微型计算机原理及其应用

中　册

目　录

…… 么字

…制数。

第四章　微处理器指令系统和汇编语言程序设计

第一节　微处理器指令的基本格式

一、概　述

如第一章所述，当用计算机计算某个问题时，首先要把计算步骤以指令的形式提供给计算机，这样一串指令就叫做解决这个问题的程序。编制程序的过程叫做程序设计。

一台计算机中，究竟有多少条指令，各条指令实现什么操作，这是在计算机的设计阶段就已被规定下来的。所规定的全部指令就构成了这台机器的“指令系统”。指令系统是随机器的不同而不同的。

在计算机中，指令是以二进制代码形式表示的，如 Z80 指令系统中，有一条指令

0 0 1 1 1 1 1 0

0 1 0 1 0 1 0 1

是把二进制数 01010101 送到累加器 A 中。

为了简便起见，在书写指令时，也可用十六进制形式来表示。如上述指令可以写成

3EH*

55 H

为了直观地理解指令的含义，人们往往采用由字母和其它符号组成的“助记符”来表示指令，例如上述指令可写成

LD A, 55H

我们把用二进制代码表示的指令称做机器指令或机器语言，把用机器语言编写的程序称为机器语言程序，这种程序是机器能够直接执行的程序，称之为目标程序。而把采用助记符表示的指令称之为符号指令或汇编语言，用汇编语言编写的程序叫做汇编语言源程序。汇编语言源程序到机器语言目标程序的转换工作一般是由驻留在计算机内部的汇编程序进行的。但是，某些单板微型机或专用微型机并不配有汇编程序，在这种情况下，转换工作可以靠手工进行。手工转换过程叫做手工汇编。但手工汇编是一项繁杂而极易出错的工作。因此在有条件的情况下，可以在别的微机系统或开发系统上，利用它们拥有的汇编程序来产生目标程序，并用键盘输入或插入写有目标程序的 EPROM 芯片到单板微型机或专用微型机中去运行。

二、微处理器指令的基本格式

1. 指令的组成部分

---

* 我们用数字后面加字母 H 表示十六进制数，以字母 B 表示二进制数，以字母 Q 表示八进制数，以 D 或不加字母表示十进制数。

一条完整的指令应包括五个域：一个操作码域和四个地址域。四个地址域是两个操作数域、操作结果域以及下一条所要执行的指令的地址域。

(1) 操作码域

指令中指示操作命令种类的部分称为操作码域，常用的操作码（以助记符形式表示）有：相加（ADD）、相减（SUB）、数据传送（LD），跳转（JP）等等。

(2) 操作数域

操作数域存放操作数据或存放操作数所在的地址。有些操作需要两个操作数（如相加，相减）；有些操作只要一个操作数（如移位、求反等操作）。

(3) 结果域

该部分用以指示操作结果的存放地址。

(4) 下条指令地址域

这部分指示下一条所要执行的指令的存放地址。

因此，具有五个域的四地址指令形式为：

Q	A_1	A_2	A_3	A_4

这里，Q 表示操作码，A 表示地址域。

一般，下一条所要执行的指令地址由计算机的程序计数器（PC）保存。这样，我们可得到三地址的指令形式为：

Q	A_1	A_2	A_3

为了减少指令的长度，计算机往往把操作结果直接存放到第二操作数域指出的地址中，而第二个操作数不再被保存。这样，就得到一个二地址的指令形式：

Q	A_1	A_2

在有些微处理器中，由于字长较短，从而采用了单地址的指令形式：

Q	A_1

这种指令形式往往是已约定参加操作的另一个操作数保存在一个指定的寄存器中，这个寄存器往往是累加器。

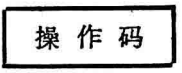
实际上，操作数并不一定就是在存贮器中，也可以放在微处理器内部的寄存器堆中。假如寄存器堆中包括 8 个寄存器，那么只要有3个二进制数位就可以指出它们的地址。因此，尽管很多微处理器的字长很短，它们还是可以使用包括两个寄存器操作数地址的二地址指令。微处理器的指令系统中，常常遇到单地址指令或二地址指令。

2. 多字节指令

在微处理器中，常把一条长指令分成两段、三段甚至四段来存放和处理，每段长度与微处理器字长相一致。对于八位微处理器，机器字长为 8 位，因此，按照指令和长度可分成一字节、双字节、三字节指令和四字节指令，其中一个字节的指令称为单字节指令。多个字节的指令称为多字节指令。

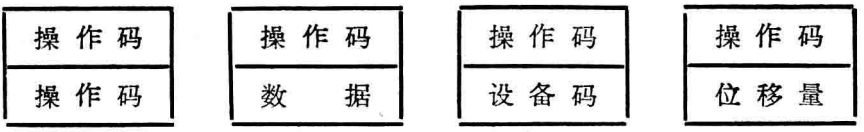
Z80 微处理器指令系统中，有单字节指令、双字节指令、三字节指令和四字节指令，它们的形式如下：

1） 单字节指令

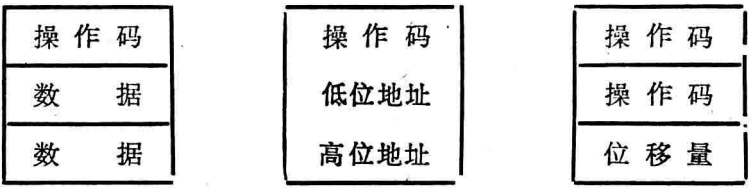


2） 双字节指令

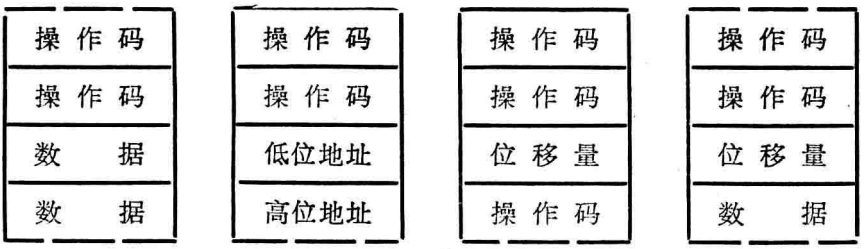
它有四种形式



3） 三字节指令



4） 四字节指令



上述指令形式中有关术语解释如下：

数据：是指本条指令所需要的直接操作数字节。它可以是二进制代码，二——十进制码或ASCII 码等。

设备码：是指输入输出设备中某个特定端口(Port)的地址号。所以它又可称为端口地址。

位移量：它出现在变址指令或相对转移指令中，是对变址寄存器 IX 和 IY 或程序计数器（PC）的修正量。它是用补码表示的。

高位地址和低位地址：分别是与指令操作有关的存贮器地址的高八位值和低八位值。当它们存放在存贮器中时，总是低位地址值在高位地址值之前。

第二节 微处理器指令的寻址方式

寻址方式是指指令存取操作数的方法。通常操作数可以直接跟在指令的操作码之后，成为指令的一部分；也可以在指令中包含某个代码信息，该信息直接或间接地告诉计算机操作数所在的地方。指令给出操作数或操作数地址的方法越多样，计算机的功能越强，灵活性越大。

Z80 微处理器共有十种寻址方式。因此，和其它八位微处理器相比，它在性能上占有很大的优势。

下面将详细介绍Z80微处理器的寻址方式。

一、立即寻址（Immediate Addressing）

立即寻址指出操作数本身就是指令中的一个字节。这个数据字节跟在指令的操作码之后，常称为立即数。它的指令格式如下：

操作码	一或二个字节
数　据	立即数

D_7　　D_0

例如，指令LD C,03H。这条指令的功能是把立即数03H送入寄存器C。它的代码结构为：

0EH	第一字节（操作码）
03H	第二字节（立即数）

二、立即扩展寻址（Immediate Extended Addrssing）

这种寻址方式中，指令中的立即数为两个字节，即为16位的操作数，指令格式为：

操作码	一或二个字节	
数　据	低位字节	16位立即数
数　据	高位字节	

D_7　　D_0

例如，指令LD BC，2000H，其功能是把立即数2000H送入寄存器对BC中，也即将立即数的高位字节20H送入B寄存器，低位字节00H送入C寄存器。这条指令的代码格式为：

01H	第一字节（操作码）	
00H	第二字节（低位字节）	16位立即数
20H	第三字节（高位字节）	

立即或立即扩展寻址经常用于给寄存器(或寄存器对)或计数器置初值，或者在计算中处理一批常数。在执行这类指令时，因为操作数已包含在指令中，不用额外花时间去寻找，所以指令的执行时间较短。但由于数据是固定的，数据传送的地方也是固定的，在使用时不够灵活。

三、寄存器寻址（Register Addressing）

在这种寻址方式中，操作数是在指令规定的寄存器中。

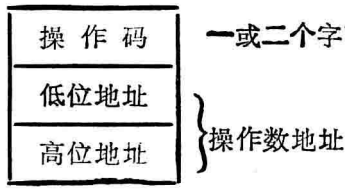
例如，指令LD C,B是把CPU中寄存器B的内容送入寄存器C中，指令中规定的操作

数地址是寄存器 B 和 C。

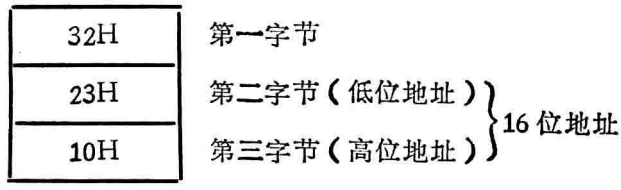
由于指令中规定的寄存器就是微处理器中通用寄存器组中的寄存器，因此采用这种寻址方式可以不必访问存贮器。这样的指令字长较短，执行时间较快。

四、扩展寻址（Extended Addressing）或直接寻址（Direct Addressing）

这种寻址方式在指令中规定了操作数所在存贮单元的真正地址（常称有效地址）。这个地址在指令中占两个字节，低位地址字节在高位地址字节之前。这类指令的格式如下：



例如，指令 LD A，（1023H）是把地址为 1023H 的存贮单元的内容送入累加器 A。其代码结构为：



五、寄存器间接寻址（Register Indirect Addrssing）

在这种寻址方式的指令中，规定某一寄存器对的内容作为操作数的地址。这和上面介绍的寄存器寻址方式是不同的，在寄存器寻址方式中，指令中规定的寄存器中直接存放着操作数。而在寄存器间接寻址方式中，指令中规定的寄存器对中存放的是操作数的存贮单元地址，而不是操作数本身。因为这种寻址没有直接在指令中给出操作数地址，只是间接地给出了存放操作数地址的寄存器对的号码，所以称之为寄存器间接寻址。

例如，指令 LD A，(HL) 表示把寄存器对 HL 中所指示的存贮单元的内容送入累加器 A。设寄存器对 HL 的内容为 2000H，而地址为 2000H 的存贮单元存放着 7EH，那么执行此指令后，累加器中的内容也是 7EH。

这种寻址方式的优点是指令字长短，执行速度快，可以简单地实现存贮器存取。

六、变址寻址（Indexed Adderssing）

在变址寻址方式中，指令中指定的变址寄存器的内容和指令中给定的位移量相加，所得到的结果作为操作数的存贮单元的地址。Z80 有两个专用的 16 位变址寄存器，其助记符分别为 IX 和 IY，它们主要用于变址寻址。

例如，指令 LD A，(IX+d) 表示把变址寄存器 IX 中的内容和位移量 d 相加后所确定的存贮单元的内容送到累加器 A 中。设 IX 中的内容为 1000H，位移量 d 为 02H，1002H 号单元的内容为 38H，那么执行此指令后，累加器 A 的内容也为 38H。

这种寻址方式中，位移量 d 是一个补码表示的数，在指令中占一个字节，其范围为 −128—+127。在进行变址寄存器内容和位移量相加时，不改变变址寄存器的内容。

七、修改零页寻址（Modified Page Zero Addressing）

Z80有一类单字节特殊指令RST P，叫做重新启动指令。它的代码结构为：

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	1	N	N	N	1	1	1

其中NNN可以构成000B—111B 8个数，根据这8个数，指令可以调用零页面（单元0001—00FFH为零页面）中有关入口处的子程序。这8个数规定的入口地址如表4—1所示。这种寻址方式主要应用于形成中断矢量。（详见第六章）

表4—1 RST P 指令入口地址表

N	入口地址	P
0	0000H	00H
1	0008H	08H
2	0010H	10H
3	0018H	18H
4	0020H	20H
5	0028H	28H
6	0030H	30H
7	0038H	38H

八、相对寻址（Relative Addressing）

Z80只在转移类指令中应用这种寻址方式，它以程序计数器PC的内容为基址，加上指令中给出的位移量作为程序的转移地址。这类指令都是二字节指令。

假设有一条相对转移指令JR SRCH

而且假设SRCH是表示某一存贮单元地址的标号，而这个存贮单元位于存贮器中离指令JR的操作码字节有0BH个字节的地方，那么这条指令的执行情况如图4—1所示。

因为当执行JR这条二字节指令时，程序计数器PC将指向下一条指令的第一字节，所以指令代码中的位移量为09H。这时的PC的内容加上09H就是程序所需转移的地址SRCH。

九、位寻址（Bit Addressing）

采用这种寻址方式的指令能对任一通用寄存器的任一位或任一存贮单元的任一位进行操作。这些操作包括测试（检查该位是否为零）、置位（将该位置"1"）、复位（将该位置"0"）。

例如，指令SET 3，A表示把累加器A的第3位置"1"。

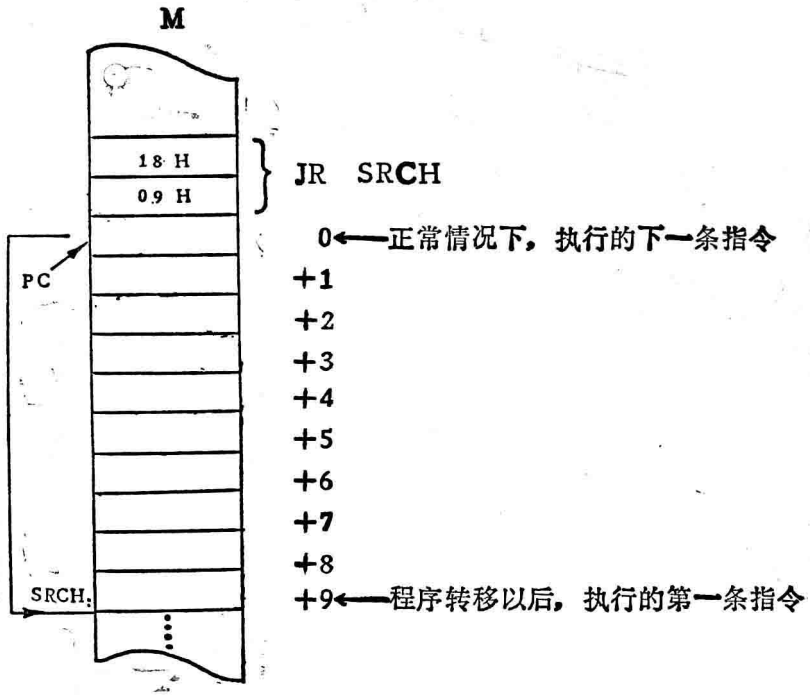
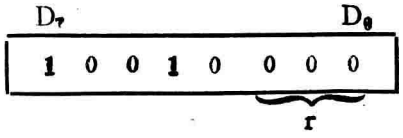


图4—1 相对寻址示意图

十、隐含寻址（Implied Addressing）

在采用这种寻址方式的指令中，有一个操作数的地址没有直接给出，但却隐含地指出以某一寄存器（通常是累加器A）中的内容作为一个操作数。Z80的算术逻辑指令就属于这类指令。它隐含累加器A中存放着一个操作数。

例如，减法指令SUB B表示把寄存器B的内容与累加器A的内容相加，结果保存在累加器A中。其指令结构为：



指令中最低三位指定通用寄存器r中存放着一个操作数，隐含着另一个操作数在累加器A中。指令中没有给出寄存器A的号码。最低三位为000B时，代表寄存器B的号码。

第三节 Z80微处理器指令系统

我们要用机器指令或汇编语言来编写程序，必须对机器的指令系统（Instruction Set）十分了解。虽然各种机器的指令系统不尽相同，但它们要执行的基本指令，它们的寻址方式总还是有共性的。Z80微处理器的指令系统共有158种指令，包括了Intel 8080的全部78种指令。这一指令系统功能较强，较齐全，在第二代微型机中有一定的代表性。所以，我们以Z80的指令系统为例来进行分析是有其一般意义的。

Z80微处理器的指令按功能分有九大类：

(1) 数据传送类指令
(2) 转移类指令
(3) 算术和逻辑运算类指令
(4) 调用和返回类指令
(5) 通用算术运算和CPU控制类指令
(6) 循环与移位类指令
(7) 位操作类指令
(8) 交换类指令、数据块传送与查找类指令
(9) 输入/输出类指令

下面我们将按上述顺序逐类介绍和分析，与中断有关的指令将留在第六章介绍。

一、数据传送类指令

数据传送类指令的功能是在通用寄存器之间、寄存器和存贮器之间传送数据。这类指令可以对8位数据进行传送，也可以对16位数据进行传送。指令中除操作码外，通常包括有数据传送的源地址和目和地址。而源地址中的内容不因数据传送而有所改变或消失。即源地址中数据有“取之不尽”的特点。

这类指令的助记符表示是LD。在LD后面紧跟着的是数据的目的地址和源地址（即LD目的地址，源地址）。

所有数据传送类指令的执行，都不会影响状态标志位的状态。

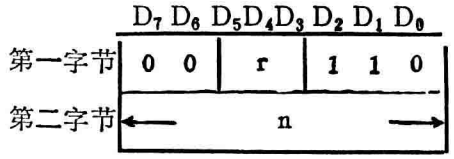
1. 8位数的传送类指令

这类指令共有21类，见附录1，表1—1。

(1) 立即数送寄存器的传送

助记符格式：LD r, n

机器代码格式：



这条两字节指令的含意是：将第二字节中的立即数n送入CPU内部的寄存器r中。机器代码中的r表示A、B、C、D、E、H、L 7个寄存器中的一个，这类指令的功能如图4—2所示。

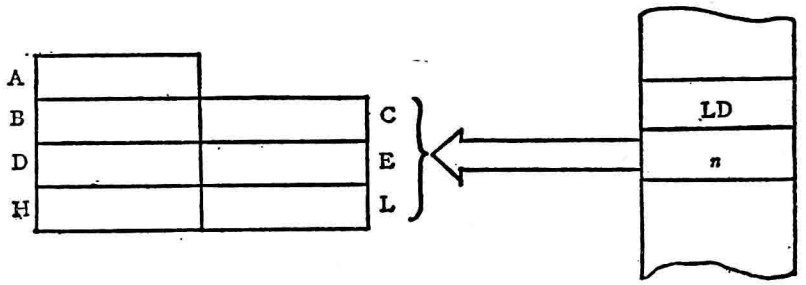


图4—2 指令LD r, n功能示意图

Z80 CPU 寄存器的代码规定如下：

0 0 0	B
0 0 1	C
0 1 0	D
0 1 1	E
1 0 0	H
1 0 1	L
1 1 1	A

代码 1 1 0 另有它用。

若将上述代码代入指令，就可得到对应的指令代码。

例如，指令 LD A，15H，其含意是将立即数 15H 送入累加器 A。

对应的机器代码格式为：

0	0	1	1	1	1	1	0
0	0	0	1	0	1	0	1

这条指令的 16 进制形式为：3E 15。

这样的机器代码可从附录 2 的表 2—1 直接查到。

(2) 寄存器之间的数据传送

助记符格式：LD r，s

机器代码格式：

D_7	D_6	D_5 D_4 D_3	D_2 D_1 D_0
0	1	r	s

这是一条单字节指令，其中 r 和 s 均为上面指出的三位二进制代码。每个代码表示一个 CPU 寄存器。其功能是将寄存器 s 的内容送入寄存器 r，如图 4—3 所示。

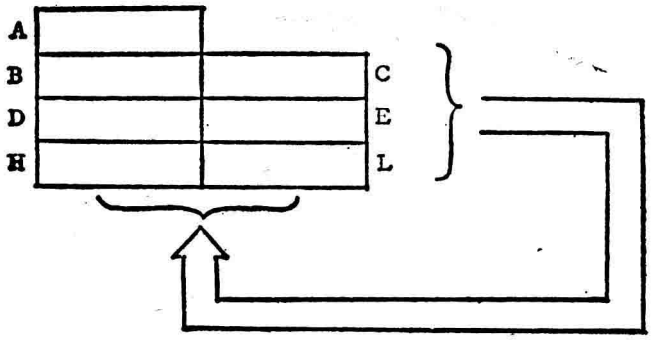


图4—3 指令 LD r，s 数据传送示意图

从附录 2 表 2—1 可看出由 r 和 s 的各种不同组合能得到 49 条寄存器寻址的传送指令。

例4—1，先用指令 LD r，n 将立即数序列 00H～04H 分别送入寄存器 A、B、C、D、

E，然后再用指令 LD、r，s 把此数的序列倒过来，成为 04H～00H。

根据题意，其源程序为：

地址（十六进制）	机器代码（十六进制）	助记符指令				注释
2000	3E00	LD	A，	00H	；*	A←00H**
2002	0601	LD	B，	01H	；	B←01H
2004	0E02	LD	C，	02H	；	C←02H
2006	1603	LD	D，	03H	；	D←03H
2008	1E04	LD	E，	04H	；	E←04H
200A	67	LD	H，	A	；	H←A
200B	7B	LD	A，	E	；	A←E
200C	5C	LD	E，	H	；	E←H
200D	60	LD	H，	B	；	H←B
200E	42	LD	B，	D	；	B←D
200F	54	LD	D，	H	；	D←H
2010	76	HALT			；	暂停

(3) 寄存器间接寻址的数据传送

① 寄存器对 HL 间接寻址的数据传送

助记符格式：

```
LD  (HL), r    ;    (HL)←r
LD  r, (HL)    ;    r←(HL)
LD  (HL), n    ;    (HL)←n
```

前两条是一组双向传送指令，是单字节指令。第三条是两字节指令。在这组指令中，是以HL 的内容作为存贮单元地址，可把任一寄存器的内容，送到该存贮单元；或把这一存贮单元的内容送到任一寄存器；或把立即数送到该单元。上述各组指令分号后的解释指出了它们的功能。

现仅以指令 LD (HL)，r 为例，画出其功能示意图，如图 4—4 所示。

---

* 通常汇编语言的格式中规定："；" 号后面是注释部分，用以说明指令的功能，便于阅读程序。汇编时，对 "；" 号后面的内容不作处理。

** 我们用 A←00H 表示立即数送累加器A，一般地有 r←n。此外还用 r←s 表示把源寄存器 s 的内容送目的寄存器 r；用 r←(nn) 表示将第 nn 号存贮单元的内容送寄存器 r；用 r←(HL) 表示将寄存器对 HL 的内容为地址的存贮单元的内容送寄存器 r。

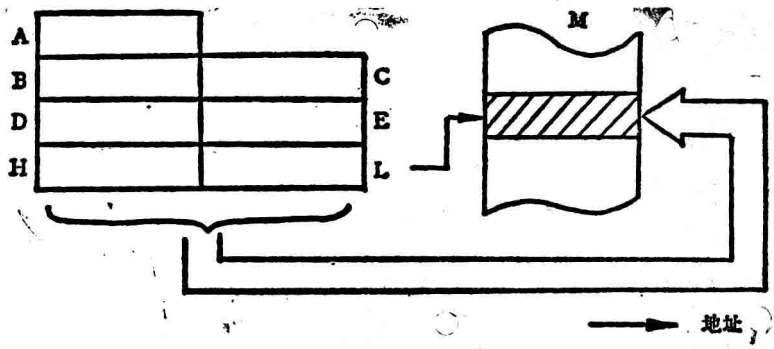
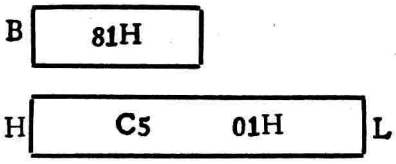
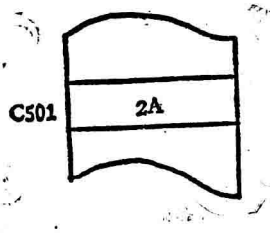


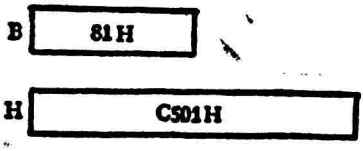
图4—4 指令LD (HL), r功能示意图

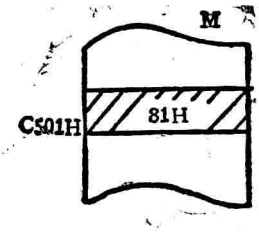
例如，设指令LD(HL)，B执行前，有关寄存器及有关存贮单元的内容为：





该指令执行后，它们的内容分别为：





例4—2 将2035H单元的内容和2045H单元的内容互相交换。

程序为：

```
1000  2620  LD  H,     20H
1002  2E35  LD  L,     35H
1004  46    LD  B,     (HL)     ;  B←(2035H)
1005  2E45  LD  L,     45H
1007  4E    LD  C,     (HL)     ;  C←(2045H)
1008  70    LD  (HL),  B        ;  (2035H)←B
1009  2E35  LD  L,     35H
100B  71    LD  (HL),  C        ;  (2045H)←C
100C  76    HALT
```

② 寄存器对BC和DE间接寻址的数据传送

指令格式：

```
LD  A,  (BC)  ;  A←(BC)
LD  A,  (DE)  ;  A←(DE)
```

LD (BC), A ; (BC)←A
LD (DE), A ; (DE)←A

这些都是单字节指令，功能与以寄存器对 HL 间接地址的数据传送一样。但只是这组指令仅对累加器 A 实现数据传送。

指令 LD A，(BC) 的功能如图 4—5 所示。

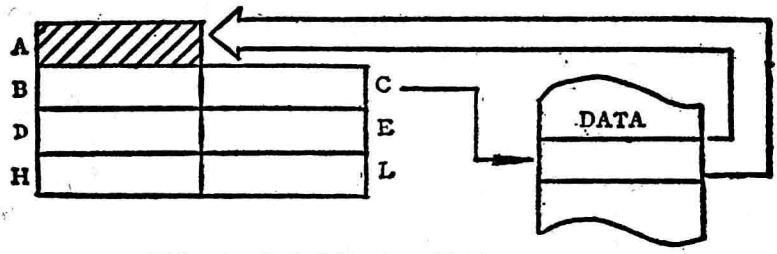


图4—5 指令 LD A，(BC) 功能示意图

(4) 扩展寻址（直接寻址）的数据传送

助记符格式：

LD A, (nn) ; A←(nn)
LD (nn), A , (nn)←A

这两组指令实现累加器 A 第 nn 号存贮单元之间的数据传送。指令 LD A，(nn) 的功能如图 4—6 所示。

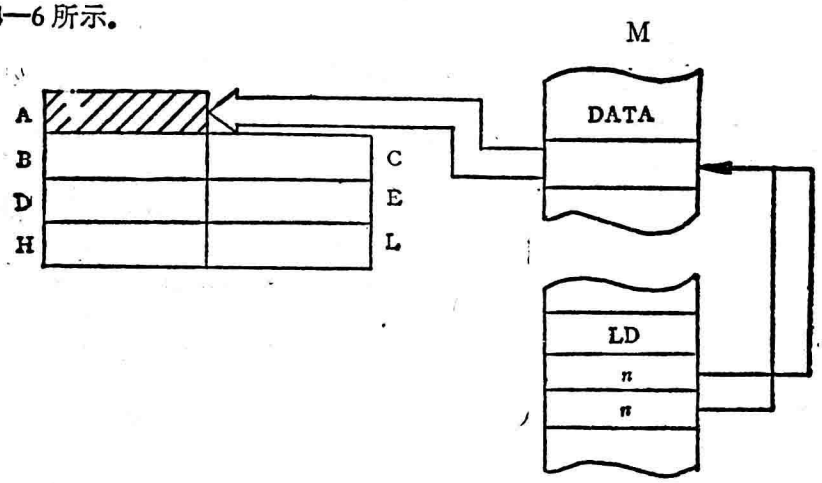


图4—6 指令 LD A，(nn) 功能示意图

由于存贮单元的地址信息是 16 位二进制数，故这类指令是三字节指令，第一字节是操作码，第二、三字节则是操作数地址，即

操作码	第一字节
地址低8位	第二字节
地址高8位	第三字节

例如：指令 LD A，(5000H) 是将内存单元 5000H 号的内容送到累加器 A 中，机器代码为：

3AH	第一字节
00H	第二字节
50H	第三字节

或写成：3A0050H

(5) 变址寻址的数据传送

助记符格式：

```
LD  r, (IX+d)        ;  r←(IX+d)
LD  r, (IY+d)        ;  r←(IY+d)
LD  (IX+d), r        ;  (IX+d)←r
LD  (IY+d), r        ;  (IY+d)←r
LD  (IX+d), n        ;  (IX+d)←n
LD  (IY+d), n        ;  (IY+d)←n
```

前四条是两组三字节的寄存器和存贮器之间的双向传送指令，以 IX(或 IY)的内容加上偏移量 d 后的结果作为存贮单元地址，该存贮单元和指令中规定的寄存器 r 进行数据传送。

后两条是四字节的立即数送入由变址寻址的存贮单元指令，在 Z80 CPU 中有两个功能相同的 16 位变址寄存器 IX 和 IY，它们由操作码指定。d 是一个由指令给定的带符号的偏移量，其范围为+127——128。

指令 LD r, (IX+d) 的功能如图 4—7 所示。

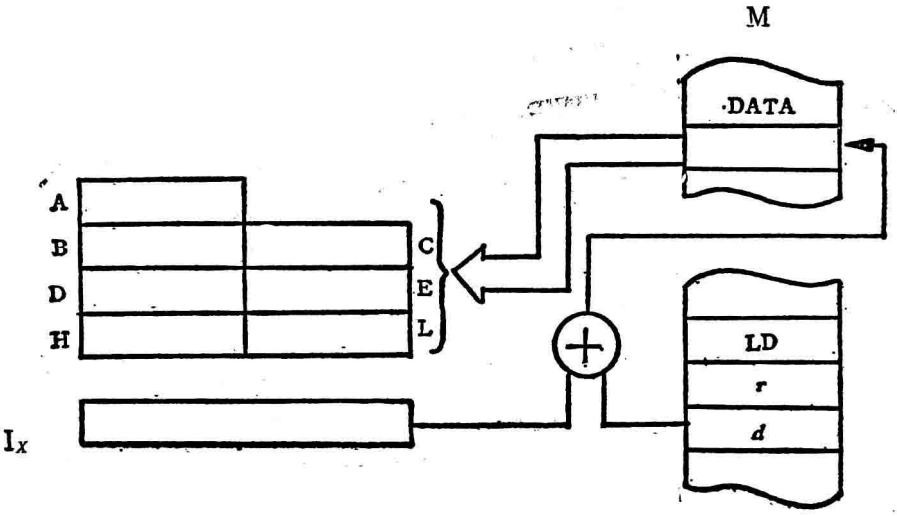


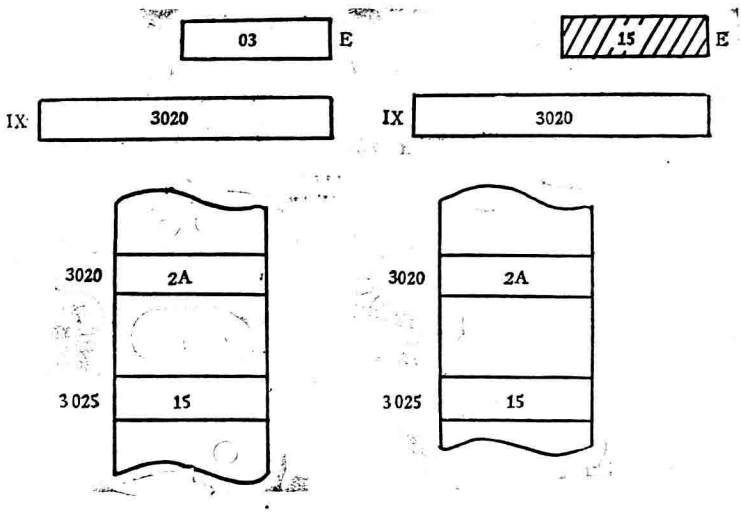
图4—7 指令 LD r, (IX+d) 功能示意图

例如，指令 LD E, (IX+5) 的机器代码为

DDH
5EH
05H

该指令在执行前后有关寄存器和存贮单元的内容别分为：

执行前　　执行后



又如，指令 LD E，(IX−5)。它的机器代码为：

DDH
5EH
FBH

其中 FBH 是−5H 的补码表示。该指令的执行情况和 LD E，(IX+5) 是相同的。

需要说明的是，变址寄存器 IX 或 IY 的内容和偏移量 d 相加后，不影响 IX、IY 原来的内容。

例4—4　把寄存器 A、B、C 和 D 的内容分别存入 2000H，2004H，2008H，200CH 号存贮单元中。

源程序如下：

```
1000  DD210420   LD   IX, 2004H   ;  置地址初值为 2004H
1004  DD77FC     LD   (IX−4), A   ;  (2000H)←A
1007  DD7000     LD   (IX+0), B   ;  (2004H)←B
100A  DD7104     LD   (IX+4), C   ;  (2008H)←C
100D  DD7208     LD   (IX+8), D   ;  (200CH)←D
1010  76         HALT
```

程序中，指令 LD IX，2004H 是一条立即扩展寻址的传送指令，其功能是把地址初值 2004H 送入变址寄存器 IX。

2. 16 位数的传送类指令

上面讨论的数据传送，每次只能传送一个字节即 8 位数，然而 Z80 CPU 有许多条指令每次能传送两个字节即 16 位数。这类指令共有 20 条（见附录1表 1—2）。这里，我们仅讨论前边 11 条指令，对后边 9 条指令将在本节第四部分讨论。这些 16 位数的传送指令的机器代码可以在附录 2 表 2—2 中查到。

Z 80 CPU 中的三个寄存器对 BC、DE、HL 和三个 16 位寄存器 IX、IY、SP 能够和任意 2 个相邻存贮单元进行 16 位数据传送，也能直接接收 16 位的立即数。也即这 11 条指令仅按扩展立即寻址和直接寻址两种方式进行数据传送。

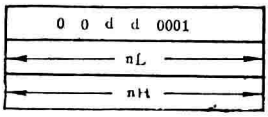
(1) 扩展立即寻址方式的 16 位数据传送

助记符格式：

```
LD    dd, nn  ;   dd←nn
LD    IX, nn  ;   IX←nn
LD    IY, nn  ;   IY←nn
```

这里把指令中包含的两个字节的立即数，送入上述寄存器对或专用寄存器。

其中，LD dd，nn 的功能如图 4—8 所示。而它的二进制机器代码的格式如下所示：



操作码
立即数低 8 位
立即数高 8 位

第一个字节中 dd 为 Z80 CPU 中三个寄存器对的代码，即

dd	寄存器对
0 0	BC
0 1	DE
1 0	HL
1 1	SP

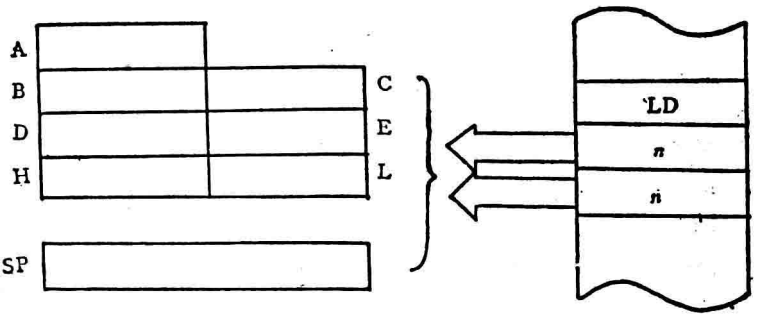


图4—8　指令 LD dd，nn 功能示意图

而 LD IX，nn 及 LD IY，nn 二条指令则为四字节指令，其格式为