r,#	C8+zz+r :CF:#	r - #	***V1*	2+#	3. 4. 6
	BWL	CP R,(mem)	80+zz+mem:F0+R	R - (mem)	***V1*	2+M	4. 4. 6
	BWL	CP (mem),R	80+zz+mem:F8+R	(mem) - R	***V1*	2+M	4. 4. 6
	BW-	CP<W> (mem),#	80+zz+mem:3F:#	(mem) - #	***V1*	2+M#	5. 6. -
INC	B--	INC #3,r	C8+r :60+#3	r ← r + #3	***V0-	2	2. -. -
	-WL	INC #3,r	C8+zz+r :60+#3	r ← r + #3	------	2	-. 2. 2
	BW-	INC<W> #3,(mem)	80+zz+mem:60+#3	(mem) ← (mem) + #3	***V0-	2+M	6. 6. -
DEC	B--	DEC #3,r	C8+r :68+#3	r ← r - #3	***V1-	2	2. -. -
	-WL	DEC #3,r	C8+zz+r :68+#3	r ← r - #3	------	2	-. 2. 2
	BW-	DEC<W> #3,(mem)	80+zz+mem:68+#3	(mem) ← (mem) - #3	***V1-	2+M	6. 6. -
NEG	BW-	NEG r	C8+zz+r :07	r ← 0 - r	***V1*	2	2. 2. -
EXTZ	-WL	EXTZ r	C8+zz+r :12	r<high> ← 0	------	2	-. 3. 3
EXTS	-WL	EXTS r	C8+zz+r :13	r<high> ← r<low. MSB>	------	2	-. 3. 3
DAA	B--	DAA r	C8+r :10	加減算後の10進補正	***P-*	2	4. -. -
PAA	-WL	PAA r	C8+zz+r :14	if r<0>=1 then INC r	------	2	-. 4. 4

(注1) INC/DEC命令では、#3のコード値が0のときは、+8/−8として機能します。
(注2) TLCS-90の “ADD R,r” 命令 (ワード型) は、S/Z/Vフラグが変化しません。

■ 900/H命令一覧表 (4/10)

命令群	サイズ	ニモニック	コード (16進)	機　能	SZHVNC	命令長	ステート

MUL	BW-
	BW-
	BW-
		MUL RR,r
		MUL rr,#
		MUL RR,(mem)
			C8+zz+r :40+R
			C8+zz+r :08:#
			80+zz+mem:40+R
				RR ← R×r
				rr ← r×#
				RR ← R×(mem)
					------ 	
					------ 	
					------ 	
						2
						2+#
						2+M
							11.14. -
							12.15. -
							13.16. -

MULS	BW-
	BW-
	BW-
		MULS RR,r
		MULS rr,#
		MULS RR,(mem)
			C8+zz+r :48+R
			C8+zz+r :09:#
			80+zz+mem:48+R
				RR ← R×r ;signed
				rr ← r×# ;signed
				RR ←R×(mem);signed
					------ 	
					------ 	
					------ 	
						2
						2+#
						2+M
							9.12. -
							10.13. -
							11.14. -

DIV	BW- BW- BW-	DIV RR,r DIV rr,# DIV RR,(mem)	C8+zz+r :50+R C8+zz+r :0A:# 80+zz+mem:50+R	R ← RR÷r r ← rr÷# R ← RR÷(mem)	---V-- ---V-- ---V--	2 2+# 2+M	15.23. - 15.23. - 16.24. -
DIVS	BW- BW- BW-	DIVS RR,r DIVS rr,# DIVS RR,(mem)	C8+zz+r :58+R C8+zz+r :0B:# 80+zz+mem:58+R	R ← RR÷r ;signed r ← rr÷# ;signed R ← RR÷(mem);signed	---V-- ---V-- ---V--	2 2+# 2+M	18.26. - 18.26. - 19.27. -
MULA	-W-	MULA rr	D8+r :19	符号付き積和演算 rr← rr+(XDE)×(XHL) 32bit 32bit 16bit 16bit XHL ← XHL-2	**-V--	2	-.19. -
MINC	-W-	MINC1 #,r (#=2**n) (1<=n<=15)	D8+r :38:#-1	モジュロインクリメント ;+1 if (r mod #)=(#-1) then r←r-(#-1) else r←r+1	------	4	-. 5. -
	-W-	MINC2 #,r (#=2**n) (2<=n<=15)	D8+r :39:#-2	モジュロインクリメント ;+2 if (r mod #)=(#-2) then r←r-(#-2) else r←r+2	------	4	-. 5. -
	-W-	MINC4 #,r (#=2**n) (3<=n<=15)	D8+r :3A:#-4	モジュロインクリメント ;+4 if (r mod #)=(#-4) then r←r-(#-4) else r←r+4	------	4	-. 5. -
MDEC	-W-	MDEC1 #,r (#=2**n) (1<=n<=15)	D8+r :3C:#-1	モジュロデクリメント ;-1 if (r mod #)=0 then r←r+(#-1) else r←r-1	------	4	-. 4. -
	-W-	MDEC2 #,r (#=2**n) (2<=n<=15)	D8+r :3D:#-2	モジュロデクリメント ;-2 if (r mod #)=0 then r←r+(#-2) else r←r-2	------	4	-. 4. -
	-W-	MDEC4 #,r (#=2**n) (3<=n<=15)	D8+r :3E:#-4	モジュロデクリメント ;-4 if (r mod #)=0 then r←r+(#-4) else r←r-4	------	4	-. 4. -

(注) MUL, MULS, DIV, DIVS 命令のオペランド “RR” は、演算サイズの倍のレジスタを指定することを示しています。演算サイズがバイトのとき (8×8ビット, 16÷8ビット) は、ワードレジスタ (16ビット) 指定し、演算サイズがワードのとき (16×16ビット, 32÷16ビット) は、ロングワードレジスタ (32ビット) を指定します。

■ 900/H命令一覧表 (5/10)

(5) 論理演算

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
AND	BWL	AND R,r	C8+zz+r :C0+R	R ← R and r	**1P00	2	2. 2. 2
	BWL	AND r,#	C8+zz+r :CC:#	r ← r and #	**1P00	2+#	3. 4. 6
	BWL	AND R,(mem)	80+zz+mem:C0+R	R ← R and (mem)	**1P00	2+M	4. 4. 6
	BWL	AND (mem),R	80+zz+mem:C8+R	(mem) ← (mem) and R	**1P00	2+M	6. 6. 10
	BW-	AND<w> (mem),#	80+zz+mem:3C:#	(mem) ← (mem) and #	**1P00	2+M#	7. 8. -
OR	BWL	OR R,r	C8+zz+r :E0+R	R ← R or r	**0P00	2	2. 2. 2
	BWL	OR r,#	C8+zz+r :CE:#	r ← r or #	**0P00	2+#	3. 4. 6
	BWL	OR R,(mem)	80+zz+mem:E0+R	R ← R or (mem)	**0P00	2+M	4. 4. 6
	BWL	OR (mem),R	80+zz+mem:E8+R	(mem) ← (mem) or R	**0P00	2+M	6. 6. 10
	BW-	OR<W> (mem),#	80+zz+mem:3E:#	(mem) ← (mem) or #	**0P00	2+M#	7. 8. -
XOR	BWL	XOR R,r	C8+zz+r :D0+R	R ← R xor r	**0P00	2	2. 2. 2
	BWL	XOR r,#	C8+zz+r :CD:#	r ← r xor #	**0P00	2+#	3. 4. 6
	BWL	XOR R,(mem)	80+zz+mem:D0+R	R ← R xor (mem)	**0P00	2+M	4. 4. 6
	BWL	XOR (mem),R	80+zz+mem:D8+R	(mem) ← (mem) xor R	**0P00	2+M	6. 6. 10
	BW-	XOR<W> (mem),#	80+zz+mem:3D:#	(mem) ← (mem) xor #	**0P00	2+M#	7. 8. -
CPL	BW-	CPL r	C8+zz+r :06	r ← not r	--1-1-	2	2. 2. -

■ 900/H命令一覧表 (6/10)

(6) ビット操作

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
LDCF	BW-	LDCF #4,r	C8+zz+r :23:#4	CY ← r<#4>	-----*	3	3.3.-
	BW-	LDCF A ,r	C8+zz+r :2B	CY ← r<A>	-----*	2	3.3.-
	B--	LDCF #3,(mem)	B0+men :98+#3	CY ← (mem)<#3>	-----*	2+M	6.-.-
	B--	LDCF A ,(mem)	B0+men :2B	CY ← (mem)<A>	-----*	2+M	6.-.-
STCF	BW-	STCF #4,r	C8+zz+r :24:#4	r<#4> ← CY	------	3	3.3.-
	BW-	STCF A ,r	C8+zz+r :2C	r<A> ← CY	------	2	3.3.-
	B--	STCF #3,(mem)	B0+mem :A0+#3	(mem)<#3> ← CY	------	2+M	7.-.-
	B--	STCF A ,(mem)	B0+mem :2C	(mem)<A> ← CY	------	2+M	7.-.-
ANDCF	BW-	ANDCF #4,r	C8+zz+r :20:#4	CY ← CY and r<#4>	-----*	3	3.3.-
	BW-	ANDCF A ,r	C8+zz+r :28	CY ← CY and r<A>	-----*	2	3.3.-
	B--	ANDCF #3,(mem)	B0+mem :80+#3	CY ← CY and (mem)<#3>	-----*	2+M	6.-.-
	B--	ANDCF A ,(mem)	B0+mem :28	CY ← CY and (mem)<A>	-----*	2+M	6.-.-
ORCF	BW-	ORCF #4,r	C8+zz+r :21:#4	CY ← CY or r<#4>	-----*	3	3.3.-
	BW-	ORCF A ,r	C8+zz+r :29	CY ← CY or r<A>	-----*	2	3.3.-
	B--	ORCF #3,(mem)	B0+mem :88+#3	CY ← CY or (mem)<#3>	-----*	2+M	6.-.-
	B--	ORCF A ,(mem)	B0+mem :29	CY ← CY or (mem)<A>	-----*	2+M	6.-.-
XORCF	BW-	XORCF #4,r	C8+zz+r :22:#4	CY ← CY xor r<#4>	-----*	3	3.3.-
	BW-	XORCF A ,r	C8+zz+r :2A	CY ← CY xor r<A>	-----*	2	3.3.-
	B--	XORCF #3,(mem)	B0+mem :90+#3	CY ← CY xor (mem)<#3>	-----*	2+M	6.-.-
	B--	XORCF A ,(mem)	B0+mem :2A	CY ← CY xor (mem)<A>	-----*	2+M	6.-.-
RCF	---	RCF	10	CY ← 0	--0-00	1	2
SCF	---	SCF	11	CY ← 1	--0-01	1	2
CCF	---	CCF	12	CY ← not CY	--X-0*	1	2
ZCF	---	ZCF	13	CY ← not "Z" フラグ	--X-0*	1	2
BIT	BW-	BIT #4,r	C8+zz+r :33:#4	Z ← not r<#4>	X*1X0-	3	3.3.-
	B--	BIT #3,(mem)	B0+mem :C8+#3	Z ← not (mem)<#3>	X*1X0-	2+M	6.-.-
RES	BW-	RES #4,r	C8+zz+r :30:#4	r<#4> ← 0	------	3	3.3.-
	B--	RES #3,(mem)	B0+mem :B0+#3	(mem)<#3> ← 0	------	2+M	7.-.-
SET	BW-	SET #4,r	C8+zz+r :31:#4	r<#4> ← 1	------	3	3.3.-
	B--	SET #3,(mem)	B0+mem :B8+#3	(mem)<#3> ← 1	------	2+M	7.-.-
CHG	BW-	CHG #4,r	C8+zz+r :32:#4	r<#4> ← not r<#4>	------	3	3.3.-
	B--	CHG #3,(mem)	B0+mem :C0+#3	(mem)<#3>←not (mem)<#3>	------	2+M	7.-.-
TSET	BW-	TSET #4,r	C8+zz+r :34:#4	Z←not r<#4> : r<#4>←1	X*1X0-	3	4.4.-
	B--	TSET #3,(mem)	B0+mem :A8+#3	Z ← not (mem)<#3> (mem)<#3> ← 1	X*1X0-	2+M	7.-.-
BS1	-W-	BS1F A,r	D8+r :0E	A ← 1 サーチ r ; Forward	---①--	2	-.3.-
	-W-	BS1B A,r	D8+r :0F	A ← 1 サーチ r ; Backward	---①--	2	-.3.-

(注) ①： サーチするビットが見つかったとき“0”にセットされ、それ以外のときは“1”にセットされAレジスタは不定値になります。

■ 900/H命令一覧表 (7/10)

(7) 特別演算、CPU制御

命令群	サイズ	ニモニック	コード(16進)	機 能	SZHVNC	命令長	ステート
NOP	---	NOP	00	no operation	------	1	2
EI	---	EI [#3]	06 :#3	割り込み許可フラグの設定 IFF←#3	------	2	3
DI	---	DI	06 :07	割り込み禁止 IFF←7	------	2	4
PUSH	-W-	PUSH SR	02	(-XSP)←SR	------	1	-.3.-
POP	-W-	POP SR	03	SR←(XSP+)	******	1	-.4.-
SWI	---	SWI [#3]	F8+#3	ソフトウエア割り込み PUSH PC&SR JP (FFFF00H+4×#3)	------	1	19
HALT	---	HALT	05	CPU停止	------	1	6
LDC	BWL BWL	LDC cr,r LDC r,cr	C8+zz+r :2E:cr C8+zz+r :2F:cr	cr ← r r ← cr	------ ------	3 3	3.3.3 3.3.3
LDX	B--	LDX (#8),#	F7:00:#8:00:#:00	(#8) ← #	------	6	8.-.-

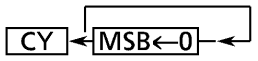
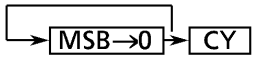
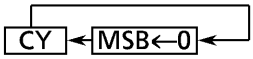
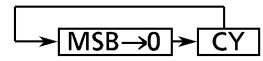
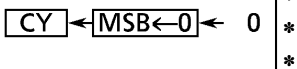
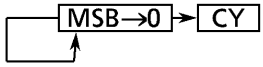
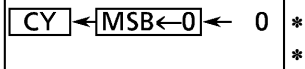
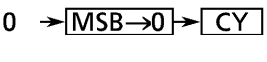
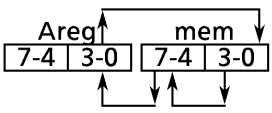
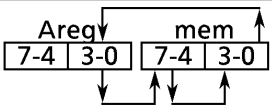
LINK	--L	LINK r,d16	E8+r :0C:d16	PUSH r
				LD r,XSP
				ADD XSP,d16
					------	4	-.-.8

UNLK	--L	UNLK r	E8+r :0D	LD XSP,r POP r	------	2	-.-.7
LDF	---	LDF #3	17 :#3	レジスタバンクの設定 RFP← #3 (リセット時"0")	------	2	2
INCF	---	INCF	0C	レジスタバンクの切り替え RFP← RFP + 1	------	1	2
DECF	---	DECF	0D	レジスタバンクの切り替え RFP← RFP − 1	------	1	2
SCC	BW-	SCC cc,r	C8+zz+r :70+cc	if cc then r ← 1 else r ← 0	------	2	2.2.-

(注1) EI命令でオペランド#3の記述を省略すると“0”を指定したものとみなされます。
(注2) SWI命令でオペランド#3の記述を省略すると“7”を指定したものとみなされます。

■ 900/H命令一覧表 (8/10)

(8) ローテート、シフト

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
RLC	BWL BWL BW-	RLC #4,r RLC A,r RLC<W> (mem)	C8+zz+r :E8:#4 C8+zz+r :F8 80+zz+mem:78		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
RRC	BWL BWL BW-	RRC #4,r RRC A,r RRC<W> (mem)	C8+zz+r :E9:#4 C8+zz+r :F9 80+zz+mem:79		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
RL	BWL BWL BW-	RL #4,r RL A,r RL<W> (mem)	C8+zz+r :EA:#4 C8+zz+r :FA 80+zz+mem:7A		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
RR	BWL BWL BW-	RR #4,r RR A,r RR<W> (mem)	C8+zz+r :EB:#4 C8+zz+r :FB 80+zz+mem:7B		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
SLA	BWL BWL BW-	SLA #4,r SLA A,r SLA<W> (mem)	C8+zz+r :EC:#4 C8+zz+r :FC 80+zz+mem:7C		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
SRA	BWL BWL BW-	SRA #4,r SRA A,r SRA<W> (mem)	C8+zz+r :ED:#4 C8+zz+r :FD 80+zz+mem:7D		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
SLL	BWL BWL BW-	SLL #4,r SLL A,r SLL<W> (mem)	C8+zz+r :EE:#4 C8+zz+r :FE 80+zz+mem:7E		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
SRL	BWL BWL BW-	SRL #4,r SRL A,r SRL<W> (mem)	C8+zz+r :EF:#4 C8+zz+r :FF 80+zz+mem:7F		**0P0* **0P0* **0P0*	3 2 2+M	3+n/4 3+n/4 6.6
RLD	B--	RLD [A,](mem)	80+mem :06		**0P0-	2+M	14.-.-
RRD	B--	RRD [A,](mem)	80+mem :07		**0P0-	2+M	14.-.-

(注1) #4/Aによるシフト回数の指定では、モジュロ16 (0~15) が使われます。コード0は、16回シフトになります。
(注2) TLCS-90のRLCA/RRCA/RLA/RRA/SLAA/SRAA/SLLA/SRLA命令は、S/Z/Vフラグが変化しません。
(注3) ステート数の計算で、少数点以下は切り上げます。

■ 900/H命令一覧表 (9/10)

(9) ジャンプ、コール、リターン

命令群	サイズ	ニモニック	コード (16進)	機 能	SZHVNC	命令長	ステート
JP	---	JP #16	1A :#16	PC ← #16	------	3	5
	---	JP #24	1B :#24	PC ← #24	------	4	6
	---	JR [cc,]$+2+d8	60+cc :d8	if cc then PC ← PC+d8	------	2	5/2 (T/F)
	---	JRL [cc,]$+3+d16	70+cc :d16	if cc then PC ← PC+d16	------	3	5/2 (T/F)
	---	JP [cc,]mem	B0+mem :D0+cc	if cc then PC ← mem	------	2+M	7/4 (T/F)
CALL	---	CALL #16	1C :#16	PUSH PC : JP #16	------	3	9
	---	CALL #24	1D :#24	PUSH PC : JP #24	------	4	10
	---	CALR $+3+d16	1E :d16	PUSH PC : JR $+3+d16	------	3	10
	---	CALL [cc,]mem	B0+mem :E0+cc	if cc then PUSH PC : JP mem	------	2+M	12/4 (T/F)
DJNZ	BW-	DJNZ [r,]$+3/4+d8	C8+zz+r :1C:d8	r←r-1(r省略でBレジスタ) if r≠0 then JR $+3+d8	------	3	6 (r≠0) 4 (r=0)
RET	---	RET	0E	POP PC	------	1	9
	---	RET cc	B0 :F0+cc	if cc then POP PC	------	2	12/4 (T/F)
	---	RETD d16	0F :d16	RET : ADD XSP,d16	------	3	11
	---	RETI	07	POP SR&PC	******	1	12

(注1) (T/F) はTrue/False時のステート数を表しています。

■ 900/H命令一覧表 (10/10)

(10) アドレッシングモード

分類	モード	ステート (追加分)
R	R	+0
r	r	+1
(mem)	(R)	+0
	(R + d8)	+1
	(#8)	+1
	(#16)	+2
	(#24)	+3
	(r)	+1
	(r + d16)	+3
	(r + r8)	+3
	(r + r16)	+3
	(− r)	+1
	(r +)	+1

(11) 割り込み

モード		動作	ステート
汎用割り込み処理		PUSH PC PUSH SR IFF ← 受け付けレベル + 1 INTNEST ← INTNEST + 1 JP (FFFF00H + ベクタ)	18
マイクロDMA	I/O to MEM	(DMADn +) ← (DMASn)	8. 8. 12
	I/O to MEM	(DMADn −) ← (DMASn)	8. 8. 12
	MEM to I/O	(DMADn) ← (DMASn +)	8. 8. 12
	MEM to I/O	(DMADn) ← (DMASn −)	8. 8. 12
	I/O to I/O	(DMADn) ← (DMASn)	8. 8. 12
	Counter	DMASn ← DMASn + 1	−. −. 5

(注) 割り込み動作の詳細は、各製品の「割り込み」の章をご覧ください。

付録C. 命令コードマップ (1/4)

1バイトオペコード命令

H/L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP		PUSH SR	POP SR		HALT	EI n	RETI	LD (n) , n	PUSH n	LDW (n) , nn	PUSHW nn	INCF	DECF	RET	RETD dd
1	RCF	SCF	CCF	ZCF	PUSH A	POP A	EX F, F'	LDF n	PUSH F	POP F	JP nn	JP nnn	CALL nn	CALL nnn	CALR PC + dd	
2	LD R, n								PUSH RR							
3	LD RR, nn								PUSH XRR							
4	LD XRR, nnnn								POP RR							
5									POP XRR							
6	F	LT	LE	ULE	PE/OV	M/MI	Z	JR C	cc,PC + d (T)	GE	GT	UGT	PO/NO V	P/PL	NZ	NC
7	JRL cc,PC + dd ↑															
8	scr. B (XWA)	(XBC)	(XDE)	(XHL)	(XIX)	(XIY)	(XIZ)	(XSP)	scr. B (XWA + d)	(XBC + d)	(XDE + d)	(XHL + d)	(XIX + d)	(XIY + d)	(XIZ + d)	(XSP + d)
9	scr. W ↑								scr. W ↑							
A	scr. L ↑								scr. L ↑							
B	dst ↑								dst ↑							
C	scr. B (n)	(nn)	(nnn)	(mem)	(– xrr)	(xrr +)		reg. B r	reg. B W	A	B	C	D	E	H	L
D	scr. W ↑							reg. W rr	reg. W WA	BC	DE	HL	IX	IY	IZ	SP
E	scr. L ↑							reg. L xrr	reg. L XWA	XBC	XDE	XHL	XIX	XIY	XIZ	XSP
F	dst ↑							LDX (n), n	0	1	2	SWI 3	n 4	5	6	7

(注1) 空白部分のコードは、未定義命令です。

(注2) コード 01H, 04Hにはダミーの命令が実装されています。使用しないでください。

付録C. 命令コードマップ (2/4)

1バイト目: reg

H/L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0				LD r,#	PUSH r	POP r	CPL BW r	NEG BW r	MUL rr,#	MULS rr,#	DIV rr,#	DIVS BW rr,#	LINK L r, dd	UNLK L r	BS1F A, r	BS1B W A, r
1	DAA B r		EXTZ WL r	EXTS WL r	PAA WL r		MIRR W r			MULA W r			DJNZ BW r, d			
2	ANDCF #, r	ORCF #, r	XORCF #, r	LDCF #, r	STCF BW #, r				ANDCF A, r	ORCF A, r	XORCF A, r	LDCF A, r	STCF BW A, r		LDC cr, r	LDC r, cr
3	RES #, r	SET #, r	CHG #, r	BIT #, r	TSET BW #, r				MINC1	MINC2 #, r	MINC4 W		MDEC1	MDEC2 #, r	MDEC4 W	
4				MUL	R, r			BW				MULS	R, r			BW
5				DIV	R, r			BW				DIVS	R, r			BW
6	8	1	2	INC 3	#3, r 4	5	6	7	8	1	2	DEC 3	#3, r 4	5	6	7
7	F	LT	LE	ULE	PE/OV	M/MI	Z	SCC C	cc. r (T)	GE	GT	UGT	PO/NOV	P/PL	NZ	BW NC
8				ADD	R, r							LD	R, r			
9				ADC	R, r							LD	r, R			
A				SUB	R, r				0	1	2	LD 3	r, #3 4	5	6	7
B				SBC	R, r							EX	R, r			BW
C				AND	R, r				ADD r, #	ADC r, #	SUB r, #	SBC r, #	AND r, #	XOR r, #	OR r, #	CP r, #
D				XOR	R, r				0	1	2	CP 3	r, #3 4	5	6	BW 7
E				OR	R, r				RLC #, r	RRC #, r	RL #, r	RR #, r	SLA #, r	SRA #, r	SLL #, r	SRL #, r
F				CP	R, r				RLC A, r	RRC A, r	RL A, r	RR A, r	SLA A, r	SRA A, r	SLL A, r	SRL A, r

r : 1バイト目のコードで指定されるレジスタを示します (すべてのCPU内部レジスタが指定可能です)。
R : 2バイト目のコードで指定されるレジスタを示します (カレントレジスタ8本のみ指定可能です)。
B : オペランドサイズは、バイト型です。
W : オペランドサイズは、ワード型です。
L : オペランドサイズは、ロング型です。

(注) コード1AH, 1BH, 3BH, 3FHにはダミーの命令が実装されています。使用しないでください。

付録C. 命令コードマップ (3/4)

1バイト目:src (mem)

H/L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0					PUSH BW (mem)		RLD A, (mem)	RRD B								
1	LDI	LDIR	LDD	LDDR BW	CPI	CPIR	CPD	CPDR BW		LD BW (nn), (m)						
2	LD R, (mem)															
3	EX (mem), R							BW	ADD (mem) ,#	ADC	SUB	SBC	AND	XOR	OR	CP BW
4	MUL R, (mem)							BW	MULS R, (mem)							BW
5	DIV R, (mem)							BW	DIVS R, (mem)							BW
6	INC #3, (mem)							BW	DEC #3, (mem)							BW
	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7
7									RLC (mem)	RRC	RL	RR	SLA	SRA	SLL	SRL BW
8	ADD R, (mem)								ADD (mem), R							
9	ADC R, (mem)								ADC (mem), R							
A	SUB R, (mem)								SUB (mem), R							
B	SBC R, (mem)								SBC (mem), R							
C	AND R, (mem)								AND (mem), R							
D	XOR R, (mem)								XOR (mem), R							
E	OR R, (mem)								OR (mem), R							
F	CP R, (mem)								CP (mem), R							

B : オペランドサイズは、バイト型です。
W : オペランドサイズは、ワード型です。

付録C. 命令コードマップ (4/4)

1バイト目: dst (mem)

H/L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	LD B (m), #		LD W (m), #		POP B (mem)		POP W (mem)									
1					LD B (m), (nn)		LD W (m), (nn)									
2	LDA R, mem W								ANDCF A, (mem)	ORCF A, (mem)	XORCF A, (mem)	LDCF A, (mem)	STCF A, (mem) B			
3	LDA R, mem L															
4	LD (mem), R B															
5	LD (mem), R W															
6	LD (mem), R L															
7																
8	ANDCF #3, (mem) B								ORCF #3, (mem) B							
	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
9	XORCF #3, (mem) B								LDCF #3, (mem) B							
	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
A	STCF #3, (mem) B								TSET #3, (mem) B							
	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
B	RES #3, (mem) B								SET #3, (mem) B							
	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
C	CHG #3, (mem) B								BIT #3, (mem) B							
	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
D	JP cc, mem															
	F	LT	LE	ULE	PE/OV	M/MI	Z	C	(T)	GE	GT	UGT	PO/NOV	P/PL	NZ	NC
E	CALL cc, mem ↑															
F	RET cc (1バイト目のコードはB0H) ↑															

B : オペランドサイズは、バイト型です。
W : オペランドサイズは、ワード型です。
L : オペランドサイズは、ロング型です。

付録 D. TLCS-90との相違点

項目＼シリーズ	TLCS-90	TLCS-900
CPU アーキテクチャ	8ビットCPU	16ビットCPU
内蔵 ROM / 内蔵 RAM	8ビットデータバス	16ビットデータバス
内蔵 I/O	8ビットデータバス	8ビットデータバス
外部データバス	8ビットデータバス	8ビット/16ビットデータバス (混在可能)
プログラム空間 (MMU付を除く)	64KB	16MB (リニア)
データ 空間	16MB (バンク)	16MB (リニア)
命令セット / 命令ニモニック	TLCS-90	TLCS-90 + α α = 16ビット乗除算命令, ビット操作命令の強化, 32ビット転送 / 演算命令, Cコンパイラ用命令, レジスタバンク操作命令 etc.
命令コード (オブジェクト コード)	TLCS-90 独自	TLCS-900 独自 (TLCS-90と違います。)
アドレッシングモード	TLCS-90	TLCS-90 + α α = (– Reg), (Reg +), (Reg + disp16), (Reg + Reg16), (nnn)
汎用レジスタ	TLCS-90	TLCS-90 + α α = 32ビット化 レジスタバンク化 システムスタックポインタの追加 [900_CPUのみ]
フラグ (F)	S Z I H X V N C	S Z "0" H "0" V N C Iフラグは, SRレジスタのIFF2～0に拡張。 Xフラグは, 削除。
リセット	PC ← 0000H (SPは不変)	PC ← (ベクタベース番地) XSP← 100H
内蔵 ROM アドレス	0000H～	不定
内蔵 RAM アドレス	～FFxxH	0080H～
内蔵 I/O アドレス	FFxxH～FFFFH	0000H～007FH
ダイレクトアドレッシング領域 (n)	FF00H～FFFFH	0000H～00FFH
割り込み		
割り込みスタートアドレス	0000H + 8 × V	ベクタベース番地 + 4 × V
セーブされるレジスタ	PC と AF	PC と SR
マスクレジスタ	IFF	IFF2～0
マスクレベル	0～1	0～7

項目＼シリーズ	TLCS-90	TLCS-900
命令		
① ADD R, r (ワード型)	S, Z, Vフラグは変化しません。 [16ビットのレジスタ間加算以外では、S, Z, Vフラグは変化します。]	S, Z, Vフラグは変化します。
② Aレジスタのシフト	「RLCA RRCA RLA RRA SLAA SRAA SLLA SRLA 」 命令では、S, Z, Vフラグは変化しません。 「RLC A RRC A RL A RR A SLA A SRA A SLL A SRL A」 命令では、S, Z, Vフラグは変化します。	S, Z, Vフラグは変化します。

補足： TLCS-900は、基本的にTLCS-90との互換性を考慮しつつ、TLCS-90のCPUを16ビット化したシリーズです。

内蔵されているI/Oは、TLCS-90と完全互換性がありますが、CPUについては、合計6種類の直接対応する命令がないので、TLCS-90用に作られたプログラムを移植するときは、下記のような等価命令に置き換えが必要です。

TLCS-90にあり TLCS-900にない命令	TLCS-900での等価命令
EXX	EX BC, BC' EX DE, DE' EX HL, HL'
EX AF, AF'	EX A, A' EX F, F'
PUSH AF	PUSH A PUSH F
POP AF	POP F POP A
INCX	(32ビット INC命令)
DECX	(32ビット DEC命令)

また、TLCS-900は、TLCS-90と同じ命令でも一部機能強化されている命令があり、オペランドでの指定項目が増えているものがあります。下記に、それらを示します。

TLCS-90	TLCS-900
INC reg INC mem	INC imm3, reg INC imm3, mem
DEC reg DEC mem	DEC imm3, reg DEC imm3, mem
RLC reg RRC reg RL reg RR reg SLA reg SRA reg SLL reg SRL reg	RLC imm, reg RRC imm, reg RL imm, reg RR imm, reg SLA imm, reg SRA imm, reg SLL imm, reg SRL imm, reg