付録 A. 命令の詳細

■ 命令リスト

① 転送

LD PUSH POP LDA LDAR

② 交換

EX MIRR

③ ブロック転送/ブロックサーチ

LDI LDIR LDD LDDR CPI CPIR CPD CPDR

④ 算術演算

ADD ADC SUB SBC CP INC DEC NEG
EXTZ EXTS DAA PAA MUL MULS DIV DIVS
MULA MINC MDEC

⑤ 論理演算

AND OR XOR CPL

⑥ ビット操作

LDCF STCF ANDCF ORCF XORCF RCF SCF CCF
ZCF BIT RES SET CHG TSET BS1

⑦ 特別演算, CPU制御

NOP EI DI PUSH_SR POP_SR SWI
HALT LDC LDX LINK UNLK LDF INCF DECF
SCC

⑧ ローテート, シフト

RLC RRC RL RR SLA SRA SLL SRL
RLD RRD

⑨ ジャンプ, コール, リターン

JP JR JRL CALL CALR DJNZ RET RETD
RETI

■ 本文中の記号の説明

本文中で使われている各記号の意味は、下記のとおりです。

dst	デスティネーション。データの転送先または演算結果の格納先を示します。
src	ソース。データの転送元または演算データの読み出し元を示します。
num	ナンバー。数値であることを示します。
condition	コンディション。フラグによる条件記号を示します。
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の8本のみ 16ビットの場合: WA, BC, DE, HL, IX, IY, IZ, SPの8本のみ 32ビットの場合: XWA, XBC, XDE, XHL, XIX, XIY, XIZ, XSPの8本のみ
r	8/16/32ビットの汎用レジスタ (CPU900H-45, 46の「レジスタマップ」で示されたすべてのレジスタ)
r16	16ビットの汎用レジスタ(CPU900H-45, 46の「レジスタマップ」で示されたすべての16ビットレジスタ)
r32	32ビットの汎用レジスタ(CPU900H-45, 46の「レジスタマップ」で示されたすべての32ビットレジスタ)
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(ミニマムモード＝16ビット,マキシマムモード＝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 or1~8 …… 短縮コード用
#4	4 ビットの即値データ; 0~15or1~16
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
CY	キャリーフラグ
Z	ゼロフラグ
(#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)
(-r32)	プリデクリメントアドレッシング
(r32+)	ポストインクリメントアドレッシング
$	その命令が置かれている先頭アドレス

■ オブジェクトコード中の記号の説明

オブジェクトコードの中で使われている各記号の意味は、下記のとおりです。

z
zz
zzz
s

オペランドサイズ指定コード

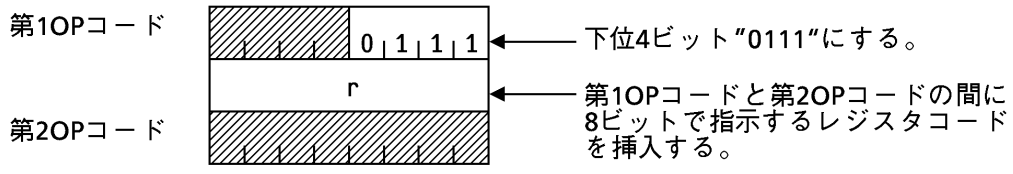
	バイト	ワード	ロング
z	0	1	—
zz	00	01	10
zzz	010	011	100
s	—	0	1

R
r

レジスタ指定コード

コード	バイト	ワード	ロング
000	W	WA	XWA
001	A	BC	XBC
010	B	DE	XDE
011	C	HL	XHL
100	D	IX	XIX
101	E	IY	XIY
110	H	IZ	XIZ
111	L	SP	XSP

補足: rで示されるレジスタは、上記以外に、拡張コードを使うと、すべてのレジスタを指定することができます。なお、この場合、実行ステート数は+1増加します。下記に、そのフォーマットを示します。



なお、"r"のコード値は、

ワードレジスタとしてアクセスするときは2の倍数

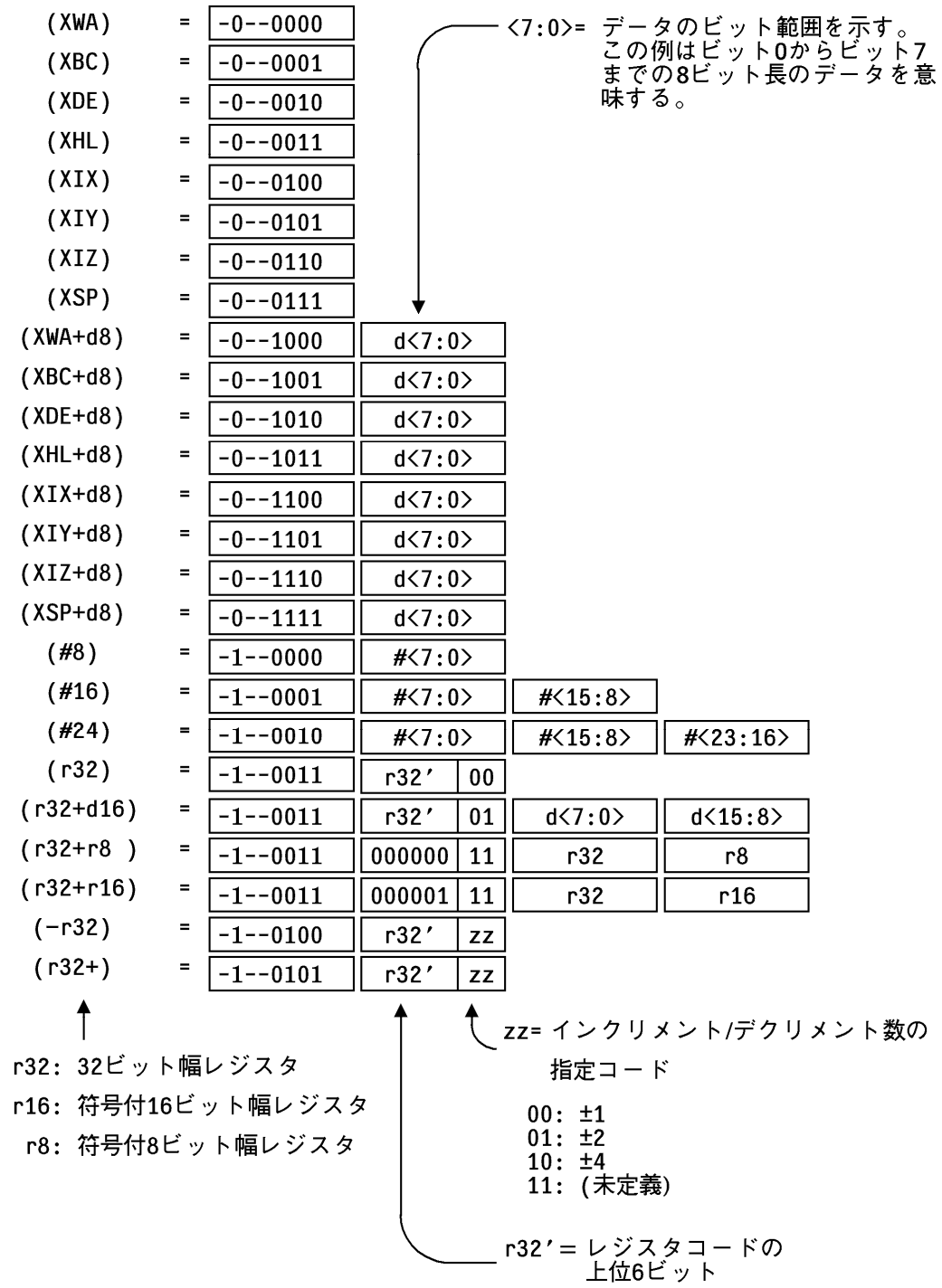
ロングレジスタとしてアクセスするときは4の倍数

の値に限られます。

8ビットで指示されるレジスタは、後述の"レジスタマップ"を参照してください。

mem

メモリアドレッシングモード指定コード



cc

コンディションコード

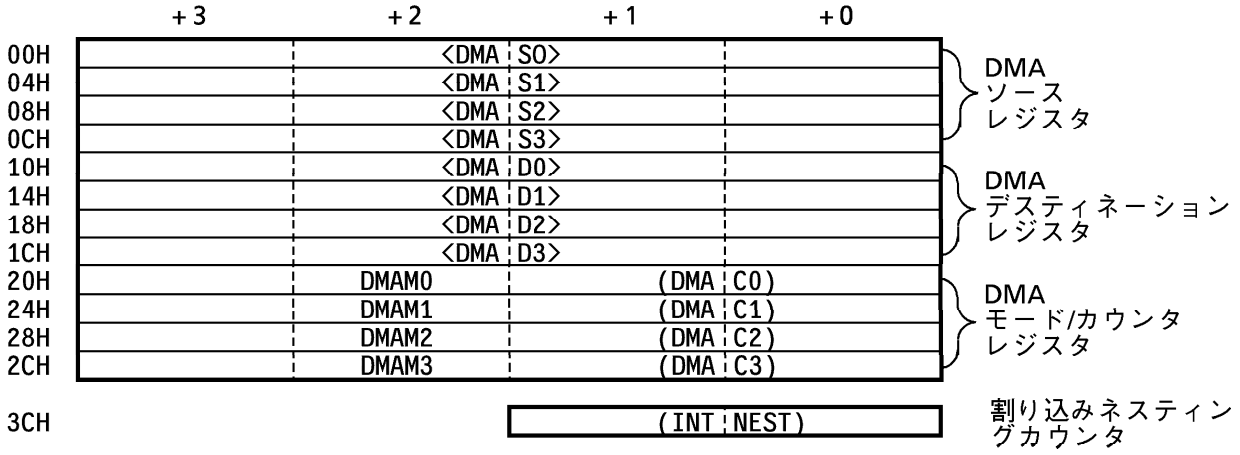
コード	記号	意味	条件式
0000	F	always False	-
1000	(なし)	always True	-
0110	Z	Zero	Z=1
1110	NZ	Not Zero	Z=0
0111	C	Carry	C=1
1111	NC	Not Carry	C=0
1101	PL or P	PLus	S=0
0101	MI or M	MInus	S=1
1110	NE	Not Equal	Z=0
0110	EQ	EQual	Z=1
0100	OV	OVerflow	P/V=1
1100	NOV	No OVerflow	P/V=0
0100	PE	Parity is Even	P/V=1
1100	PO	Parity is Odd	P/V=0
1001	GE	Greater than or Equal (signed)	(S xor P/V) =0
0001	LT	Less Than (signed)	(S xor P/V) =1
1010	GT	Greater Than (signed)	[Z or (S xor P/V)]=0
0010	LE	Less than or Equal (signed)	[Z or (S xor P/V)]=1
1111	UGE	Unsigned Greater than or Equal	C=0
0111	ULT	Unsigned Less Than	C=1
1011	UGT	Unsigned Greater Than	(C or Z) =0
0011	ULE	Unsigned Less than or Equal	(C or Z) =1

■レジスタマップ "r"

	+3	+2	+1	+0	
00H	QW0 (QWA 0)	QA0 <XWA 0>	RW0 (RWA 0)	RA0	バンク0
04H	QB0 (QBC 0)	QC0 <XBC 0>	RB0 (RBC 0)	RC0	
08H	QD0 (QDE 0)	QE0 <XDE 0>	RD0 (RDE 0)	RE0	
0CH	QH0 (QHL 0)	QL0 <XHL 0>	RH0 (RHL 0)	RL0	
10H	QW1 (QWA 1)	QA1 <XWA 1>	RW1 (RWA 1)	RA1	バンク1
14H	QB1 (QBC 1)	QC1 <XBC 1>	RB1 (RBC 1)	RC1	
18H	QD1 (QDE 1)	QE1 <XDE 1>	RD1 (RDE 1)	RE1	
1CH	QH1 (QHL 1)	QL1 <XHL 1>	RH1 (RHL 1)	RL1	
20H	QW2 (QWA 2)	QA2 <XWA 2>	RW2 (RWA 2)	RA2	バンク2
24H	QB2 (QBC 2)	QC2 <XBC 2>	RB2 (RBC 2)	RC2	
28H	QD2 (QDE 2)	QE2 <XDE 2>	RD2 (RDE 2)	RE2	
2CH	QH2 (QHL 2)	QL2 <XHL 2>	RH2 (RHL 2)	RL2	
30H	QW3 (QWA 3)	QA3 <XWA 3>	RW3 (RWA 3)	RA3	バンク3
34H	QB3 (QBC 3)	QC3 <XBC 3>	RB3 (RBC 3)	RC3	
38H	QD3 (QDE 3)	QE3 <XDE 3>	RD3 (RDE 3)	RE3	
3CH	QH3 (QHL 3)	QL3 <XHL 3>	RH3 (RHL 3)	RL3	
40H					
44H					
48H					
4CH					
50H					
54H					
58H					
5CH					
60H					
64H					
68H					
6CH					
70H					
74H					
78H					
7CH					
D0H	QW' (Q WA')	QA' <X WA'>	W' (W A')	A'	プレビアスバンク
D4H	QB' (Q BC')	QC' <X BC'>	B' (B C')	C'	
D8H	QD' (Q DE')	QE' <X DE'>	D' (D E')	E'	
DCH	QH' (Q HL')	QL' <X HL'>	H' (H L')	L'	
E0H	QW (Q WA)	QA <X WA >	W (W A)	A	カレントバンク
E4H	QB (Q BC)	QC <X BC >	B (B C)	C	
E8H	QD (Q DE)	QE <X DE >	D (D E)	E	
ECH	QH (Q HL)	QL <X HL >	H (H L)	L	
F0H	QIXH (Q IX)	QIXL <X IX>	IXH (I X)	IXL	
F4H	QIYH (Q IY)	QIYL <X IY>	IYH (I Y)	IYL	
F8H	QIZH (Q IZ)	QIZL <X IZ>	IZH (I Z)	IZL	
FCH	QSPH (Q SP)	QSPL <X SP>	SPH (S P)	SPL	

() : ワードレジスタ (16ビット)名
< > : ロングワードレジスタ(32ビット)名

■コントロールレジスタマップ “cr”



() : ワードレジスタ (16ビット)名
< > : ロングワードレジスタ(32ビット)名

ADC dst, src

< Add with Carry キャリー付き加算 >

動 作 : dst←dst+src+CY

説 明 : dstの内容とsrcの内容とキャリーフラグCYの内容が加算され、dstへ転送されます。

詳 細 :

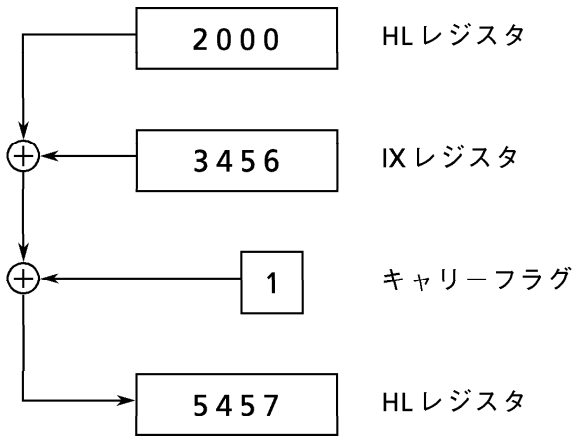
サイズ バイト	ワード	ロング	ニモニック		コード
○	○	○	ADC	R, r	11zz1r 10010R
○	○	○	ADC	r, #	11zz1r 11001001 #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	ADC	R, (mem)	1mzzmmmm 10010R
○	○	○	ADC	(mem), R	1mzzmmmm 10011R
○	○	×	ADC<W>	(mem), #	1m0zmmmm 00111001 #<7:0> #<15:8>

フラグ ：

S	Z	H	V	N	C
*	*	*	*	0	*

S ＝ 演算結果の最上位ビットの値がセットされます。
Z ＝ 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H ＝ 演算結果、ビット3からビット4へキャリーが発生したときは“1”、それ以外のときは“0”がセットされます。ただし、オペランド長が32ビットの場合は、不定値がセットされます。
V ＝ 演算結果、オーバフローが発生したときは“1”、それ以外のときは“0”がセットされます。
N ＝ “0”にクリアされます。
C ＝ 演算結果、最上位ビットからキャリーが発生したときは“1”、それ以外のときは“0”がセットされます。

実行例 ： HLレジスタが2000H, IXレジスタが3456H, キャリーフラグCYが1のとき、
ADC HL, IX
を実行すると、HLレジスタは5457Hになります。



ADD dst, src

< Add 加算 >

動　作 : dst←dst＋src

説　明 : dstの内容とsrcの内容が加算され、dstへ転送されます。

詳　細 :

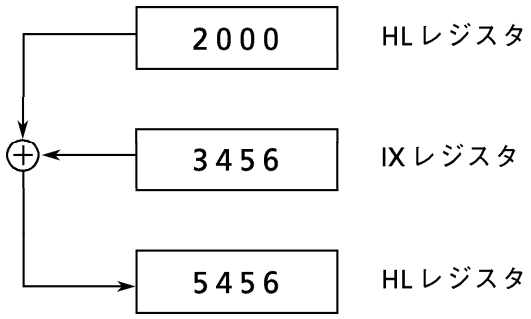
サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	○	ADD	R, r	1 1 z z 1 r 1 0 0 0 0 R
○	○	○	ADD	r, #	1 1 z z 1 r 1 1 0 0 1 0 0 0 #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	ADD	R, (mem)	1 m z z m m m m 1 0 0 0 0 R
○	○	○	ADD	(mem), R	1 m z z m m m m 1 0 0 0 1 R
○	○	×	ADD<W>	(mem), #	1 m 0 z m m m m 0 0 1 1 1 0 0 0 #<7:0> #<15:8>

フラグ：

S	Z	H	V	N	C
*	*	*	*	0	*

S ＝ 演算結果の最上位ビットの値がセットされます。
Z ＝ 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H ＝ 演算結果、ビット3からビット4へキャリーが発生したときは“1”、それ以外のときは“0”がセットされます。ただし、オペランド長が32ビットの場合は、不定値がセットされます。
V ＝ 演算結果、オーバフローが発生したときは“1”、それ以外のときは“0”がセットされます。
N ＝ “0”にクリアされます。
C ＝ 演算結果、最上位ビットからキャリーが発生したときは“1”、それ以外のときは“0”がセットされます。

実行例 ： HLレジスタが2000H, IXレジスタが3456Hのとき、
ADD HL, IX
を実行すると、HLレジスタは5456Hになります。



AND dst, src

< And 論理積 >

動 作 : dst←dst AND src

説 明 : dstの内容とsrcの内容が論理積演算され、dstへ転送されます。

(真理値表)

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

詳 細 :

サイズ			ニモニック		コード
バイト	ワード	ロング			
○	○	○	AND	R, r	1 1 z z 1 r 1 1 0 0 0 R
○	○	○	AND	r, #	1 1 z z 1 r 1 1 0 0 1 1 0 0 #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	AND	R, (mem)	1 m z z m m m m 1 1 0 0 0 R
○	○	○	AND	(mem), R	1 m z z m m m m 1 1 0 0 1 R
○	○	×	AND<W>(mem), #		1 m 0 z m m m m 0 0 1 1 1 1 0 0 #<7:0> #<15:8>

フラグ ： S Z H V N C

S	Z	H	V	N	C
*	*	1	*	0	0

S ＝ 演算結果の最上位ビットの値がセットされます。
Z ＝ 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H ＝ “1”にセットされます。
V ＝ 演算結果のパリティ(“1”の数)が偶数のときは“1”、奇数のときは“0”がセットされます。ただし、オペランド長が32ビットの場合は、不定値がセットされます。
N ＝ “0”にクリアされます。
C ＝ “0”にクリアされます。

実行例 ： HLレジスタが7350H, IXレジスタが3456Hのとき、
AND HL, IX
を実行すると、HLレジスタは3050Hになります。

```
      0111  0011  0101  0000  ←  HLレジスタ(実行前)
AND)  0011  0100  0101  0110  ←  IX レジスタ(実行前)
      0011  0000  0101  0000  ←  HLレジスタ(実行後)
```

ANDCF num, src

< And Carry Flag キャリーフラグとの1ビット論理積 >

動 作 : CY ← CY AND src<num>

説 明 : キャリーフラグCYの内容とsrcのビットnumの内容が論理積演算され、キャリーフラグCYへ転送されます。

詳 細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	×	ANDCF	#4, r	1 1 0 z 1 r 0 0 1 0 0 0 0 0 0 0 0 0 #4
○	○	×	ANDCF	A, r	1 1 0 z 1 r 0 0 1 0 1 0 0 0
○	×	×	ANDCF	#3, (mem)	1 m 1 1 m m m m 1 0 0 0 0 #3
○	×	×	ANDCF	A, (mem)	1 m 1 1 m m m m 0 0 1 0 1 0 0 0

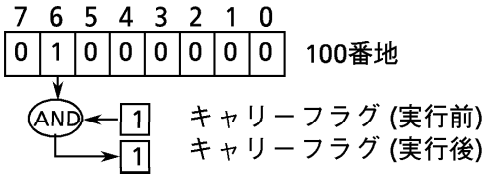
補足 : ビットnumがAレジスタで指定された場合、Aレジスタの下位4ビットの値がビットnumとして使われます。オペランドがバイトのとき、ビットnumの下位4ビットの値が8~15の場合、演算結果は不定になります。

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	*

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = キャリーフラグCYの内容とsrcのビットnumの内容の論理積演算された値がセットされます。

実行例 : 100番地のメモリの内容が01000000B (2進数)で、キャリーフラグCYが1のとき、
ANDCF 6, (100H)
を実行すると、キャリーフラグは1になります。



BIT num, src

< Bit test 1ビットのテスト >

動 作 : Zフラグ ←src<num>の反転値

説 明 : srcのビットnumの反転値が、Zフラグへ転送されます。

詳 細 :

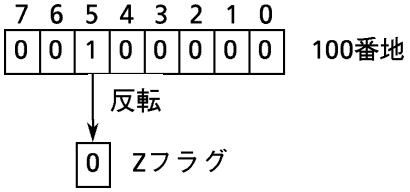
サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	×	BIT	#4, r	1 1 0 z 1 r 0 0 1 1 0 0 1 1 0 0 0 0 # 4
○	×	×	BIT	#3, (mem)	1 m 1 1 m m m m 1 1 0 0 1 #3

フラグ :

S	Z	H	V	N	C
×	*	1	×	0	–

S = 不定値がセットされます。
Z = src<num>の反転値がセットされます。
H = “1”にセットされます。
V = 不定値がセットされます。
N = “0”にリセットされます。
C = 変化なし。

実行例 : 100番地のメモリの内容が00100000B (2進数)のとき、
BIT 5, (100H)
を実行すると、Zフラグは0になります。



BS1B dst, src

＜ Bit Search 1 Backward　ビットパターン中の最初の1を後方向からサーチ ＞

動　作 ： dst←srcを後方向サーチした結果

説　明 ： srcのビットパターン中の最初の1が後方向から(MSBからLSBへ)サーチされ、その1の位置のビットナンバーが、dstへ転送されます。

詳　細 ：

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
×	○	×	BS1B	A, r	1 1 0 1 1 r 0 0 0 0 1 1 1 1

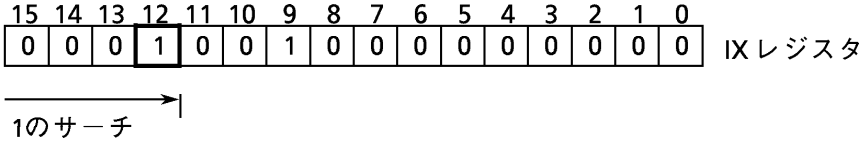
補足 ： オペランド中のdstはAレジスタ、srcはワード長のレジスタに限られます。なお、サーチしたビットパターン中に1がなかった場合、Aレジスタの値は不定になり、Vフラグが“1”にセットされます。

フラグ ：

S	Z	H	V	N	C
−	−	−	*	−	−

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ srcの内容がオールゼロの場合(ビットパターン中に1がない場合)は“1”、それ以外のときは“0”がセットされます。
N ＝ 変化なし。
C ＝ 変化なし。

実行例 ： IXレジスタが1200Hのとき、
BS1B A, IX
を実行すると、Aレジスタは0CHになります。



BS1F dst, src

< Bit Search 1 Forward ビットパターン中の最初の1を前方向からサーチ >

動 作 : dst←srcを前方向サーチした結果

説 明 : srcのビットパターン中の最初の1が前方向から(LSBからMSBへ)サーチされ、その1の位置のビットナンバーが、dstへ転送されます。

詳 細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
×	○	×	BS1F	A, r	1 1 0 1 1 r / 0 0 0 0 1 1 1 0

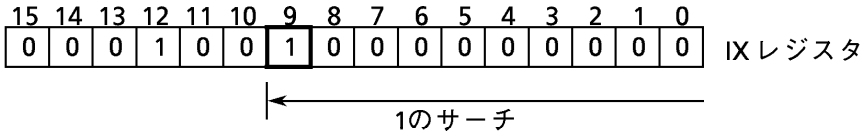
補足 : オペランド中のdstはAレジスタ、srcはワード長のレジスタに限られます。なお、サーチしたビットパターン中に1がなかった場合、Aレジスタの値は不定になり、Vフラグが“1”にセットされます。

フラグ :

S	Z	H	V	N	C
−	−	−	*	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = srcの内容がオールゼロの場合(ビットパターン中に1がない場合)は“1”、それ以外のときは“0”がセットされます。
N = 変化なし。
C = 変化なし。

実行例 : IXレジスタが1200Hのとき、
BS1F A, IX
を実行すると、Aレジスタは09Hになります。



CALL condition, dst

< Call サブルーチンコール >

動 作 : if ccが真 then XSP←XSP−4, (XSP)←32ビットPC, PC←dst.

説 明 : オペランドのconditionが真の場合、スタック領域へプログラムカウンタPCの内容が退避され、dstで示されたプログラム番地へジャンプします。

詳 細 :

ニモニック	コード
CALL #16	0 0 0 1 1 1 0 0 #<7:0> #<15:8>
CALL #24	0 0 0 1 1 1 0 1 #<7:0> #<15:8> #<23:16>
CALL [cc,] mem	1 m 1 1 m m m m 1 1 1 0 c c

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : スタックポインタXSPが100Hのとき、8000H番地のメモリにある命令、
CALL 9000H
を実行すると、メモリの0FCH番地にリターンアドレス8003H (ロングデータ) がライトされ、スタックポインタXSPは0FCHになり、9000H番地へジャンプします。

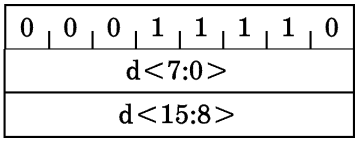
CALR dst

< Call Relative 相対サブルーチンコール >

動 作 : XSP←XSP−4, (XSP)←32ビットPC, PC←dst.

説 明 : スタック領域へプログラムカウンタPCの内容が退避され、dstで示されたプログラム番地へ相対ジャンプします。

詳 細 :

ニモニック	コード
CALR $+3+d16	

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

CCF

< Complement Carry Flag　キャリーフラグの反転 >

動　作 : CY ← CYの反転値

説　明 : キャリーフラグCYの内容が、反転されます。

詳　細 :

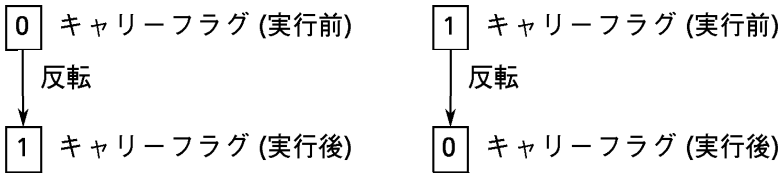
ニモニック	コード
CCF	0 0 0 1 0 0 1 0

フラグ :

S	Z	H	V	N	C
−	−	×	−	0	*

S = 変化なし。
Z = 変化なし。
H = 不定値がセットされます。
V = 変化なし。
N = “0”にリセットされます。
C = 自分自身の反転値がセットされます。

実行例 : キャリーフラグCYが0のとき、
　CCF
を実行すると、キャリーフラグCYは1になります。さらに、
　CCF
を実行すると、キャリーフラグCYは0になります。



CHG num, dst

< Change 1ビットの反転 >

動　作 : dst <num> ←dst <num> の反転値

説　明 : dstのビットnumの値が反転されます。

詳　細 :

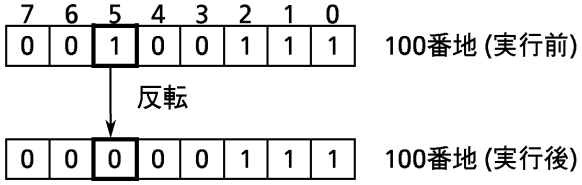
サイズ バイト	ワード	ロング	ニモニック		コード
○	○	×	CHG	#4, r	1 1 0 z 1 r 0 0 1 1 0 0 1 0 0 0 0 0 #4
○	×	×	CHG	#3, (mem)	1 m 1 1 m m m m 1 1 0 0 0 #3

フラグ :

S	Z	H	V	N	C
–	–	–	–	–	–

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : 100番地のメモリの内容が00100111B (2進数)のとき、
CHG 5, (100H)
を実行すると、100番地のメモリの内容は00000111B (2進数)になります。



CP src1, src2

< Compare 比較 >

動 作 : src1－src2

説 明 : src1の内容とsrc2の内容が比較され、その結果がフラグレジスタFに反影されます。

詳 細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	○	CP	R, r	1 1 z z 1 r 1 1 1 1 0 R
○	○	×	CP	r, #3	1 1 0 z 1 r 1 1 0 1 1 #3
○	○	○	CP	r, #	1 1 z z 1 r 1 1 0 0 1 1 1 1 #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	CP	R, (mem)	1 m z z m m m m 1 1 1 1 0 R
○	○	○	CP	(mem), R	1 m z z m m m m 1 1 1 1 1 R
○	○	×	CP<W>	(mem), #	1 m 0 z m m m m 0 0 1 1 1 1 1 1 #<7:0> #<15:8>

補 足 : オペランド中の#3は, 0~7を示します。

フラグ : S Z H V N C

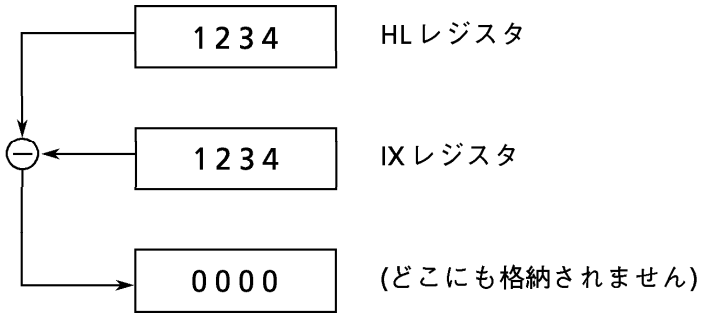
S	Z	H	V	N	C
*	*	*	*	1	*

S = 演算結果の最上位ビットの値がセットされます。
Z = 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H = 演算結果、ビット3からビット4へボローが発生したときは“1”、それ以外のときは“0”がセットされます。ただし、オペランド長が32ビットの場合は、不定値がセットされます。
V = 演算結果、オーバフローが発生したときは“1”、それ以外のときは“0”がセットされます。
N = “1”にセットされます。
C = 演算結果、最上位ビットからボローが発生したときは“1”、それ以外のときは“0”がセットされます。

実行例 : HLレジスタが1234H, IXレジスタが1234Hのとき、

```
CP  HL, IX
```

を実行すると、ZフラグとNフラグは“1”にセットされ、SフラグとHフラグ, Vフラグ, Cフラグは“0”にクリアされます。



CPD src1, src2

＜ Compare Decrement 逆方向のブロック部分サーチ ＞

動 作 : src1－src2, BC ← BC－1

説 明 : src1とsrc2の内容が比較されます。その後、BCレジスタの内容が－1されます。なお、src1はAレジスタまたはWAレジスタに限られ、src2のアドレッシングモードはポストデクリメントのレジスタ間接に限られます。

詳 細 :

サイズ			ニモニック		コード
バイト	ワード	ロング			
○	○	×	CPD	[A/WA, (R－)]	1 0 0 z 0 R 0 0 0 1 0 1 1 0

補 足 : [ ]の内側のオペランドの記述を省略すると、“A, (XHL－)”を指定したことになります。

フラグ :

S	Z	H	V	N	C
*	*	*	*	1	－

S = “src1－src2”の演算結果の最上位ビットの値がセットされます。
Z = “src1－src2”の演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H = “src1－src2”の演算結果、ビット3からビット4へボローが発生したときは“1”、それ以外のときは“0”がセットされます。
V = 命令実行後、BCレジスタの値が0のときは“0”、それ以外のときは“1”がセットされます。
N = “1”にセットされます。
C = 変化なし。

実行例 : XIXレジスタが00123456H, BCレジスタが0200Hのとき、

```
CPD  A, (XIX－)
```

を実行すると、Aレジスタの内容と123456H番地の内容が比較され、XIXレジスタは00123455H, BCレジスタは01FFHになります。

CPDR src1, src2

< Compare Decrement Repeat 逆方向のブロックサーチ >

動 作 : src1－src2, BC ← BC－1, Repeat until src1＝src2 or BC＝0

説 明 : src1とsrc2の内容が比較されます。その後、BCレジスタの内容が－1され、src1＝src2またはBC＝0でなければ、再び前記動作を繰り返します。なお、src1はAレジスタまたはWAレジスタに限られ、src2のアドレッシングモードはポストデクリメントのレジスタ間接に限られます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック		コード
○	○	×	CPDR	[A/WA, (R－)]	1 0 0 z 0 R 0 0 0 1 0 1 1 1

補足 : [ ]の内側のオペランドの記述を省略すると、“A, (XHL－)”を指定したことになります。

フラグ :

S	Z	H	V	N	C
*	*	*	*	1	–

S ＝ “src1－src2”の演算結果の最上位ビットの値がセットされます。
Z ＝ “src1－src2”の演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H ＝ “src1－src2”の演算結果、ビット3からビット4へボローが発生したときは“1”、それ以外のときは“0”がセットされます。
V ＝ 命令実行後、BCレジスタの値が0のときは“0”、それ以外のときは“1”がセットされます。
N ＝ “1”にセットされます。
C ＝ 変化なし。

実行例 : Aレジスタが55H, XIXレジスタが00123456H, BCレジスタが0003Hで、
メモリの123456H番地の内容が11H,
メモリの123455H番地の内容が22H
メモリの123454H番地の内容が33Hのとき、
CPDR A, (XIX－)
を実行すると、メモリの123456H番地, 123455番地, 123454H番地をリードした後、この命令の実行が“BC＝0”の条件で終了し、XIXレジスタは00123453H, BCレジスタは0000Hになります。

CPI src1, src2

< Compare Increment 正方向のブロック部分サーチ >

動 作 : src1－src2, BC ← BC－1

説 明 : src1とsrc2の内容が比較されます。その後、BCレジスタの内容が－1されます。なお、src1はAレジスタまたはWAレジスタに限られ、src2のアドレッシングモードはポストインクリメントのレジスタ間接に限られます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック		コード
○	○	×	CPI	[A/WA, (R+)]	1 0 0 z 0 R 0 0 0 1 0 1 0 0

補 足 : [ ]の内側のオペランドの記述を省略すると、“A, (XHL+)” を指定したことになります。

フラグ :

S	Z	H	V	N	C
*	*	*	*	1	－

S = “src1－src2” の演算結果の最上位ビットの値がセットされます。
Z = “src1－src2” の演算結果がゼロのときは “1”、それ以外のときは “0” がセットされます。
H = “src1－src2” の演算結果、ビット3からビット4へボローが発生したときは “1”、それ以外のときは “0” がセットされます。
V = 命令実行後、BCレジスタの値が0のときは “0”、それ以外のときは “1” がセットされます。
N = “1” にセットされます。
C = 変化なし。

実行例 : XIXレジスタが00123456H, BCレジスタが0200Hのとき、

```
CPI  A, (XIX+)
```

を実行すると、Aレジスタの内容と123456H番地の内容が比較され、XIXレジスタは00123457H, BCレジスタは01FFHになります。

CPIR src1, src2

＜ Compare Increment Repeat 正方向のブロックサーチ ＞

動 作 : src1－src2, BC ← BC－1, Repeat until src1＝src2 or BC＝0

説 明 : src1とsrc2の内容が比較されます。その後、BCレジスタの内容が－1され、src1＝src2またはBC＝0でなければ、再び前記動作を繰り返します。なお、src1はAレジスタまたはWAレジスタに限られ、src2のアドレッシングモードはポストインクリメントのレジスタ間接に限られます。

詳 細 :

サイズ			ニモニック		コード
バイト	ワード	ロング			
○	○	×	CPIR	[A/WA, (R＋)]	1 0 0 z 0 R 0 0 0 1 0 1 0 1

補 足 : [ ]の内側のオペランドの記述を省略すると、“A, (XHL＋)”を指定したことになります。

フラグ :

S	Z	H	V	N	C
*	*	*	*	1	－

S ＝ “src1－src2”の演算結果の最上位ビットの値がセットされます。
Z ＝ “src1－src2”の演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H ＝ “src1－src2”の演算結果、ビット3からビット4へボローが発生したときは“1”、それ以外のときは“0”がセットされます。
V ＝ 命令実行後、BCレジスタの値が0のときは“0”、それ以外のときは“1”がセットされます。
N ＝ “1”にセットされます。
C ＝ 変化なし。

実行例 : Aレジスタが33H, XIXレジスタが00123456H, BCレジスタが0200Hで、
メモリの123456H番地の内容が11H,
メモリの123457H番地の内容が22H
メモリの123458H番地の内容が33Hのとき、
CPIR A, (XIX＋)
を実行すると、メモリの123456H番地, 123457H番地, 123458H番地をリードした後、この命令の実行が“src1＝src2”の条件で終了し、XIXレジスタは00123459H, BCレジスタは01FDHになります。

CPL dst

< Complement 1の補数 >

動　作 : dst←dstの1の補数

説　明 : dstの1の補数(各ビットの0/1反転)値が、dstへ転送されます。

詳　細 :

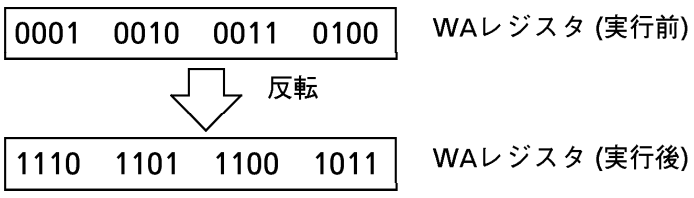
サイズ バイト	ワード	ロング	ニモニック		コード
○	○	×	CPL	r	1 1 0 z 1 r 0 0 0 0 0 1 1 0

フラグ :

S	Z	H	V	N	C
–	–	1	–	1	–

S = 変化なし。
Z = 変化なし。
H = “1”にセットされます。
V = 変化なし。
N = “1”にセットされます。
C = 変化なし。

実行例 : WAレジスタが1234Hのとき、
CPL WA
を実行すると、WAレジスタはEDCBHになります。



DAA dst

< Decimal Adjust Accumulator 10進補正 >

動 作 : dst ← dstの10進補正

説 明 : dstの内容がCフラグとHフラグ、Nフラグの状態により、10進補正されます。この命令は、加算命令または減算命令の後に、その演算結果を、2進化10進 (BCD) 表現に補正するためのものです。

詳 細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	×	×	DAA	r	1 1 0 0 1 r 0 0 0 1 0 0 0 0

演算	DAA命令を実行する前のNフラグ	DAA命令を実行する前のCフラグ	dstの上位4ビット	DAA命令を実行する前のHフラグ	dstの下位4ビット	加算される数	DAA命令を実行した後のCフラグ
	0	0	0~9	0	0~9	00	0
	0	0	0~8	0	A~F	06	0
ADD	0	0	0~9	1	0~3	06	0
	0	0	A~F	0	0~9	60	1
ADC	0	0	9~F	0	A~F	66	1
	0	0	A~F	1	0~3	66	1
	0	1	0~2	0	0~9	60	1
	0	1	0~2	0	A~F	66	1
	0	1	0~3	1	0~3	66	1
SUB	1	0	0~9	0	0~9	00	0
SBC	1	0	0~8	1	6~F	FA	0
NEG	1	1	7~F	0	0~9	A0	1
	1	1	6~F	1	6~F	9A	1

補 足 : “INC”または“DEC”命令の演算に対しては、10進補正できません。これらの命令では、Cフラグが変化しないからです。

フラグ：

S	Z	H	V	N	C
*	*	*	*	−	*

S ＝ 演算結果の最上位ビットの値がセットされます。
Z ＝ 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H ＝ 演算結果、ビット3からビット4へキャリーが発生したときは“1”、それ以外のときは“0”がセットされます。
V ＝ 演算結果のパリティ(“1”の数)が偶数のときは“1”, 奇数のときは“0”がセットされます。
N ＝ 変化なし。
C ＝ 演算結果、最上位ビットからキャリーが発生した場合、または演算前にキャリーが“1”であった場合は“1”、それ以外のときは“0”にセットされます。

実行例 ： Aレジスタが59Hで、Bレジスタが13Hのとき、

```
ADD  A, B
DAA  A
```

を実行すると、Aレジスタは72Hになります。

DEC num, dst

< Decrement 減少 >

動 作 : dst←dst−num

説 明 : dstの内容からnumの内容が減算され、dstへ転送されます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック	コード
○	○	○	DEC #3, r	1 1 z z 1 r 0 1 1 0 1 #3
○	○	×	DEC<W> #3, (mem)	1 m 0 z m m m m 0 1 1 0 1 #3

補足 : オペランド中の#3は1~8を示し、オブジェクトコードは1~7, 0に対応します。

フラグ :

S	Z	H	V	N	C
*	*	*	*	1	−

S = 演算結果の最上位ビットの値がセットされます。
Z = 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H = 演算結果、ビット3からビット4へボローが発生したときは“1”、それ以外のときは“0”がセットされます。
V = 演算結果、オーバフローが発生したときは“1”、それ以外のときは“0”がセットされます。
N = “1”にセットされます。
C = 変化なし。
(注) “DEC #3, r”でオペランドサイズがワードまたはロングの場合は、すべてのフラグは変化しません。

実行例 : HLレジスタが5678Hのとき、
DEC 4, HL
を実行すると、HLレジスタは5674Hになります。

DECF

＜ Decrement Register File Pointer　レジスタバンクの－1切り替え ＞

動　作 ： RFP＜2:0＞←RFP＜2:0＞－1

説　明 ： ステータスレジスタSR内のレジスタファイルポインタRFP＜2:0＞の内容が－1されます。なお、“RFP2”は0に固定されます。

詳　細 ：

ニモニック	コード
DECF	0 0 0 0 1 1 0 1

フラグ ：

S	Z	H	V	N	C
－	－	－	－	－	－

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ 変化なし。

実行例 ： RFP＜2:0＞の内容が“2”のとき、
　DECF
を実行すると、RFP＜2:0＞の内容は“1”になります。

DI

＜Disable Interrupt　割り込み禁止＞

動　作：IFF＜2:0＞←7

説　明：ステータスレジスタSR中の割り込み許可フラグIFF＜2:0＞の内容が“7”になります。この命令の実行後、割り込みは、ノンマスカブル割り込み(割り込み要求レベルが“7”のもの)しか受け付けなくなります。

詳　細：

ニモニック	コード
DI	0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 1

フラグ：

S	Z	H	V	N	C
−	−	−	−	−	−

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ 変化なし。

DIV dst, src

< Divide 符号なし除算 >

動 作 : dst<下位半分> ← dst ÷ src, dst<上位半分> ← 余り (符号なし)

説 明 : dstの内容がsrcの内容で符号なし除算され、商がdstの下位半分へ、余りがdstの上位半分へ転送されます。

詳 細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	×	DIV	RR, r	1 1 0 z 1 r 0 1 0 1 0 R
○	○	×	DIV	rr, #	1 1 0 z 1 r 0 0 0 0 1 0 1 0 #<7:0> #<15:8>
○	○	×	DIV	RR, (mem)	1 m 0 z m m m m 0 1 0 1 0 R

(注) RRについては次ページ参照

補 足 : 演算サイズがバイトのときは、“dst(下位バイト) ← dst(ワード) ÷ src(バイト)、dst(上位バイト) ← 余り”
演算サイズがワードのときは、“dst(下位ワード) ← dst(ロング) ÷ src(ワード)、dst(上位ワード) ← 余り”になります。オペランドのdstの記述は、被除数のサイズに合わせてください。
オーバフローしたとき(Vフラグが“1”にセットされたとき)は、dstへは不定値がセットされています。

フラグ :

S	Z	H	V	N	C
–	–	–	*	–	–

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 0で割った場合、または、商が格納用dstのビット長で表現できる数を超えた場合は“1”、それ以外のときは“0”がセットされます。
N = 変化なし。
C = 変化なし。

実行例 ： XIXレジスタが12345678Hで、IYレジスタが89ABHのとき、
DIV XIX, IY
を実行すると、商は21DAH, 余りは0FDAHとなり、XIXレジスタは
0FDA21DAHになります。

注 意 ： “DIV RR, r” と “DIV RR, (mem)” 命令のRRは、下記に示す表のようになります。

演算サイズがバイト
(8ビット←16ビット÷8ビット)

RR	コード R
WA	001
BC	011
DE	101
HL	111
IX	指定不可!
IY	
IZ	
SP	

演算サイズがワード
(16ビット←32ビット÷16ビット)

RR	コード R
XWA	000
XBC	001
XDE	010
XHL	011
XIX	100
XIY	101
XIZ	110
XSP	111

“DIV rr, #” 命令のrrは、下記に示す表のようになります。

演算サイズがバイト
(8ビット←16ビット÷8ビット)

rr	コード r
WA	001
BC	011
DE	101
HL	111
IX	C7H : F0H
IY	C7H : F4H
IZ	C7H : F8H
SP	C7H : FCH
	1バイト目 2バイト目

(注) その他のワードレジスタも
IX~SPと同様の拡張コード方式で、
すべて指定可能。

演算サイズがワード
(16ビット←32ビット÷16ビット)

rr	コード r
XWA	000
XBC	001
XDE	010
XHL	011
XIX	100
XIY	101
XIZ	110
XSP	111

(注) その他のロングワードレジスタも
拡張コード方式ですべて指定可能。

DIVS dst, src

< Divide Signed 符号付き除算 >

動 作 : dst<下位半分> ← dst ÷ src, dst<上位半分> ← 余り (符号付き)

説 明 : dstの内容がsrcの内容で符号付き除算され、商がdstの下位半分へ、余りがdstの上位半分へ転送されます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック		コード
○	○	×	DIVS	RR, r	1 1 0 z 1 r 0 1 0 1 1 R
○	○	×	DIVS	rr, #	1 1 0 z 1 r 0 0 0 0 1 0 1 1 #<7:0> #<15:8>
○	○	×	DIVS	RR, (mem)	1 m 0 z m m m m 0 1 0 1 1 R

(注) RRについては次ページ参照

補 足 : 演算サイズがバイトのときは、“dst(下位バイト) ← dst(ワード) ÷ src(バイト)、dst(上位バイト) ← 余り”
演算サイズがワードのときは、“dst(下位ワード) ← dst(ロング) ÷ src(ワード)、dst(上位ワード) ← 余り”になります。オペランドのdstの記述は、被除数のサイズに合わせてください。
なお、余りの符号は、被余数と同じになります。
オーバフローしたとき(Vフラグが“1”にセットされたとき)は、dstへは不定値がセットされています。

フラグ :

S	Z	H	V	N	C
-	-	-	*	-	-

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 0で割った場合、または、商が格納用dstのビット長で表現できる数を超えた場合は“1”、それ以外のときは“0”がセットされます。
N = 変化なし。
C = 変化なし。

実行例 ： XIXレジスタが12345678Hで、IYレジスタが89ABHのとき、
DIV XIX, IY
を実行すると、余りは16EEH、商はD89EHとなり、XIXレジスタは
16EED89EHになります。

注意 ： “DIVS RR, r” と “DIVS RR, (MEM)” 命令のRRは、下記に示す表のようになります。

演算サイズがバイト
(8ビット←16ビット÷8ビット)

RR	コード R
WA	001
BC	011
DE	101
HL	111
IX	指定不可!
IY	
IZ	
SP	

演算サイズがワード
(16ビット←32ビット÷16ビット)

RR	コード R
XWA	000
XBC	001
XDE	010
XHL	011
XIX	100
XIY	101
XIZ	110
XSP	111

“DIVS rr, #” 命令のrrは、下記に示す表のようになります。

演算サイズがバイト
(8ビット←16ビット÷8ビット)

rr	コード r
WA	001
BC	011
DE	101
HL	111
IX	C7H : F0H
IY	C7H : F4H
IZ	C7H : F8H
SP	C7H : FCH
	1バイト目 2バイト目

(注) その他のワードレジスタも
IX~SPと同様の拡張コード方式で、
すべて指定可能。

演算サイズがワード
(16ビット←32ビット÷16ビット)

rr	コード r
XWA	000
XBC	001
XDE	010
XHL	011
XIX	100
XIY	101
XIZ	110
XSP	111

(注) その他のロングワードレジスタも
拡張コード方式ですべて指定可能。

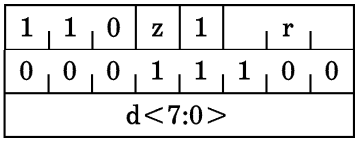
DJNZ dst1, dst2

< Decrement and Jump if Non Zero 減少&ジャンプノンゼロ >

動 作 : dst1 ← dst1 − 1. if dst1 ≠ 0 then PC ← dst2.

説 明 : dst1の内容が−1され、その値がゼロでなければ、dst2で示されたプログラム番地へ相対ジャンプします。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック		コード
○	○	×	DJNZ	[r,] $+3/4+d8	

(注) rが拡張コードで指定されるときは
$+4+d8、それ以外のときは$+3+d8
となります。

補 足 : [ ]の内側のオペランドの“r,”の記述を省略すると、Bレジスタが指定されたものと解釈されます。

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : Aレジスタが12Hで、Wレジスタが03Hのとき、

```
LOOP: ADD   A, A
      DJNZ  W, LOOP
```

を実行すると、3回ループして、Aレジスタは24H → 48H → 90H, Wレジスタは02H → 01H → 00Hになります。

EI num

＜Enable Interrupt 割り込み許可＞

動 作 : IFF＜2:0＞←num

説 明 : ステータスレジスタSR中の割り込み許可フラグIFF＜2:0＞の内容が、numになります。この命令の実行後、CPUの割り込み受け付けレベルはnumになります。

詳 細 :

ニモニック	コード
EI [#3]	0 0 0 0 0 1 1 0 0 0 0 0 0 #3

補 足 : オペランドの値は、0~7まで指定できます。オペランドの記述を省略すると、"EI 0"と認識されます。

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

EX dst, src

< Exchange 交換 >

動　作 : dst ↔ src

説　明 : dstの内容とsrcの内容が、交換されます。

詳　細 :

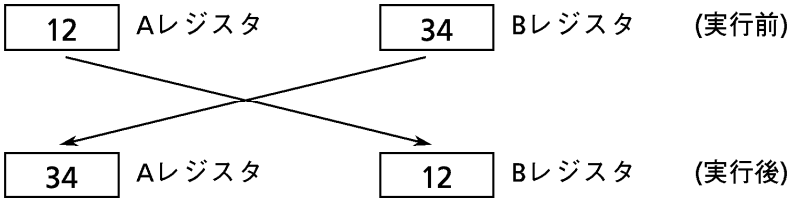
サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	×	×	EX	F, F′	0 0 0 1 0 1 1 0
○	○	×	EX	R, r	1 1 z z 1 r 1 0 1 1 1 R
○	○	×	EX	(mem), r	1 m z z m m m m 0 0 1 1 0 R

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。
(注) “EX F, F′”を実行すると、フラグはすべて変化します。

実行例 : Aレジスタが12Hで、Bレジスタが34Hのとき、
EX A, B
を実行すると、Aレジスタは34H、Bレジスタは12Hになります。



EXTS dst

< Extend Sign 符号拡張 >

動 作 : dst <上位半分> ← dst <下位半分> の符号ビット

説 明 : dstの下位半分の符号ビット(オペランドサイズがワードのときはビット7, ロングワードのときはビット15)が、dstの上位半分の全ビットに転送(コピー)されます。

詳 細 :

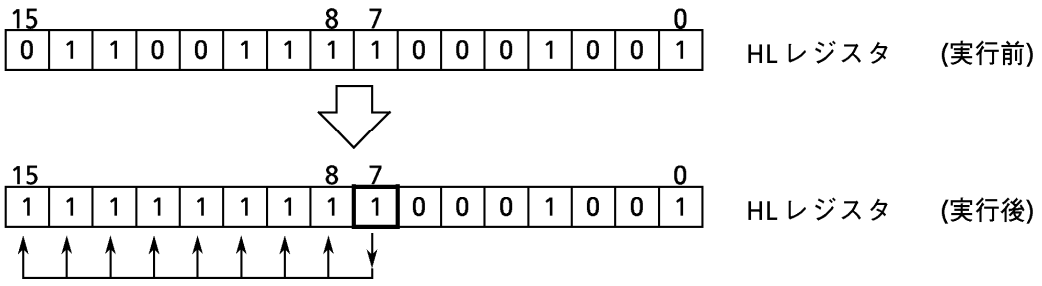
サイズ			ニモニック		コード
バイト	ワード	ロング			
×	○	○	EXTS	r	1 1 z z 1 r / 0 0 0 1 0 0 1 1

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : HLレジスタが6789Hのとき、
EXTS HL
を実行すると、HLレジスタはFF89Hになります。



EXTZ dst

＜ Extend Zero ゼロ拡張＞

動 作 : dst＜上位半分＞←0

説 明 : dstの上位半分がゼロにクリアされます。通常オペランドサイズが異なる演算を行う前、オペランドサイズを揃えるために使用します。

詳 細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
×	○	○	EXTZ	r	1 1 z z 1 r 0 0 0 1 0 0 1 0

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ 変化なし。

実行例 : HLレジスタが6789Hのとき、
EXTZ HL
を実行すると、HLレジスタは0089Hになります。

XIXレジスタが12345678Hのとき、
EXTZ XIX
を実行すると、XIXレジスタは00005678Hになります。

HALT

< Halt　CPUの停止 >

動　作 : CPUの停止

説　明 : 命令の実行が停止されます。再開は、割り込みを受け付けることによって行われます。

詳　細 :

ニモニック	コード
HALT	0 0 0 0 0 1 0 1

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

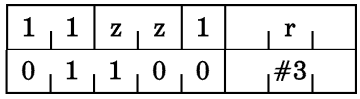
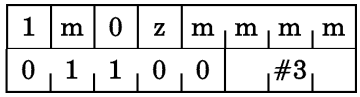
INC num, dst

< Increment 増加 >

動　作 : dst←dst+num

説　明 : dstの内容とnumの内容が加算され、dstへ転送されます。

詳　細 :

サイズ バイト	ワード	ロング	ニモニック		コード
○	○	○	INC	#3, r	
○	○	×	INC<W>	#3, (mem)	

補　足 : オペランド中の#3は1~8を示し、オブジェクトコードは1~7, 0に対応します。

フラグ :

S	Z	H	V	N	C
*	*	*	*	0	–

S = 演算結果の最上位ビットの値がセットされます。
Z = 演算結果がゼロのときは“1”、それ以外のときは“0”がセットされます。
H = 演算結果、ビット3からビット4へキャリーが発生したときは“1”、それ以外のときは“0”がセットされます。
V = 演算結果、オーバフローが発生したときは“1”、それ以外のときは“0”がセットされます。
N = “0”にクリアされます。
C = 変化なし。
(注) “INC #3, r”でオペランドサイズがワードまたはロングの場合は、すべてのフラグは変化しません。

実行例 : WAレジスタが1234Hのとき、
INC 5, WA
を実行すると、WAレジスタは1239Hになります。

INCF

＜ Increment Register File Pointer　レジスタバンクの＋1切り替え ＞

動　作 ： RFP＜2:0＞←RFP＜2:0＞＋1

説　明 ： ステータスレジスタSP内のレジスタファイルポインタRFP＜2:0＞の内容が＋1されます。なお、“RFP2”は0に固定されます。

詳　細 ：

ニモニック	コード
INCF	0 0 0 0 1 1 0 0

フラグ ：

S	Z	H	V	N	C
－	－	－	－	－	－

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ 変化なし。

実行例 ： RFP＜2:0＞の内容が“2”のとき、
INCF
を実行すると、RFP＜2:0＞の内容は“3”になります。

JP condition, dst

< Jump ジャンプ >

動 作 : if ccが真 then PC←dst.

説 明 : オペランドのconditionが真の場合、dstで示されたプログラム番地へジャンプします。

詳 細 :

ニモニック		コード
JP	#16	0 0 0 1 1 0 1 0 #<7:0> #<15:8>
JP	#24	0 0 0 1 1 0 1 1 #<7:0> #<15:8> #<23:16>
JP	[cc,] mem	1 m 1 1 m m m m 1 1 0 1 c c

フラグ :

S	Z	H	V	N	C
–	–	–	–	–	–

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : 下記の命令、

JP 2000H

を実行すると、無条件で2000H番地へジャンプします。
XIXレジスタが00123456Hで、

JP C, XIX+2

を実行すると、キャリーフラグの値が“1”の場合、123458H番地へジャンプします。

JR condition, dst

< Jump Relative 相対ジャンプ >

動 作 : if ccが真 then PC←dst.

説 明 : オペランドのconditionが真の場合、dstで示されたプログラム番地へ相対ジャンプします。

詳 細 :

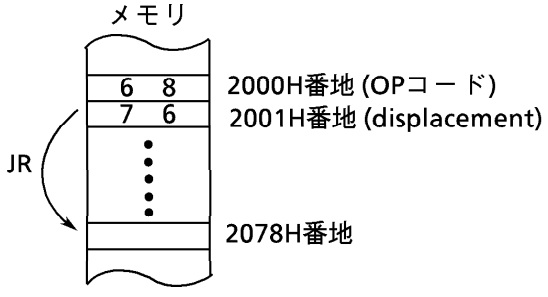
ニモニック		コード
JR	[cc,] $+2+d8	0 1 1 0 c c / d<7:0>
JRL	[cc,] $+3+d16	0 1 1 1 c c / d<7:0> / d<15:8>

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : 2000H番地のメモリにある下記の命令、
JR 2078H
を実行すると、無条件で2078H番地へ相対ジャンプします。
なお、上記命令のオブジェクトコードは、68H:76Hになります。



LD dst, src

< Load 転送 >

動　作 : dst ← src

説　明 : srcの内容が、dstへ転送されます。

詳　細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	○	LD	R, r	11zz1r 10001R
○	○	○	LD	r, R	11zz1r 10011R
○	○	○	LD	r, #3	11zz1r 10101#3
○	○	○	LD	R, #	0zzz0R #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	LD	r, #	11zz1r 00000011 #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	LD	R, (mem)	1mzzmmmm 00100R
○	○	○	LD	(mem), R	1m11mmmm 01zz0R
○	○	×	LD<W>	(#8), #	00010z0 #8 #<7:0> #<15:8>

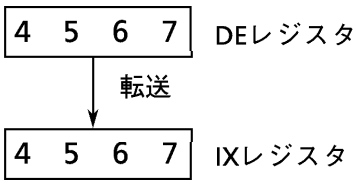
サイズ バイト	ワード	ロング	ニモニック	コード
○	○	×	LD<W> (mem), #	1 m 1 1 m m m m 0 0 0 0 0 0 z 0 #<7:0> #<15:8>
○	○	×	LD<W> (#16), (mem)	1 m 0 z m m m m 0 0 0 1 1 0 0 1 #16<7:0> #16<15:8>
○	○	×	LD<W> (mem), (#16)	1 m 1 1 m m m m 0 0 0 1 0 1 z 0 #16<7:0> #16<15:8>

フラグ：

S	Z	H	V	N	C
–	–	–	–	–	–

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ 変化なし。

実行例 ： DEレジスタが4567Hのとき、
LD IX, DE
を実行すると、IXレジスタは4567Hになります。



LDA dst, src

< Load Address 実効アドレスの転送 >

動　作 : dst ← srcの実効アドレス値

説　明 : srcの実効アドレス値が、dstへ転送されます。

詳　細 :

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
×	○	○	LDA	R, mem	1 m 1 1 m m m m 0 0 1 s 0 R

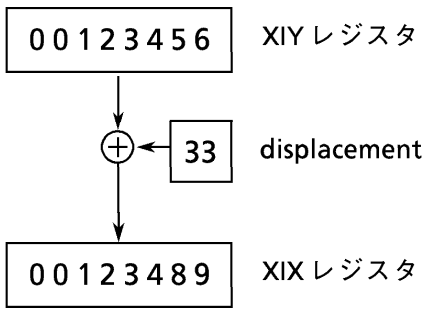
補　足 : 本命令は、ADD命令と動作が似ていますが、dstの指定がsrcの指定とは別にできる点が特長です。主に、Cコンパイラなどでポインタを扱う場合に使用します。

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : XIYレジスタが00123456Hで
LDA XIX, XIY+33H
を実行すると、 XIXレジスタは00123489Hになります。



LDAR dst, src

< Load Address Relative 相対アドレスの転送 >

動 作 : dst ← srcの相対アドレス値

説 明 : srcで指定された相対アドレス値が、dstへ転送されます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック		コード
×	○	○	LDAR	R, $+4+d16	

1	1	1	1	0	0	1	1
0	0	0	1	0	0	1	1
d<7:0>							
d<15:8>							
0	0	1	s	0	R		

フラグ :

S	Z	H	V	N	C
–	–	–	–	–	–

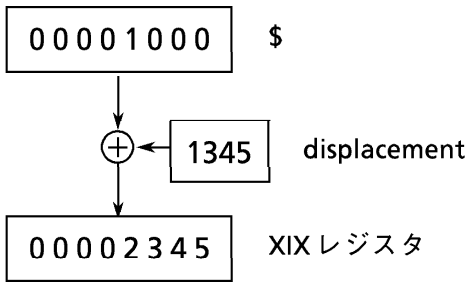
S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : 1000H番地のメモリにある下記の命令、

LDAR XIX, $+1345H

を実行すると、XIXレジスタは00002345Hになります。"$"は、その命令が置かれている先頭番地を示します。

なお、上記命令のオブジェクトコードは、F3H: 13H: 41H: 13H: 34Hになります。



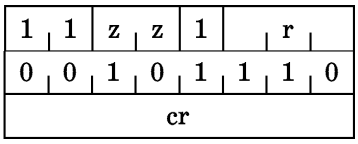
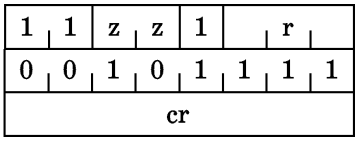
LDC dst, src

< Load Control コントロールレジスタの転送 >

動　作 : dst ← src

説　明 : srcの内容が、dstへ転送されます。

詳　細 :

サイズ バイト	ワード	ロング	ニモニック		コード
○	○	○	LDC	cr, r	
○	○	○	LDC	r, cr	

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S = 変化なし。
Z = 変化なし。
H = 変化なし。
V = 変化なし。
N = 変化なし。
C = 変化なし。

実行例 : WAレジスタが1234Hのとき、
LDC DMAC0, WA
を実行すると、コントロールレジスタDMAC0は1234Hになります。

LDCF num, src

＜ Load Carry Flag　キャリーフラグへの1ビット転送 ＞

動　作 ： CY←src＜num＞

説　明 ： srcのビットnumの内容が、キャリーフラグCYへ転送されます。

詳　細 ：

サイズ バイト	サイズ ワード	サイズ ロング	ニモニック		コード
○	○	×	LDCF	#4, r	1 1 0 z 1 r 0 0 1 0 0 0 1 1 0 0 0 0 # 4
○	○	×	LDCF	A, r	1 1 0 z 1 r 0 0 1 0 1 0 1 1
○	×	×	LDCF	#3, (mem)	1 m 1 1 m m m m 1 0 0 1 1 #3
○	×	×	LDCF	A, (mem)	1 m 1 1 m m m m 0 0 1 0 1 0 1 1

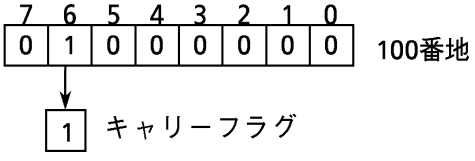
補　足 ： ビットnumがAレジスタで指定された場合、Aレジスタの下位4ビットの値がビットnumとして使われます。オペランドがバイトのとき、ビットnumの下位4ビットの値が8~15の場合、キャリーフラグの値は不定になります。

フラグ ：

S	Z	H	V	N	C
−	−	−	−	−	*

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ srcのビットnumの内容がセットされます。

実行例 ： 100番地のメモリの内容が01000000B (2進数)のとき、
　LDCF　6, (100H)
を実行すると、キャリーフラグは1になります。



LDD dst, src

< Load Decrement 逆方向のブロック部分転送 >

動 作 : dst ← src, BC ← BC − 1

説 明 : srcの内容が、dstへ転送されます。その後、BCレジスタの内容が−1されます。なお、srcおよびdstのアドレッシングモードはポストデクリメントのレジスタ間接に限られます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック	コード
○	○	×	LDD<W> [(XDE−), (XHL−)]	1 0 0 z 0 0 1 1 0 0 0 1 0 0 1 0
○	○	×	LDD<W> (XIX−), (XIY−)	1 0 0 z 0 1 0 1 0 0 0 1 0 0 1 0

[ ]はその内側の記述が省略可能であることを示しています。

フラグ :

S	Z	H	V	N	C
−	−	0	*	0	−

S = 変化なし。
Z = 変化なし。
H = "0"にクリアされます。
V = 命令実行後、BCレジスタの値が0のときは"0"、それ以外のときは"1"がセットされます。
N = "0"にクリアされます。
C = 変化なし。

実行例 : XIXレジスタが00123456Hで、XIYレジスタが00335577HでBCレジスタが0700Hのとき、

LDD (XIX−), (XIY−)

を実行すると、335577H番地の内容が123456H番地へ転送され、XIXレジスタは123455H, XIYレジスタは00335576H, BCレジスタは06FFHになります。

LDDR dst, src

< Load Decrement Repeat 逆方向のブロック転送 >

動 作 : dst ← src, BC ← BC－1, Repeat until BC＝0

説 明 : srcの内容が、dstへ転送されます。その後、BCレジスタの内容が－1され、その値が0でなければ、再び前記動作を繰り返します。なお、srcおよびdstのアドレッシングモードはポストデクリメントのレジスタ間接に限られます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック	コード
○	○	×	LDDR<W> [(XDE－), (XHL－)]	1 0 0 z 0 0 1 1 0 0 0 1 0 0 1 1
○	○	×	LDDR<W> (XIX－), (XIY－)	1 0 0 z 0 1 0 1 0 0 0 1 0 0 1 1

[ ]はその内側の記述が省略可能であることを示しています。

補 足 : 割り込み要求は、1データを転送するごとにサンプリングされます。

フラグ :

S	Z	H	V	N	C
－	－	0	0	0	－

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ “0”にクリアされます。
V ＝ “0”にクリアされます。
N ＝ “0”にクリアされます。
C ＝ 変化なし。

実行例 : XIXレジスタが00123456Hで、XIYレジスタが00335577Hで、BCレジスタが0003Hのとき、

LDDR (XIX－), (XIY－)

を実行すると、

335577H番地の内容が123456H番地へ、
335576H番地の内容が123455H番地へ、
335575H番地の内容が123454H番地へ

転送され、XIXレジスタは00123453H, XIYレジスタは00335574H, BCレジスタは0000Hになります。

LDF num

＜ Load Register File Pointer　レジスタバンクの設定＞

動　作 : RFP＜2:0＞←num

説　明 : numの値が、ステータスレジスタSR内のレジスタファイルポインタRFP＜2:0＞へ転送されます。なお、“RFP2”は0に固定されています。

詳　細 :

ニモニック	コード
LDF #3	0 0 0 1 0 1 1 1 0 0 0 0 0 #3

補　足: オペランドの値は、0~3まで指定できます。

フラグ :

S	Z	H	V	N	C
−	−	−	−	−	−

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ 変化なし。
V ＝ 変化なし。
N ＝ 変化なし。
C ＝ 変化なし。

LDI dst, src

＜ Load Increment　正方向のブロック部分転送 ＞

動　作 : dst ← src, BC ← BC－1

説　明 : srcの内容が、dstへ転送されます。その後、BCレジスタの内容が－1されます。
なお、srcおよびdstのアドレッシングモードはポストインクリメントのレジスタ間接に限られます。

詳　細 :

サイズ バイト	ワード	ロング	ニモニック	コード
○	○	×	LDI＜W＞ [(XDE＋),(XHL＋)]	1 0 0 z 0 0 1 1 0 0 0 1 0 0 0 0
○	○	×	LDI＜W＞ (XIX＋),(XIY＋)	1 0 0 z 0 1 0 1 0 0 0 1 0 0 0 0

[ ]はその内側の記述が省略
可能であることを示しています。

フラグ :

S	Z	H	V	N	C
－	－	0	*	0	－

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ “0”にクリアされます。
V ＝ 命令実行後、BCレジスタの値が0のときは“0”、それ以外のときは“1”がセットされます。
N ＝ “0”にクリアされます。
C ＝ 変化なし。

実行例 : XIXレジスタが00123456Hで、XIYレジスタが00335577Hで
BCレジスタが0700Hのとき、
　LDI　(XIX＋),(XIY＋)
を実行すると、335577H番地の内容が123456H番地へ転送され、XIXレジスタは00123457H、XIYレジスタは00335578H、BCレジスタは06FFHになります。

LDIR dst, src

＜ Load Increment Repeat 正方向のブロック転送 ＞

動 作 : dst ← src, BC ← BC－1, Repeat until BC＝0

説 明 : srcの内容が、dstへ転送されます。その後、BCレジスタの内容が－1され、その値が0でなければ、再び前記動作を繰り返します。なお、srcおよびdstのアドレッシングモードはポストインクリメントのレジスタ間接に限られます。

詳 細 :

サイズ バイト	ワード	ロング	ニモニック	コード
○	○	×	LDIR＜W＞ [(XDE＋),(XHL＋)]	1 0 0 z 0 0 1 1 0 0 0 1 0 0 0 1
○	○	×	LDIR＜W＞ (XIX＋),(XIY＋)	1 0 0 z 0 1 0 1 0 0 0 1 0 0 0 1

[ ]はその内側の記述が省略可能であることを示しています。

補 足 : 割り込み要求は、1データを転送するごとにサンプリングされます。

フラグ :

S	Z	H	V	N	C
－	－	0	0	0	－

S ＝ 変化なし。
Z ＝ 変化なし。
H ＝ “0”にクリアされます。
V ＝ “0”にクリアされます。
N ＝ “0”にクリアされます。
C ＝ 変化なし。

実行例 : XIXレジスタが00123456Hで、XIYレジスタが00335577Hで、
BCレジスタが0003Hのとき、
LDIR (XIX＋),(XIY＋)
を実行すると、
335577H番地の内容が123456H番地へ、
335578H番地の内容が123457H番地へ、
335579H番地の内容が123458H番地へ、
転送され、XIXレジスタは00123459H、XIYレジスタは0033557AH, BCレジスタは0000Hになります。