900, 900/L, 900/H, 900/L1, 900/H2 CPU Core Different Points

There are 5 type CPU core : ① 900, ② 900/L, ③ 900/H, ④ 900/L1, ⑤ 900/H2 in TLCS-900 series and they are different from following points.

CPU Different	900	900/L	900/H, 900/L1	900/H2
Address Bus	24-bit	←	←	←
Data Bus	16-bit	←	←	32-bit
Instruction Queue	4-byte	←	←	12-byte
Instruction Set	TLCS-900	Deleted instruction NORMAL MAX Added instruction MIN	Deleted instruction NORMAL MAX	Deleted instruction NORMAL MAX LDX
Code Fetch	Only when branch, CPU fetch branch destination code.	←	←	Even when not branch, CPU fetch branch destination code.
Micro DMA	4 channels	←	←	8 channels
Operation Mode	Normal mode, System mode	System mode	←	←
Register Mode	MIN mode, MAX mode, (MIN mode at reset)	MIN mode, MAX mode, (MAX mode at reset)	MAX mode	←
Interrupt	Restart formula	Vector formula	←	←
Normal Stack Pointer (XNSP)	exist	not exist	←	←
Interrupt Nesting Counter (INTNEST)	not exist	exist	←	←

Figure 1 CPU Different Points

1. Outline

The TLCS-900 series has an original Toshiba high-performance 16-bit CPU. Combining the CPU with various I/O function blocks (such as timers, serial I/Os, ADs) creates broad possibilities in application fields.

The TLCS-900 CPU, being 16-bit CPU, has a 32-bit/16-bit register bank configuration, therefore it is suitable as an embedded controller.

The TLCS-900 CPU features are as follows :

(1)TLCS-90 extended architecture

- Upward compatibility on mnemonic and register set levels

(2)General-purpose registers

- All 8 registers usable as accumulator

(3)Register bank system

- four 32-bit register banks

(4)16M-byte linear address space ; 9 types addressing modes

(5)Dynamic bus sizing system

- Can consist 8- / 16-bit external data bus together

(6)High reliability

- Supporting system mode and normal mode (900)
- Supporting only system mode (900/L, 900/H)

(7)Orthogonal instruction sets

- 8-/16-/32-bit data transfer/arithmetic instructions
- 16-bit multiplication/division
  16 × 16 to 32-bits (signed/unsigned)
  32 ÷ 16 to 16-bits ··· remainder 16-bits (signed/unsigned)
- Bit processing including bit arithmetic
- Supporting instruction for C compiler
- Filter calculations : multiplication-addition arithmetic, modulo increment instruction

(8)High-speed processing

- Minimum instruction execution time: 160 ns at 25 MHz
- Pipeline system with 4-byte instruction queue buffer
- 32-bit ALU

2. CPU Operating Modes

The 900/H has only system mode.
In system mode, there are no restrictions on using instructions or registers.
The CPU resources effective in system mode are as follows :

1) General-purpose registers
   - Four 32-bit general-purpose registers × 4 banks
   - Four 32-bit general-purpose registers (including system stack pointer : XSP)
2) Status register (SR)
3) Program counter (PC): 32 bits
4) Control register: parameter register for micro DMA, etc.
5) All CPU instructions
6) All built-in I/O registers
7) All built-in memories

3. Registers

3.1 Register Structure······ 16M-byte program area / 16M-byte data area

Figure 3.1.1 illustrates the format of registers.

Four 32-bit general-purpose registers × 4 banks
+
Four 32-bit general-purpose registers
+
32-bit program counter
+
Status register

Register mode changing

The <MAX> bit in status register (SR) is initialized to "1" and set to Maximum mode by resetting. The 900/H has only Maximum mode.

Stack Pointer

The stack pointer (SP) is provided for only System mode (XSP). The System stack pointer (XSP) is set to 100H by resetting.

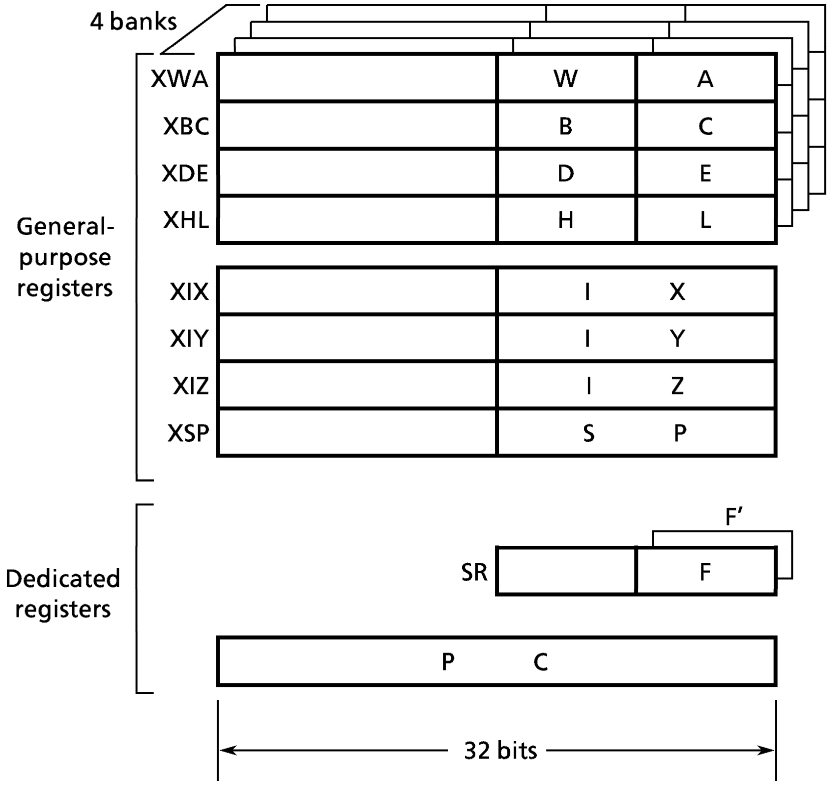


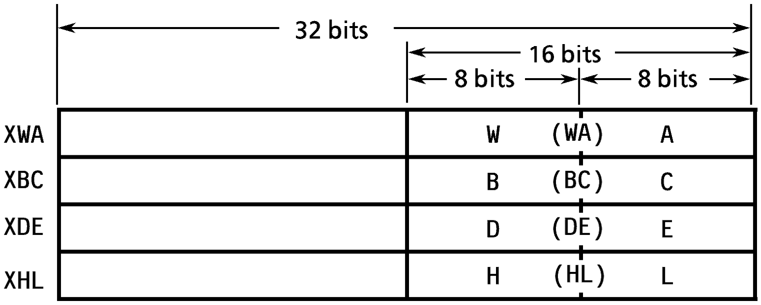
Figure 3.1.1 Register Format (16M-byte program area)

3.2 Register Details

3.2.1 General-purpose bank registers

In maximum mode, the following four 32-bit general-purpose registers consisting of 4 banks can be used. The register format in a bank is shown below.

Four 32-bit registers (XWA, XBC, XDE, and XHL) are general-purpose registers and can be used as an accumulators and index registers. They can also be used as 16-bit registers (WA, BC, DE, and HL), in which case, the lower 16 bits of the 32-bit registers are assigned.



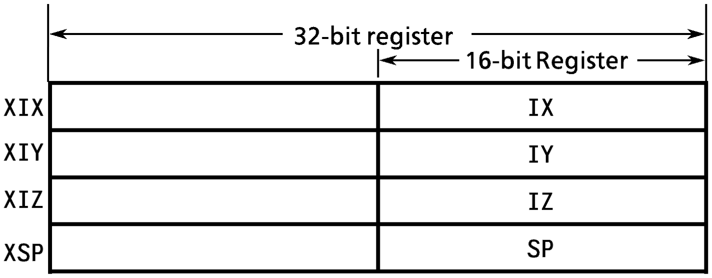
Note: Round brackets ( ) signify 16-bit registers.

16-bit registers can be used as accumulators, index registers in index addressing mode, and displacement registers. They can also be used as two 8-bit general-purpose registers (W, A, B, C, D, E, H, and L) to function for example as accumulators.

3.2.2 32-bit General-purpose Registers

The TLCS-900 series has four 32-bit general-purpose registers (XIX, XIY, XIZ, and XSP). The register format is shown below.

These registers can also be used as accumulators, index registers, and displacement registers. They can be used either as 16-bit, or 8-bit registers. Names when registers are used as 8-bit registers are listed later.



Stack Pointer

The XSP register is utilized for stack pointer. It is used when the interrupt is occured or “CALL”, “RET” instruction are executed. The stack pointer (XSP) is set to 100H by resetting.

3.2.3 Status Register (SR)

The status register contains flags indicating the status (operating mode, register format, etc.) of the CPU and operation results. This register consists of two parts. The upper byte of the status register (bits 8 to 15) indicates the CPU status. The lower byte (bits 0 to 7) are referred to as the flag register (F). This indicates the status of the operation result. The TLCS-900 series has two flag registers (F and F'). They can be switched using the EX instruction.

(1) Upper Byte of Status Register

15	14	13	12	11	10	9	8
SYSM	IFF2	IFF1	IFF0	MAX	RFP2	RFP1	RFP0

① SYSM (SYStem Mode)

Indicates the CPU operating mode, system or normal. 900/H has only system mode.

Initialized to 1 (system mode) by reset.

0	Normal mode
1	System mode (900/H has only this mode.)

② IFF2 to IFF0 (Interrupt mask Flip-Flop2 to 0)

Mask registers with interrupt levels from 1 to 7. Level 7 has the highest priority. Any value can be set using the EI instruction.

Initialized to 111 by reset.

000	Enables interrupts with level 1 or higher.	Same
001	Enables interrupts with level 1 or higher.	
010	Enables interrupts with level 2 or higher.	
011	Enables interrupts with level 3 or higher.	
100	Enables interrupts with level 4 or higher.	
101	Enables interrupts with level 5 or higher.	
110	Enables interrupts with level 6 or higher.	
111	Enables interrupts with level 7 only (non-maskable interrupt).	

When an interrupt is received, the mask register sets a value higher by 1 than the interrupt level received. When an interrupt with level 7 is received, 111 is set. Unlike with the TLCS-90 series, the EI instruction becomes effective immediately after execution.

③ MAX ($\overline{\text{MINimum}}$ / MAXimum)

Bit used to specify the register mode which determines the sizes of the register banks and the program counter.

0	Minimum mode
1	Maximum mode (900/H has only this mode.)

Initialized to "1" (maximum mode) for 900/H by reset. 900/H does not have Minimum mode. Don't set to "0".

④ RFP2 to RFP0 (Register File Pointer2 to 0)

Indicates the number of register file (register bank) currently being used. Initialized to 000 by reset.

The values in these registers can be operated on using the following three instructions. RFP2 is fixed to 0 in maximum mode. It remains 0 even if an attempt to change it to 1 using following instructions.

- LDF imm ; RFP←imm (0 to 3) (160 ns at 25 MHz)
- INCF ; RFP←RFP + 1 (160 ns at 25 MHz)
- DECF ; RFP←RFP − 1 (160 ns at 25 MHz)

(2) Flag Register, F

7	6	5	4	3	2	1	0	
S	Z	"0"	H	"0"	V	N	C	: R/W

① S (Sign flag)

"1" is set when the operation result is negative, "0" when positive.
(The value of the most significant bit of the operation result is copied.)

② Z (Zero flag)

"1" is set when the operation result is zero, otherwise "0".

③ H (Half carry flag)

"1" is set when a carry or borrow from bit 3 to bit 4 occurs as a result of the operation, otherwise "0". With a 32-bit operation instruction, an undefined value is set.

④ V (Parity/over-flow flag)

Indicates either parity or overflow, depending on the operation type.
Parity (P): “0” is set when the number of bits set to 1 is odd, “1” when even.
An undefined value is set with a 32-bit operation instruction.
Overflow (V): “0” is set if no overflow, if overflow “1”.

⑤ N (Negative)

ADD/SUB flag
“0” is set after an addition instruction such as ADD is executed, “1” after a subtraction instruction such as SUB.
Used when executing the DAA (decimal addition adjust accumulator) instruction.

⑥ C (Carry flag)

“1” is set when a carry or borrow occurs, otherwise “0”.

Read and write process of status register

Read from bits 0 to 15	① PUSH SR POP dst
Write to bits 0 to 15	① POP SR
Only bit 15 <SYSM>	“1” is always set, because 900/H CPU has only system mode.
Only bits 14 to 12 <IFF2 to 0>	① EI num A value of “num” is written.
Only bit 11 <MAX>	“1” is always set, because 900/H CPU has only maximum mode.
Only bits 10 to 8 <RFP2 to 0>	① LDF imm ② INCF ③ DECF
Only bits 7 to 0	① PUSH F/POP F ② EX F, F’ ③ A flag is set indirectly by executing arithmetic instructions etc.

3.2.4 Program Counter (PC)

The program counter is a pointer indicating the memory address to be executed next.

In maximum mode, the program counter consists of 32 bits. The size of the program area depends on the number of the address pins that the product has. With 24 address pins (A0 to A23), a maximum program area of 16M bytes can be accessed as a linear address space. In this case, the upper 8 bits of the program counter (bits 24 to 31) are ignored.

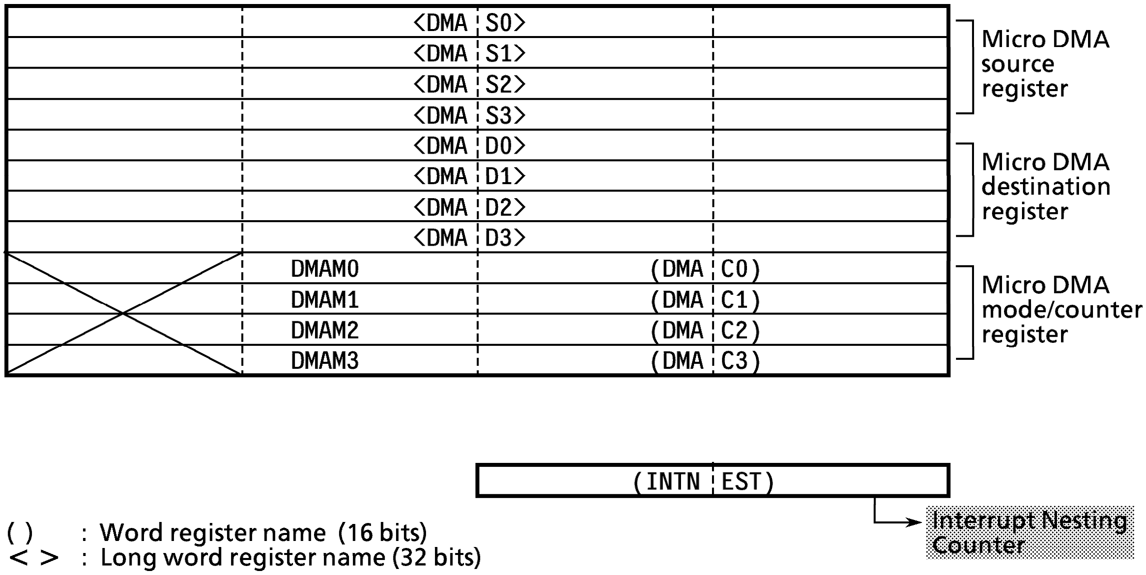
PC after reset

The 900/H reads a value of a reset vector from a vector base address by reset and sets the value into a program counter. Then, program after the vector specified by the program counter are executed.

3.2.5 Control registers (CR)

The control registers consist of registers used to control micro DMA operation and an interrupt nesting counter. Control registers can be accessed by using the LDC instruction.

Control registers are illustrated below.



( ) : Word register name (16 bits)
< > : Long word register name (32 bits)

For micro DMA, refer to "Chapter 4 TLCS-900/H LSI Devices".

3.3 Register Bank Switching

Register banks are classified into the following three types.

Current bank registers
Previous bank registers
Absolute bank registers

The current bank is indicated by the register file pointer, <RFP>, (status register bits 8 to 10). The registers in the current bank are used as general-purpose registers, as described in the previous section. By changing the contents of the <RFP>, another register bank becomes the current register bank.

The previous bank is indicated by the value obtained by subtracting 1 from the <RFP>. For example, if the current bank is bank 3, bank 2 is the previous bank. The names of registers in the previous bank are indicated with a dash (WA', BC', DE', HL'). The EX instruction (EX A,A') is used to switch between current and previous banks.

All bank registers, including the current and previous ones, have a numerical value (absolute bank number) to indicate the bank. With a register name which includes a numerical value such as RW0, RA0, etc., all bank registers can be used. These registers (that is, all registers) are called absolute bank registers.

The TLCS-900 series CPU is designed to perform optimally when the current bank registers are operated as the working registers. In other words, if the CPU uses other bank registers, its performance degrades somewhat. In order to obtain maximum CPU efficiency, the TLCS-900 series has a function which easily switches register banks.

The bank switching function provides the following advantages:

- Optimum CPU operating efficiency
- Reduced programming size (Object codes)
- Higher response speed and reduced programming size when used as a context switch for an interrupt service routine.

Bank switching is performed by the instructions listed below.

LDF imm	:	Sets the contents of the immediate value in <RFP>. imm: 0 to 3
INCF	:	Increments <RFP> by 1.
DECF	:	Decrements <RFP> by 1.

The immediate values used by the LDF instruction are from 0 to 3. If a carry or borrow occurs when the INCF or DECF instruction is executed, it is ignored. The value of the <RFP> rotates. For example, if the INCF instruction is executed with bank 3, the result is bank 0. If the DECF instruction is executed with bank 0, the result is bank 3. Note that careless execution of the INCF or DECF instruction may destroy the contents of the register bank.

- Example of Register Bank Usage

The TLCS-900 series registers are formatted in banks. Banks can be used for processing objectives or interrupt levels. Two examples are given below.

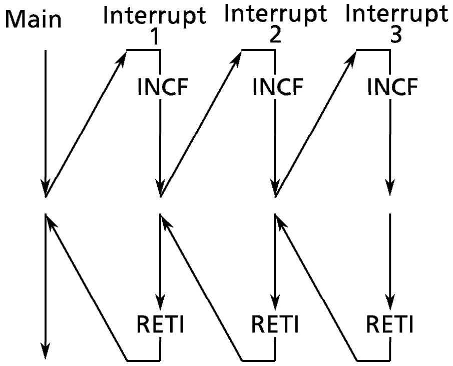
<Example 1> When assigning register banks to interrupt processing routines.

Register bank 0 = Used for the main program and interrupt processing other than that shown below.
Register bank 1 = Used for processing INT0 .
Register bank 2 = Used for processing timer 0.
Register bank 3 = Used for processing timer 1.

For example, if a timer 1 interrupt occurs during main program execution, processing jumps to a subroutine as follows. PUSH/POP processing for the register is unnecessary.

```
LDF  3      ;  Sets register bank to 3.  0.16 μs (at 25 MHz)
  :
  :
RETI        ;  Returns to previous status including <RFP>.
                0.96 μs (at 25 MHz)
```

<Example 2> When assigning register banks to their appropriate interrupt level nesting.



Note 1 : In the above example, when interrupt nesting exceeds the number of register banks (4), the <RFP> becomes 000 and the contents of register bank 0 are destroyed.

Note 2 : The INCF instruction is used to execute <RFP>←<RFP> + 1.
0.16 μs (at 25 MHz)

3.4 Accessing General-purpose Registers

The register access code is formatted in a varied code length on byte basis. The current bank registers can be accessed by the shortest code length. All general-purpose registers can be accessed by an instruction code which is 1 byte longer. General-purpose registers are as follows.

① General-purpose registers in current bank

QW	(Q	WA)	QA	<X	WA >	W	(W	A)	A
QB	(Q	BC)	QC	<X	BC >	B	(B	C)	C
QD	(Q	DE)	QE	<X	DE >	D	(D	E)	E
QH	(Q	HL)	QL	<X	HL >	H	(H	L)	L

( ) : Word register name (16 bits)
< > : Long word register name (32 bits)

② General-purpose registers in previous bank

QW’	(Q	WA’)	QA’	<X	WA’>	W’	(W	A’)	A’
QB’	(Q	BC’)	QC’	<X	BC’>	B’	(B	C’)	C’
QD’	(Q	DE’)	QE’	<X	DE’>	D’	(D	E’)	E’
QH’	(Q	HL’)	QL’	<X	HL’>	H’	(H	L’)	L’

③ 32-bit general-purpose registers

QIXH	(Q	IX)	QIXL	<X	IX>	IXH	(I	X)	IXL
QIYH	(Q	IY)	QIYL	<X	IY>	IYH	(I	Y)	IYL
QIZH	(Q	IZ)	QIZL	<X	IZ>	IZH	(I	Z)	IZL
QSPH	(Q	SP)	QSPL	<X	SP>	SPH	(S	P)	SPL

④ Absolute bank registers

QW0	(QWA	0)	QA0	<XWA	0>	RW0	(RWA	0)	RA0	Bank0
QB0	(QBC	0)	QC0	<XBC	0>	RB0	(RBC	0)	RC0	
QD0	(QDE	0)	QE0	<XDE	0>	RD0	(RDE	0)	RE0	
QH0	(QHL	0)	QL0	<XHL	0>	RH0	(RHL	0)	RL0	
QW1	(QWA	1)	QA1	<XWA	1>	RW1	(RWA	1)	RA1	Bank1
QB1	(QBC	1)	QC1	<XBC	1>	RB1	(RBC	1)	RC1	
QD1	(QDE	1)	QE1	<XDE	1>	RD1	(RDE	1)	RE1	
QH1	(QHL	1)	QL1	<XHL	1>	RH1	(RHL	1)	RL1	
QW2	(QWA	2)	QA2	<XWA	2>	RW2	(RWA	2)	RA2	Bank2
QB2	(QBC	2)	QC2	<XBC	2>	RB2	(RBC	2)	RC2	
QD2	(QDE	2)	QE2	<XDE	2>	RD2	(RDE	2)	RE2	
QH2	(QHL	2)	QL2	<XHL	2>	RH2	(RHL	2)	RL2	
QW3	(QWA	3)	QA3	<XWA	3>	RW3	(RWA	3)	RA3	Bank3
QB3	(QBC	3)	QC3	<XBC	3>	RB3	(RBC	3)	RC3	
QD3	(QDE	3)	QE3	<XDE	3>	RD3	(RDE	3)	RE3	
QH3	(QHL	3)	QL3	<XHL	3>	RH3	(RHL	3)	RL3	

( ) : Word register name (16 bits)
< > : Long word register name (32 bits)

4. Addressing Modes

The TLCS-900 series has nine addressing modes. These are combined with most instructions to improve CPU processing capabilities.

TLCS-900 series addressing modes are listed below. They cover the entire TLCS-90 addressing modes.

No.	Addressing mode	Description
1.	Register	reg8 reg16 reg32
2.	Immediate	n8 n16 n32
3.	Register indirect	(reg)
4.	Register indirect pre-decrement	(– reg)
5.	Register indirect post-increment	(reg +)
6.	Index	(reg + d8) (reg + d16)
7.	Register index	(reg + reg8) (reg + reg16)
8.	Absolute (Direct addressing mode)	(n8) (n16) (n24)
9.	Relative	(PC + d8) (PC + d16)

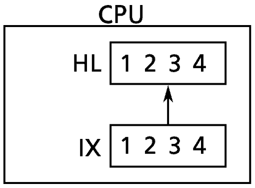
reg 8	: All 8-bit registers such as W, A, B, C, D, E, H, L, etc.
reg 16	: All 16-bit registers such as WA, BC, DE, HL, IX, IY, IZ, SP, etc.
reg 32	: All 32-bit registers such as XWA, WBC, XDE, XHL, XIX, XIY, XIZ, XSP, etc.
reg	: All 32-bit registers such as XWA, WBC, XDE, XHL, XIX, XIY, XIZ, XSP, etc.
d8	: 8-bit displacement (−80H to +7FH)
d16	: 16-bit displacement (−8000H to +7FFFH)
n8	: 8-bit constant (00H to FFH)
n16	: 16-bit constant (0000H to FFFFH)
n32	: 32-bit constant (00000000H to FFFFFFFFH)

Note: Relative addressing mode can only be used with the following instructions:
LDAR, JR, JRL, DJNZ, and CALR

(1) Register Addressing Mode

In this mode, the operand is the specified register.

Example: LD HL,IX

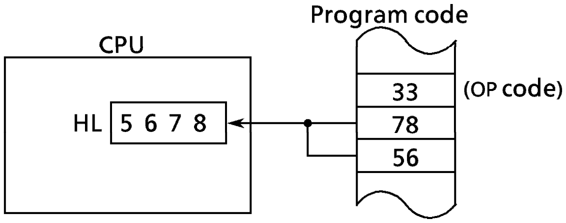


The IX register contents, 1234H, are loaded to the HL register.

(2) Immediate Addressing Mode

In this mode, the operand is in the instruction code.

Example: LD HL,5678H

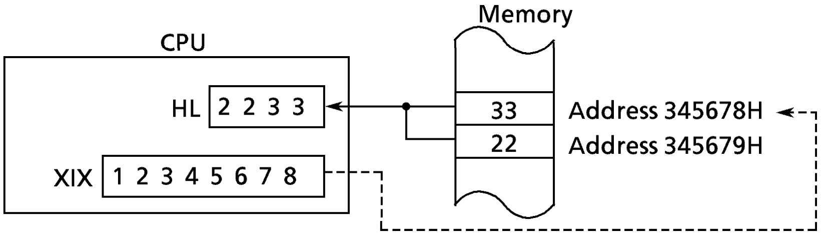


The immediate data, 5678H, is loaded to the HL register.

(3) Register Indirect Addressing Mode

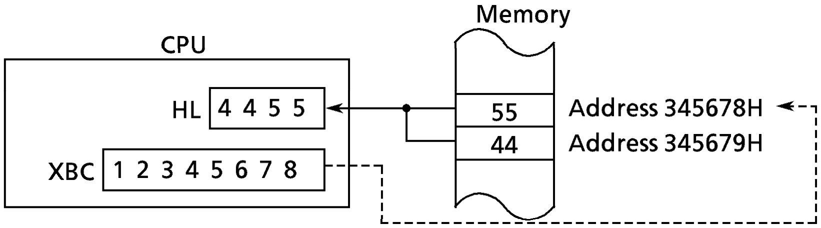
In this mode, the operand is the memory address specified by the contents of the register.

Example 1: LD, HL, (XIX)



Memory data, 2233H, at address 345678H is loaded to the HL register.

Example 2: LD HL,(XBC)

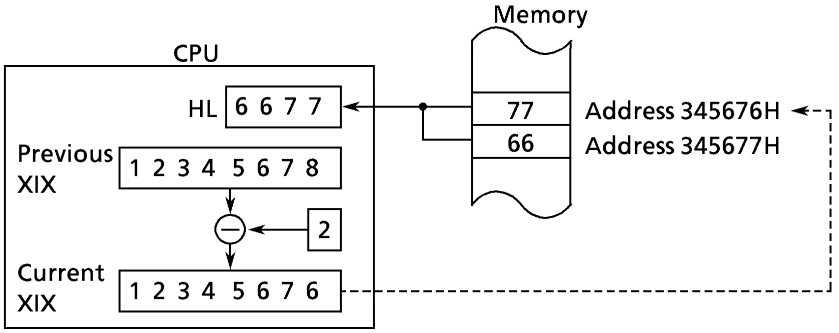


If a bank register (XWA, XBC, XDE, or XHL) is used for addressing, the values of bits 0 to 23 are output to the address bus.

(4) Register Indirect Pre-decrement Addressing Mode

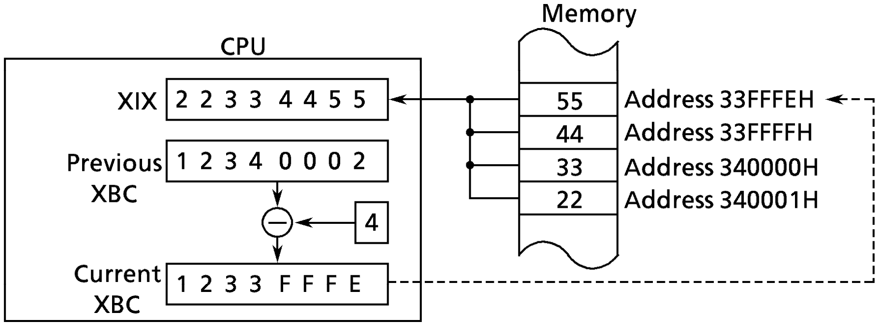
In this mode, the contents of the register is decremented by the pre-decrement values. In this case, the operand is the memory address specified by the decremented register.

Example 1: LD HL, (– XIX)



The pre-decrement values are as follows:
When the size of the operand is one byte (8 bits) : −1
When the size of the operand is one word (16 bits) : −2
When the size of the operand is one long word (32 bits) : −4

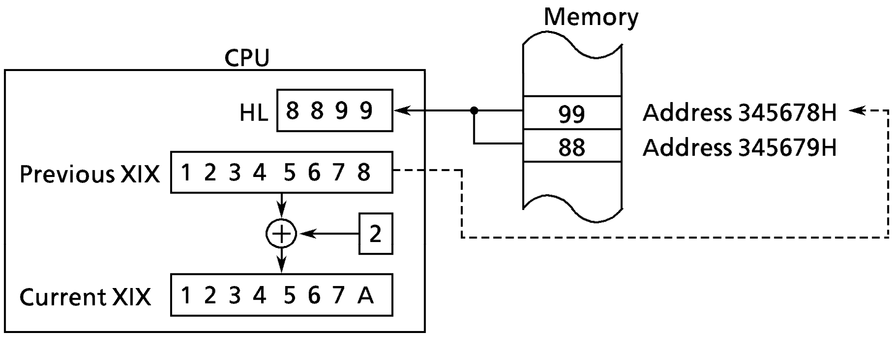
Example 2: LD XIX,(-XBC)



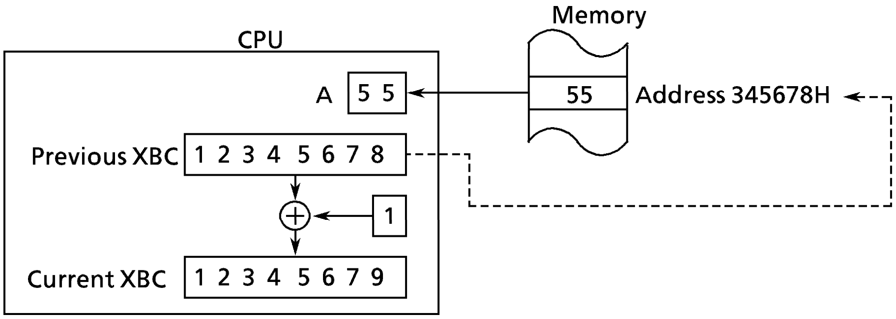
(5) Register Indirect Post-increment Addressing Mode

In this mode, the operand is the memory address specified by the contents of the register. After the operation, the contents of the register are incremented by the size of the operand.

Example 1: LD HL,(XIX +)



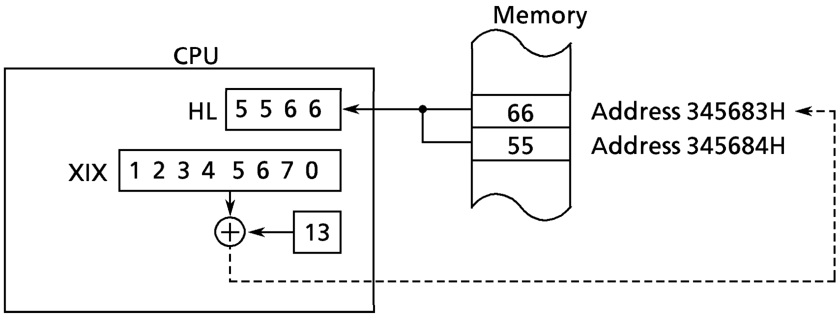
Example 2: LD A,(XBC +)



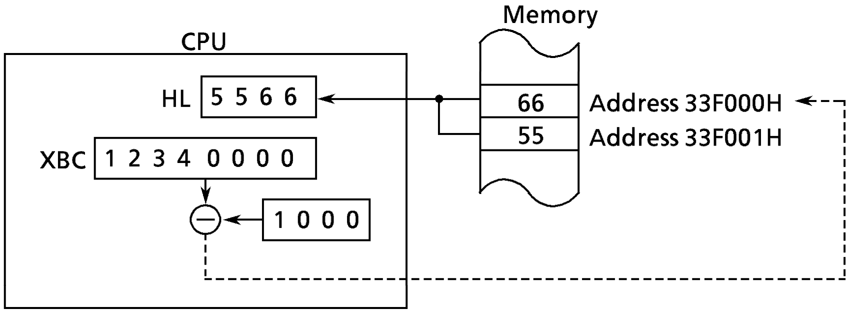
(6) Index Addressing Mode

In this mode, the operand is the memory address obtained by adding the contents of the specified register to the 8- or 16-bit displacement value in the instruction code.

Example 1: LD HL,(XIX + 13H)



Example 2: LD HL,(XBC-1000H)

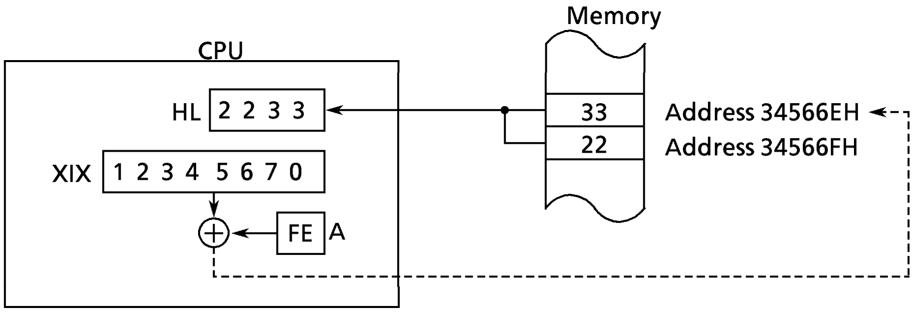


The displacement values range from −8000H to +7FFFH.

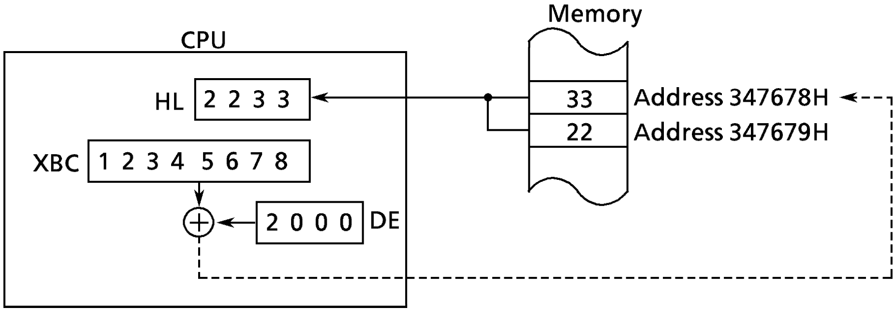
(7) Register Index Addressing Mode

The operand is the memory address obtained by adding the contents of the register specified to the base and the register specified to the displacement (signed 8- or 16-bit integer).

Example 1: LD HL,(XIX + A)



Example 2: LD HL,(XBC + DE)



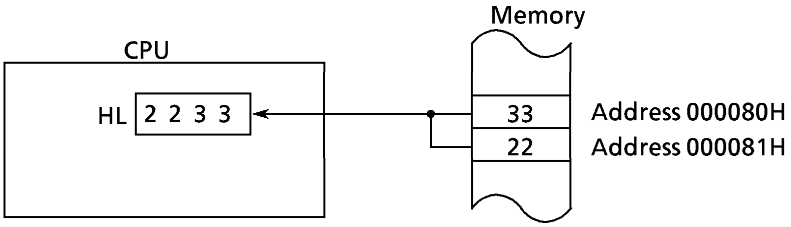
The range of displacement is :
−80H ~ 7FH in case of 8-bit and −8000H ~ +7FFFH in case of 16-bit.

(8) Absolute Addressing Mode

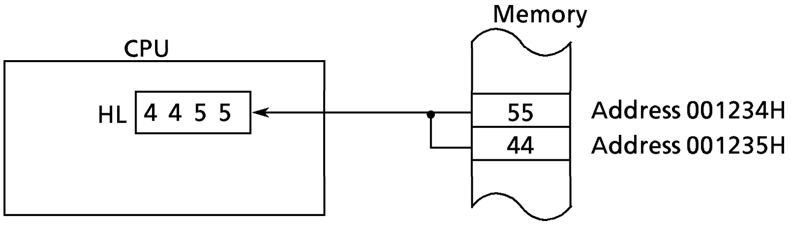
In this mode, the operand is the memory address specified by 1 to 3 bytes in the instruction code. Addresses 000000H to 0000FFH can be specified by 1 byte. Addresses 000000H to 00FFFFH can be specified by 2 bytes. Addresses 000000H to FFFFFFH can be specified by 3 bytes.

In this mode, addressing to 256-byte area (0H to FFH) which can be specified by 1 byte is called the direct addressing mode. In the direct addressing mode, a program memory area and execution time can be cut down.

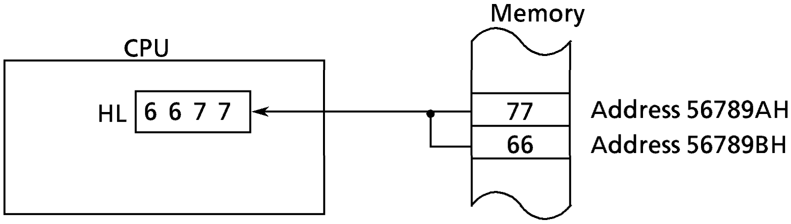
Example 1: LD HL,(80H)



Example 2: LD HL,(1234H)



Example 3: LD HL,(56789AH)



(9) Relative Addressing Mode

In this mode, the operand is the memory address obtained by adding the 8- or 16-bit displacement value to the address where the instruction code being executed is located.

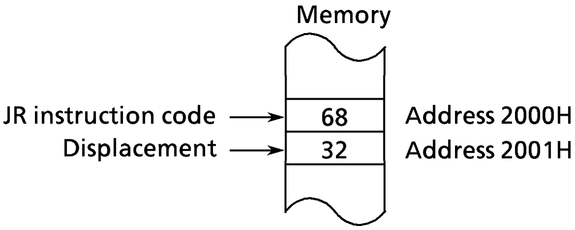
In this mode, only the following five instructions can be used.

LDAR	R, $+4+d16
JR	cc, $+2+d8
JRL	cc, $+3+d16
CALR	$+3+d16
DJNZ	r, $+3+d8

($: start address of instruction code)

In calculating the displacement object code value, the adjustment value (+2 to +4) depends on the instruction type.

Example 1: JR 2034H



In the above example, the displacement object code value is:

$$2034H - (2000H + 2) = 32H.$$