HALT

< Halt CPU >

Operation : CPU halt

Description : Halts the instruction execution. To resume, an interrupt must de received.

Details :

Mnemonic	Code
HALT	0 0 0 0 0 1 0 1

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

INC num, dst

< Increment >

Operation : dst←dst+num

Description : Adds the contents of dst and num and transfers the result to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	○	○	INC	#3, r	1 1 z z 1 r 0 1 1 0 0 #3
○	○	×	INC<W>	#3, (mem)	1 m 0 z m m m m 0 1 1 0 0 #3

Note : #3 in operands indicates from 1 to 8 and object codes correspond from 1 to 7,0.

Flags :

S	Z	H	V	N	C
*	*	*	*	0	–

S = MSB value of the result is set.
Z = 1 is set if the result is 0, otherwise 0.
H = 1 is set if a carry occurs from bit 3 to bit 4 as a result of the operation, otherwise 0.
V = 1 is set if an overflow occurs as a result of the operation, otherwise 0.
N = Cleared to zero.
C = No change

Note : With the INC #3,r instruction, if the operand is a word or a long word, no flags change.

Execution exampl : INC 5,WA
When the WA register = 1234H, execution sets the WA register to 1239H.

INCF

< Increment Register File Pointer >

Operation : RFP<2:0> ← RFP<2:0> + 1
Description : Increments the contents of RFP<2:0> in the status register by 1. RFP2 is fixed to 0.

Details :

Mnemonic	Code
INCF	0 0 0 0 1 1 0 0

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: INCF
When the contents of RFP<2:0> = 2, execution sets the contents of RFP<2:0> to 3.

JP condition, dst

< Jump >

Operation : If cc is true, then PC ← dst.

Description : If the operand condition is true, jumps to the program address specified by dst.

Details :

Mnemonic		Code
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

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: JP 2000H
Execution jumps unconditionally to address 2000H.

JP C, XIX+2
When the carry flag = 1, execution jumps to address 123458H.
Condition: Register XIX = 00123456H

JR condition, dst

< Jump Relative >

Operation : If cc is true, then PC ← dst.

Description : If the operand condition is true, makes a relative jump to the program address specified by dst.

Details :

Mnemonic		Code
JR	[cc,] $+2+d8	0 1 1 0 c c d<7:0>
JRL	[cc,] $+3+d16	0 1 1 1 c c #<7:0> #<15:8>

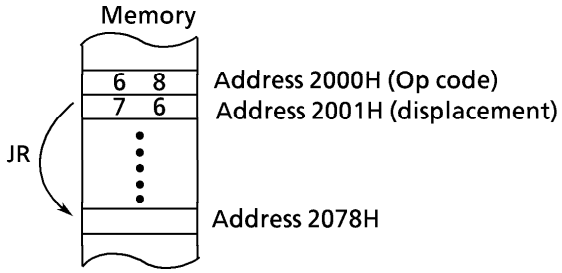
Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: JR 2078H

When this instruction is executed at memory address 2000H, execution relative jumps unconditionally to address 2078H. The object code of the instruction is 68H : 76H.



LD dst, src

< Load >

Operation : dst ← src

Description : Loads the contents of src to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	○	○	LD	R, r	1 1 z z 1 r 1 0 0 0 1 R
○	○	○	LD	r, R	1 1 z z 1 r 1 0 0 1 1 R
○	○	○	LD	r, #3	1 1 z z 1 r 1 0 1 0 1 #3
○	○	○	LD	R, #	0 z z z 0 R #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	LD	r, #	1 1 z z 1 r 0 0 0 0 0 0 1 1 #<7:0> #<15:8> #<23:16> #<31:24>
○	○	○	LD	R, (mem)	1 m z z m m m m 0 0 1 0 0 R
○	○	○	LD	(mem), R	1 m 1 1 m m m m 0 1 z z 0 R
○	○	×	LD<W>	(#8), #	0 0 0 0 1 0 z 0 #8 #<7:0> #<15:8>

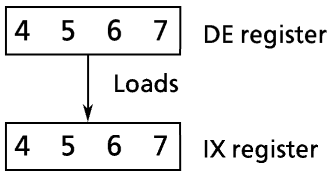
Size			Mnemonic		Code
Byte	Word	Long word			
○	○	×	LD<W>	(mem), #	1 m 1 1 m m m m 0 0 0 0 0 0 z 0 #<7:0> #<15:8>
○	○	×	LD<W>	(#16), (mem)	1 m 0 z m m m m 0 0 0 1 1 0 0 1 #16<7:0> #16<15:8>
○	○	×	LD<W>	(mem), (#16)	1 m 1 1 m m m m 0 0 0 1 0 1 z 0 #16<7:0> #16<15:8>

Flags : S Z H V N C

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: LD IX, DE
When the DE register = 4567H, execution sets the IX register to 4567H.



LDA dst, src

< Load Address >

Operation : dst ← src effective address value

Description : Loads the src effective address value to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	○	LDA	R, mem	1 m 1 1 m m m m / 0 0 1 s 0 R

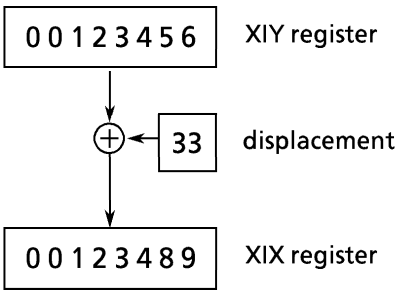
Note : This instruction operates much like the ADD instruction; the difference is that dst is specified independently from src. Mainly used for handling the pointer with the C compiler.

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: LDA XIX, XIY + 33H
When the XIY register = 00123456H, execution sets the XIX register to 00123489H.



LDAR dst, src

< Load Address Relative >

Operation : dst ← src relative address value

Description : Loads the relative address value specified in src to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	○	LDAR	R, $+4+d16	

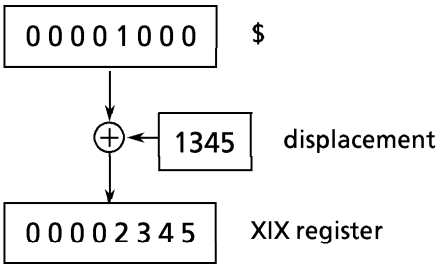
1	1	1	1	0	0	1	1	
0	0	0	1	0	0	1	1	
d<7:0>								
d<15:8>								
0	0	1	s	0	R			

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: LDAR XIX, $+1345H
When this instruction is executed at memory address 1000H, execution sets the XIX register to 00002345H. $ indicates the start address of the instruction. The instruction's object codes are: F3H:13H:41H:13H:34H.



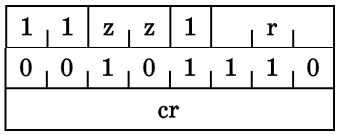
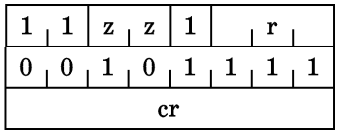
LDC dst, src

< Load Control Register >

Operation : dst ← src

Description : Loads the contents of src to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	○	○	LDC	cr, r	
○	○	○	LDC	r, cr	

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: LDC DMAC0, WA
When the WA register = 1234H, execution sets control register DMAC0 to 1234H.

LDCF num, src

< Load Carry Flag >

Operation : CY ← src<num>

Description : Loads the contents of bit num of src to the carry flag.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	○	×	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

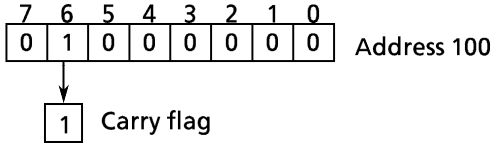
Notes : When bit num is specified by the A register, the value of the lower 4 bits of the A register is used as bit num. When the operand is a byte and the value of the lower 4 bits of bit num is from 8 to 15, the value of the carry flag is undefined.

Flags :

S	Z	H	V	N	C
–	–	–	–	–	*

S = No change
Z = No change
H = No change
V = No change
N = No change
C = Contents of bit num of src is set.

Execution example: LDCF 6, (100H)
When the contents of memoryad address 100 = 01000000B (binary), execution sets the carry flag to 1.



LDD dst, src

< Load Decrement >

Operation : dst ← src, BC ← BC − 1

Description : Loads the contents of src to dst, then decrements the contents of the BC register by 1. src and dst must be in post-decrement register indirect addressing mode.

Details :

Size			Mnemonic	Code
Byte	Word	Long word		
○	○	×	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

* Coding in square brackets [ ] can be omitted.

Flags :

S	Z	H	V	N	C
−	−	0	*	0	−

S = No change
Z = No change
H = Cleared to 0.
V = 0 is set if the BC register value is 0 after execution, otherwise 1.
N = Cleared to zero.
C = No change

Execution example: LDD (XIX−), (XIY−)
When the XIX register = 00123456H, the XIY register = 00335577H, and the BC register = 0700H, the results of the execution are as follows:
Loads the contents of address 335577H to 123456H.
Sets the XIX register to 00123456H.
Sets the XIY register to 00335576H.
Sets the BC register to 06FFH.

LDDR dst, src

< Load Decrement Repeat >

Operation : dst ← src, BC ← BC − 1, Repeat until BC = 0

Description : Loads the contents of src to dst, then decrements the contents of the BC register by 1. If the result is other than 0, the operation is repeated. src and dst must be in post-decrement register indirect addressing mode.

Details :

Size			Mnemonic	Code
Byte	Word	Long word		
○	○	×	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

* Coding in square brackets [ ] can be omitted.

Note : Interrupt requests are sampled every time 1 item of data is loaded.

Flags :

S	Z	H	V	N	C
–	–	0	0	0	–

S = No change
Z = No change
H = Cleared to zero.
V = Cleared to zero.
N = Cleared to zero.
C = No change

Execution example: LDDR (XIX−), (XIY−)
When the XIX register = 00123456H, the XIY register = 00335577H, and the BC register = 0003H, the results of the execution are as follows:
Loads the contents of address 335577H to 123456H.
Loads the contents of address 335576H to 123455H.
Loads the contents of address 335575H to 123454H.
Sets the XIX register to 00123453H.
Sets the XIY register to 00335574H.
Sets the BC register to 0000H.

LDF num

< Load Register File Pointer >

Operation : RFP<2: 0> ← num

Description : Loads the num value to the register file pointer RFP<2:0> in status register. RFP2 is fixed to 0.

Details :

Mnemonic		Code
LDF	#3	0 0 0 1 0 1 1 1 0 0 0 0 0 #3

Note : In minimum mode, the operand value can be specified from 0 to 7; in maximum mode, from 0 to 3.

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

LDI dst, src

< Load Increment >

Operation : dst ← src, BC ← BC – 1

Description : Loads the contents of src to dst, then decrements the contents of the BC register by 1. src and dst must be in post-increment register indirect addressing mode.

Details :

Size			Mnemonic	Code
Byte	Word	Long word		
○	○	×	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

* Coding in square brackets [ ] can be omitted.

Flags :

S	Z	H	V	N	C
–	–	0	*	0	–

S = No change
Z = No change
H = Cleared to zero.
V = 0 is set when the BC register value is 0 after execution, otherwise 1.
N = Cleared to zero.
C = No change

Execution example: LDI (XIX+), (XIY+)
When the XIX register = 00123456H, the XIY register = 00335577H, and the BC register = 0700H, execution results as follows:
Loads the contents of address 335577H to 123456H.
Sets the XIX register to 00123457H.
Sets the XIY register to 00335578H.
Sets the BC register to 06FFH.

LDIR dst, src

< Load Increment Repeat >

Operation : dst ← src, BC ← BC − 1, Repeat until BC = 0

Description : Loads the contents of src to dst, then decrements the contents of the BC register by 1. If the result is other than 0, the operation is repeated. src and dst must be in post-increment register indirect addressing mode.

Details :

Size			Mnemonic	Code
Byte	Word	Long word		
○	○	×	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

* Coding in square brackets [ ] can be omitted.

Note : Interrupt requests are sampled every time 1 item of data is loaded.

Flags :

S	Z	H	V	N	C
–	–	0	0	0	–

S = No change
Z = No change
H = Cleared to zero.
V = Cleared to zero.
N = Cleared to zero.
C = No change

Execution example: LDIR (XIX+), (XIY+)
When the XIX register = 00123456H, the XIY register = 00335577H, and the BC register = 0003H, execution results as follows:
Loads the contents of address 335577H to 123456H.
Loads the contents of address 335578H to 123457H.
Loads the contents of address 335579H to 123458H.
Sets the XIX register to 00123459H.
Sets the XIY register to 0033557AH.
Sets the BC register to 0000H.

LDX dst, src

< Load eXtract >

Operation : dst ← src

Description : Loads the contents of src to dst. The effective code is assigned to this instruction every other byte. Used to fetch the code from 8-bit data bus memory in 16-bit data bus mode.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	×	×	LDX	(#8) , #	

1	1	1	1	0	1	1	1
0	0	0	0	0	0	0	0
#8							
0	0	0	0	0	0	0	0
#							
0	0	0	0	0	0	0	0

Note : Even if the second, fourth, or sixth instruction code value is not 00H, the instruction operates correctly.

Flags :

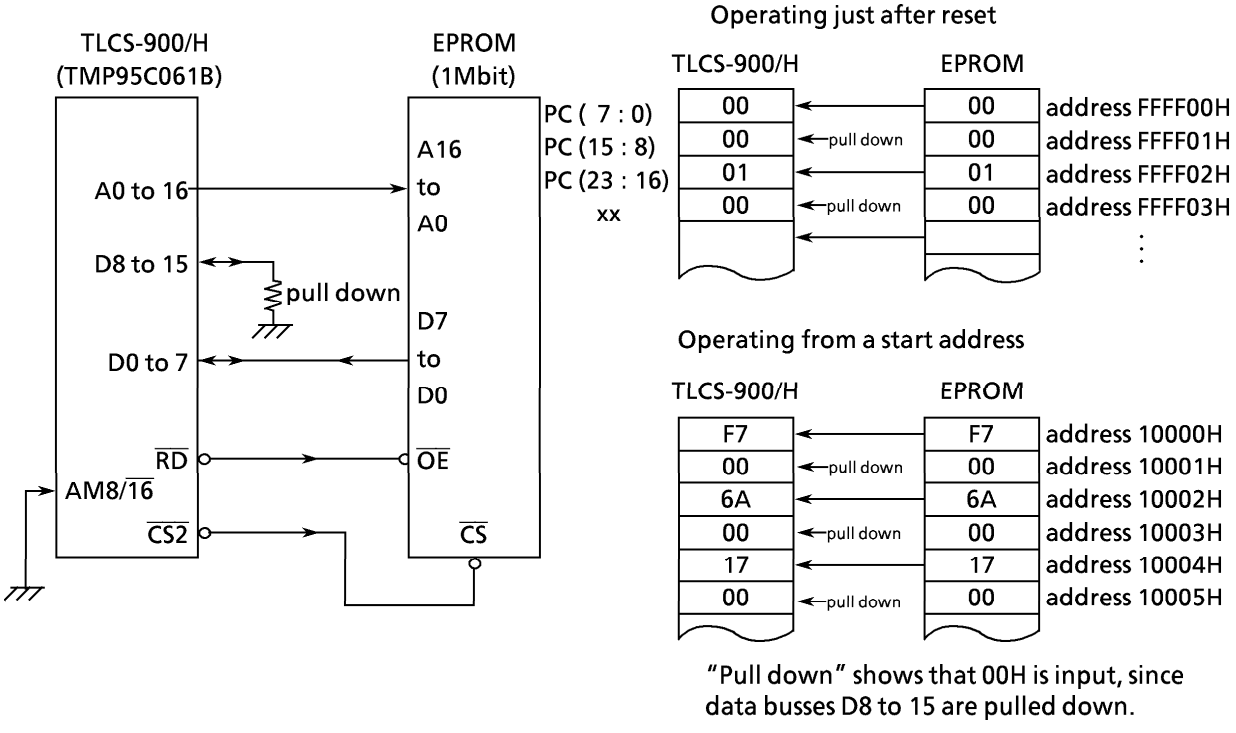
S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: In TMP95C061B, an example to run a program with EPROM which has a data bus of 8-bit size is shown as follows:
When AM8/$\overline{16}$ pin is fixed to low level at reset, TMP95C061B starts fetching a start address in 16-bit data bus mode from address FFFF00H of the reset vector. When a program is started with an external memory which has 8-bit data bus, the data buses D8 to 15 pins must be fixed with pull-up/down at the upper side.
For example, when a start address of the program is set to address 10000H, place address 10000H to address FFFF00H of a memory address and fix the data buses D8 to 15 to pull-down. Additionally, place an instruction mentioned below to address 10000H of a start address.

LDX (6AH), 17H

The instruction mentioned above is performed, and a data of 17H is written to a control register which is located at address 6AH in a built-in programmable chip select/wait controller. As a result, memory addresses 000080H to FFFFFFH become 0WAIT mode with 8-bit data bus.



"Pull down" shows that 00H is input, since data busses D8 to 15 are pulled down.

LINK dst, num

< Link >

Operation : (−XSP) ← dst, dst← XSP, XSP← XSP + num

Description : Saves the contents of dst to the stack area. Loads the contents of stack pointer XSP to dst. Adds the contents of XSP to those of num (signed) and loads the result to XSP. Used for obtaining a local variable area in the stack area for -num bytes.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	×	○	LINK	r, d16	1 1 1 0 1 r 0 0 0 0 1 1 0 0 d<7:0> d<15:8>

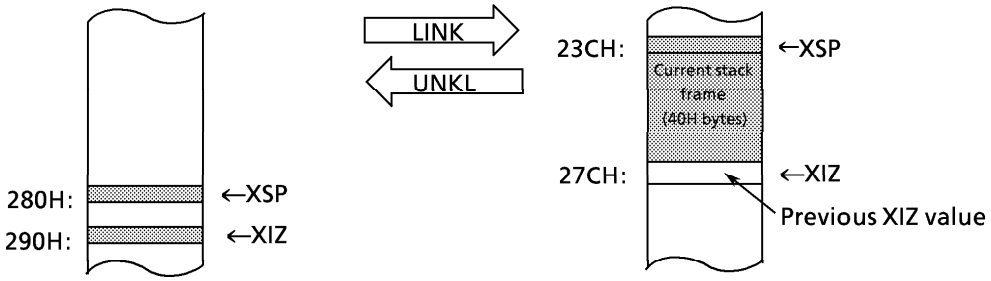
Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: LINK XIZ, −40H

When stack pointer XSP = 280H and the XIZ register = 290H, execution writes 00000290H (long data) at memory address 27CH and sets the XIZ register to 27CH and the stack pointer to XSP 23CH.



MDEC1 num, dst

< Modulo Decrement 1 >

Operation : if (dst mod num) = 0 then dst ← dst + (num − 1) else dst ← dst − 1.

Description : When the modulo num of dst is 0, increments dst by num -1 .
Otherwise, decrements dst by 1. Used to operate pointers for cyclic memory table.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MDEC1	#, r	

1	1	0	1	1	r			
0	0	1	1	1	1	0	0	
#<7:0> − 1								
#<15:8>								

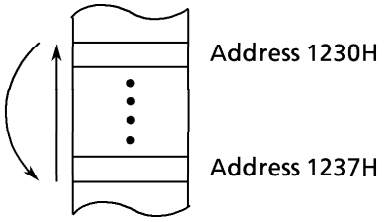
Note : The operand # must be 2 to the nth power. (n = 1 to 15)

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: Decrements the IX register by cycling from 1230H to 1237H.
MDEC1 8, IX
When the IX register = 1231H, execution sets the IX register to 1230H. Further execution increments the IX register by 8-1 and sets the IX register to 1237H, since the IX register modulo 8 = 0.



MDEC2 num, dst

< Modulo Decrement 2 >

Operation : if (dst mod num) = 0 then dst ← dst + (num − 2) else dst ← dst − 2.

Description : When the modulo num of dst is 0, increments dst by num − 2.
Otherwise, decrements dst by 2. Used to operate pointers for cyclic memory table.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MDEC2	#, r	

1	1	0	1	1	r			
0	0	1	1	1	1	0	1	
#<7:0> − 2								
#<15:8>								

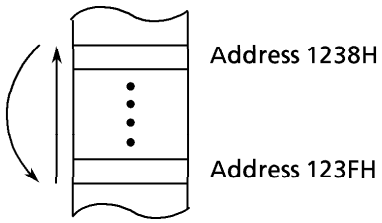
Note : The operand # must be 2 to the nth power. (n = 2 to 15)

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: Decrements the IX register by cycling from 1238H to 123FH.
MDEC2 8,IX
When the IX register = 123AH, execution sets the IX register to 1238H. Further execution increments the IX register by 8-2 and sets the IX register to 123EH, since the IX register modulo 8 = 0.



MDEC4 num, dst

< Modulo Decrement 4 >

Operation : if (dst mod num)=0 then dst←dst+(num−4) else dst←dst−4.

Description : When the modulo num of dst is 0, increments dst by num−4. Otherwise, decrements dst by 4. Used to operate pointers for cyclic memory table.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MDEC4	#, r	

1	1	0	1	1	r			
0	0	1	1	1	1	1	0	
#<7:0>−4								
#<15:8>								

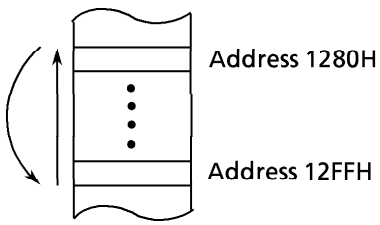
Note : The operand # must be 2 to the nth power. (n = 3 to 15)

Flags :

S	Z	H	V	N	C
−	−	−	−	−	−

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: Decrements the IX register by cycling from 1280H to 12FFH.
MDEC4 80H,IX
When the IX register = 1284H, execution sets the IX register to 1280H. Further execution increments the IX register by 80H−4 and sets the IX register to 12FCH, since the IX register modulo 80H = 0.



MINC1 num, dst

< Modulo Increment 1 >

Operation : if (dst mod num) = (num−1) then dst ← dst− (num−1) else dst ← dst+1.

Description : When the modulo num of dst is num−1, decrements dst by num−1. Otherwise, increments dst by 1. Used to operate pointers for cyclic memory table .

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MINC1	#, r	

1	1	0	1	1	r		
0	0	1	1	1	0	0	0
#<7:0>−1							
#<15:8>							

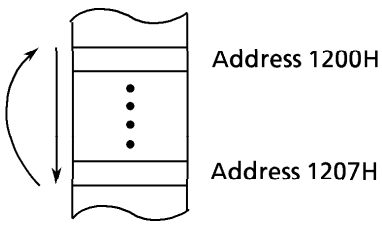
Note : The operand # must be 2 to the nth power. (n = 1 to 15)

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: Increments the IX register by cycling from 1200H to 1207H.
MINC1 8, IX
When the IX register = 1206H, execution sets the IX register to 1207H. Further execution decrements the IX register by 8−1 and sets the IX register to 1200H, since the IX register modulo 8 = 8−1.



MINC2 num, dst

< Modulo Increment 2 >

Operation : if (dst mod num) = (num−2) then dst ← dst− (num−2) else dst ← dst+2.

Description : When the modulo num of dst is num−2, decrements dst by num−2. Otherwise, increments dst by 2. Used to operate pointers for cyclic memory table.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MINC2	#, r	

1	1	0	1	1	r			
0	0	1	1	1	0	0	1	
#<7:0>−2								
#<15:8>								

Note : The operand # must be 2 to the nth power. (n = 2 to 15)

Flags :

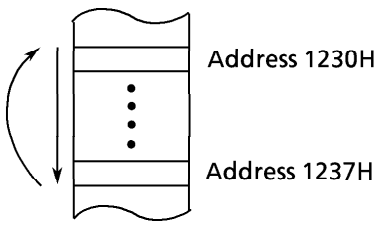
S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: Increments the IX register by cycling from 1230H to 1237H.

```
MINC2  8,IX
```

When the IX register = 1234H, execution sets the IX register to 1236H. Further execution decrements the IX register by 8−2 and sets the IX Register to 1230H, since the IX register modulo 8 = 8−2.



MINC4 num, dst

< Modulo Increment 4 >

Operation : if (dst mod num) = (num−4) then dst ← dst− (num−4) else dst ← dst+4.

Description : When the modulo num of dst is num−4, decrements dst by num−4. Otherwise, increments dst by 4. Used to operate pointers for cyclic memory table.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MINC4	#, r	

1	1	0	1	1	r			
0	0	1	1	1	0	1	0	
#<7:0>−4								
#<15:8>								

Note : The operand # must be 2 to the nth power. (n = 3 to 15)

Flags :

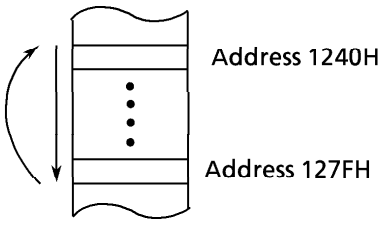
S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: Increments the IX register by cycling from 1240H to 127FH.

```
MINC4   40H,IX
```

When the IX register = 1278H, execution sets the IX register to 127CH. Further execution decrements the IX register by 40H−4 and sets the IX register to 1240H, since the IX register modulo 40H = 40H−4.



MIRR dst

< Mirror >

Operation : dst<MSB:LSB> ← dst<LSB : MSB>

Description : Mirror-exchanges the contents of dst using the bit pattern image.

Details :

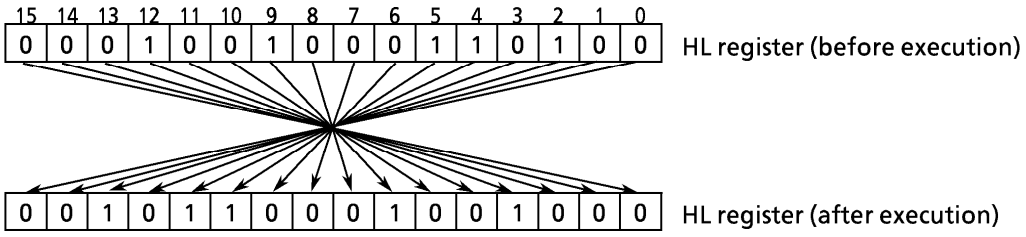
Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MIRR	r	1 1 0 1 1 r / 0 0 0 1 0 1 1 0

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: MIRR HL
When the HL register = 0001 0010 0011 0100B (binary), execution sets the HL register to 0010 1100 0100 1000B (binary).



MUL dst, src

< Multiply >

Operation : dst ← dst<lower half> × src (unsigned)

Description : Multiplies unsigned the contents of lower half of dst by those of src and loads the result to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	○	×	MUL	RR, r	1 1 0 z 1 r 0 1 0 0 0 R
○	○	×	MUL	rr, #	1 1 0 z 1 r 0 0 0 0 1 0 0 0 # <7:0> # <15:8>
○	○	×	MUL	RR, (mem)	1 m 0 z m m m m 0 1 0 0 0 R

Note : When the operation is in bytes, dst (word) ← dst (byte) × src (byte).
When the operation is in words, dst (long word) ← dst (word) × src (word).
Match coding of the operand dst with the size of the result.

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: MUL XIX, IY
When the IX register = 1234H and the IY register = 89ABH, execution multiplies unsigned the contents of the IX register by those of the IY register and sets the XIX register to 09C9FCBCH.

Note : "RR" for the MUL RR,r and MUL RR,(mem) instructions is as listed below:

Operation size in bytes
(16 bits ← 8 bits × 8 bits)

RR	Code R
WA	001
BC	011
DE	101
HL	111
IX	Specification not possible!
IY	
IZ	
SP	

Operation size in words
(32 bits ← 16 bits × 16 bits)

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

"rr" for the MUL rr,# instruction is as listed below.

Operation size in bytes
(16 bits ← 8 bits × 8 bits)

rr	Code r
WA	001
BC	011
DE	101
HL	111
IX	C7H : F0H
IY	C7H : F4H
IZ	C7H : F8H
SP	C7H : FCH
	1st byte 2nd byte

Note: Any other word registers can be specified in the same extension coding as those for IX to SP.

Operation size in words
(32 bits ← 16 bits × 16 bits)

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

Note: Any other long word registers can be specified in the extension coding.

MULA dst

< Multiply and Add >

Operation : dst ← dst + (XDE) × (XHL) , XHL ← XHL − 2

Description : Multiplies signed the memory data (16 bits) specified by the XDE register by the memory data (16 bits) specified by the XHL register . Adds the result (32 bits) to the contents of dst (32 bits) and loads the sum to dst (32 bits). Then, decrements the contents of the XHL register by 2.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
×	○	×	MULA	rr	1 1 0 1 1 r 0 0 0 1 1 0 0 1

Note : Match coding of the operand dst with the operation size (long word).

Flags :

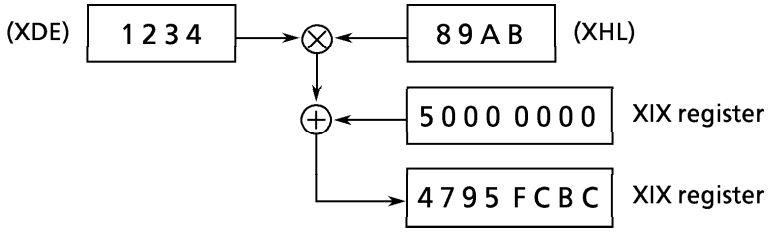
S	Z	H	V	N	C
*	*	–	*	–	–

S = MSB value of the result is set.
Z = 1 is set when the result is 0, otherwise 0.
H = No change.
V = 1 is set when an overflow occurs as a result, otherwise 0.
N = No change.
C = No change.

Execution example: MULA XIX

Under the following conditions, execution sets the XIX register to 4795FCBCH and the XHL register to 1FEH.

Conditions: XIX register = 50000000H
XDE register = 100H
XHL register = 200H
Memory data (word) at address 100H = 1234H
Memory data (word) at address 200H = 89ABH



MULS dst, src

< Multiply Signed >

Operation : dst ← dst<lower half> × src (signed)

Description : Multiplies signed the contents of the lower half of dst by those of src and loads the result to dst.

Details :

Size			Mnemonic		Code
Byte	Word	Long word			
○	○	×	MULS	RR, r	1 1 0 z 1 r 0 1 0 0 1 R
○	○	×	MULS	rr, #	1 1 0 z 1 r 0 0 0 0 1 0 0 1 # <7:0> # <15:8>
○	○	×	MULS	RR, (mem)	1 m 0 z m m m m 0 1 0 0 1 R

Note : When the operation is in bytes, dst(word) ← dst (byte) × src (byte).
When the operation is in words, dst (long word) ← dst (word) × src (word).
Match coding of the operand dst with the size of the result.

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change

Execution example: MULS XIX, IY
When the IX register = 1234H and the IY register = 89ABH, execution multiplies signed the contents of the IX register by those of the IY register and sets the XIX register to F795FCBCH.

Note : "RR" for the MULS RR,r and MULS RR,(mem) instructions is as listed below:

Operation size in bytes
(16 bits←8 bits × 8 bits)

RR	Code R
WA	001
BC	011
DE	101
HL	111
IX	Specification not possible!
IY	
IZ	
SP	

Operation size in words
(32 bits←16 bits × 16 bits)

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

"rr" for the MULS rr,# instruction is as listed below.

Operation size in bytes
(16 bits←8 bits × 8 bits)

rr	Code r
WA	001
BC	011
DE	101
HL	111
IX	C7H : F0H
IY	C7H : F4H
IZ	C7H : F8H
SP	C7H : FCH
	1st byte 2nd byte

*1 Any other word registers can be specified in the same extension coding as those for IX to SP.

Operation size in words
(32 bits←16 bits × 16 bits)

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

*2 Any other long word registers can be specified in the extension coding.

NEG dst

< Negate >

Operation : dst←0−dst

Description : Decrements 0 by the contents of dst and loads the result to dst. (Twos complement)

Details :

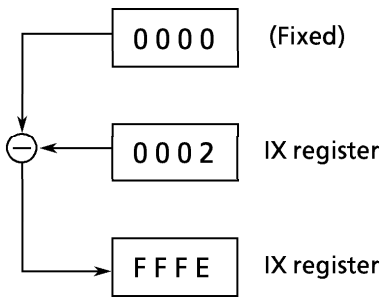
Size			Mnemonic		Code
Byte	Word	Long word			
○	○	×	NEG	r	1 1 0 z 1 r 0 0 0 0 0 1 1 1

Flags :

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

S = MSB value of the result is set.
Z = 1 is set when the result is 0, otherwise 0.
H = 1 is set when a borrow from bit 3 to bit 4 occurs as a result, otherwise 0.
V = 1 is set when an overflow occurs as a result, otherwise 0.
N = 1 is set.
C = 1 is set when a borrow from the MSB occurs as a result, otherwise 0.

Execution example: NEG IX
When the IX register = 0002H, execution sets the IX register to FFFEH.



NOP

<No Operation>

Operation : None.

Description : Does nothing but moves execution to the next instruction. The object code of this instruction is 00H.

Details :

Mnemonic	Code
NOP	0 0 0 0 0 0 0 0

Flags :

S	Z	H	V	N	C
–	–	–	–	–	–

S = No change
Z = No change
H = No change
V = No change
N = No change
C = No change