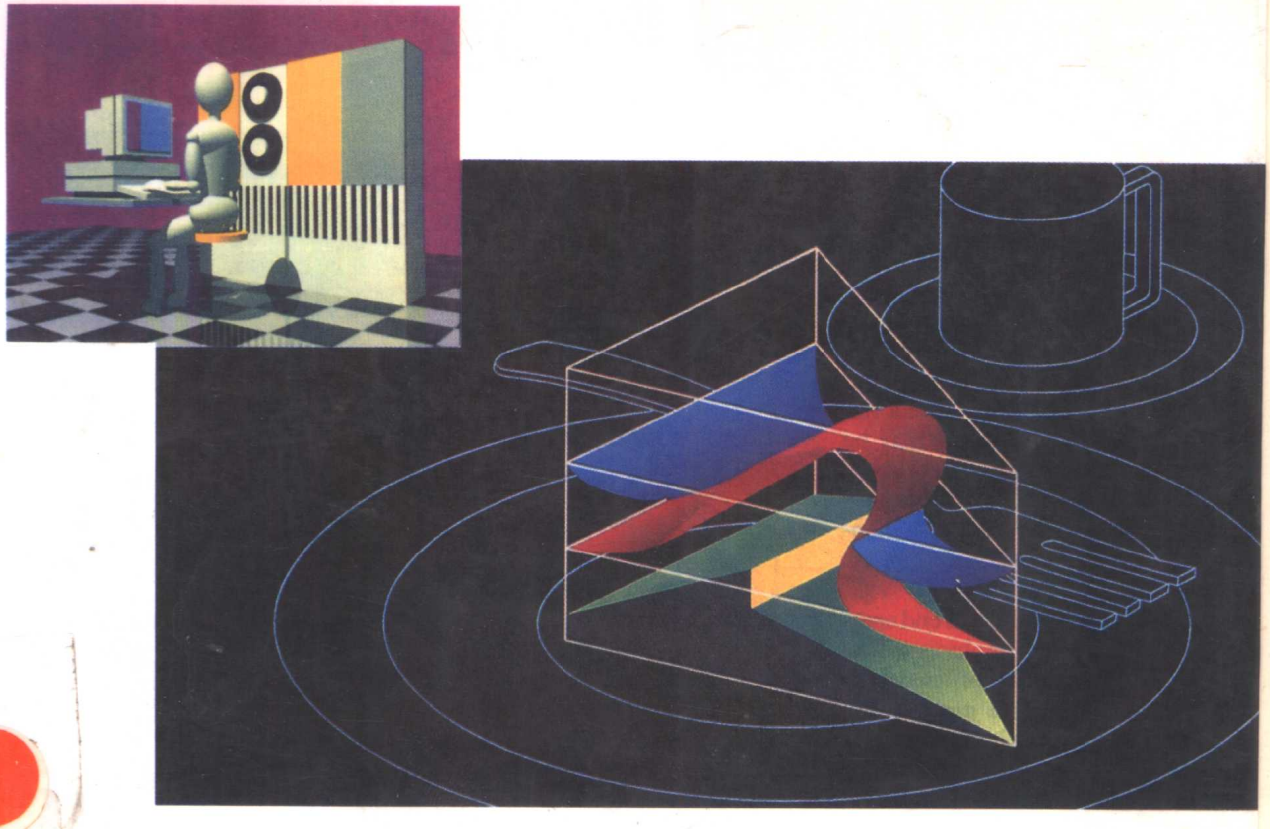
intel

Intel单片机高级开发语言

PL/M程序设计技巧与实例荟萃

张良杰　林　盛　范存奎　编著



科学出版社

Intel单片机高级开发语言

PL/M程序设计技巧与实例荟萃

张良杰　林　盛　范存奎　编著

科学出版社

1996

（京）新登字 092 号

内 容 简 介

本书系统阐述具有广阔应用前景的 Intel 单片机高级开发语言 PL/M 程序设计技巧与实例. 全书共分三部分，第一部分阐述 PL/M 语言的高效、简洁、调试维护方便的特点及基本程序设计技巧. 第二部分阐述 PL/M 语言在键盘/显示器接口、多机串行口通信、信号滤波、时频域分析、基本的矩阵与向量运算、图像处理、模糊逻辑推理与控制、神经网络智能信息处理等领域的具体程序设计技巧与设计实例. 第三部分介绍运用 PL/M 语言进行应用系统设计的技巧. 本书是在作者积累多年开发经验的基础上撰写的，书中给出了大量面向应用对象的程序和实例，并全部通过了上机调试，可直接用于实际项目的开发应用之中. 本书具有很强的可读性.

本书可作为高等学校计算机、电子工程、自动控制、信息处理、通信等专业的大学生和研究生的参考书，也可供从事这方面工作的广大科技人员及应用开发人员学习参考.

Intel 单片机高级开发语言

PL/M 程序设计技巧与实例荟萃

张良杰 林 盛 范存奎 编著

责任编辑 鞠丽娜

科学出版社出版

北京东黄城根北街 16 号

邮政编码：100717

中国科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1996 年 1 月第 一 版 开本：787×1092 1/16

1996 年 1 月第一次印刷 印张：18 1/2

印数：1—4 500 字数：495 000

ISBN 7-03-004736-2/TP・446

定价： 29.00 元

序

作为一个单片机开发用户，一般要经历单片机选型、开发语言的选择、应用程序的编制、实际系统的调试直至产品定型等阶段. 而这些阶段又相互关联，融为一体. 如何以最快的速度将全新的思想变成实用的产品已成为应用开发者孜孜以求的目标.

此书正是拟在此方面所作的一次尝试. 它是集作者多年 PL/M 语言实际应用开发与高新技术研究经验于一体，在实际应用系统设计的基础上提炼出的一本专著. 概括起来此书具有如下特点：

1. 取材独特、新颖和实用. 它不仅介绍了键盘/显示器接口、多机串行口通信等常规硬件接口软件设计、信号滤波、时频域分析、基本的图像处理等实用数字信号处理程序设计，而且还涉足了模糊逻辑推理与控制、神经网络智能信息处理等新兴技术领域，并在此基础上，详细介绍了这些新兴技术在家电设备智能控制系统设计中的应用.

2. 详细介绍了实际应用系统的软件设计技巧，包括基于 MCS96 系列单片机的模糊智能变频空调控制系统和基于 MCS51 系列单片机的智能空调控制系统设计等. 书中充分展现了 PL/M-96 和 PL/M-51 高级开发语言的应用编程技巧.

3. 涉及面广. 它不仅对于学习 PL/M 语言及其程序设计技巧有很高的实用价值，而且对于如何把新兴技术(如模糊控制、人工神经网络技术等)应用到实际系统开发中亦有很高的参考价值.

鉴于此，我愿意将它推荐给广大的单片机用户，希望它能成为读者在实际应用中的一个得力工具.

Intel 技术发展有限公司半导体事业部
技术专案经理　姜　洪
1995 年 6 月

前　言

传统的单片机开发工作无一例外地都采用汇编语言作为开发语言.对于小型的单片机系统,汇编语言的确起到了高效运行、调试方便的作用,然而,当系统规模增大时,用汇编语言编制上千行乃至上万行的程序无疑是一件枯燥、乏味而难于调试的工作.针对单片机开发中出现的这些问题,Intel公司以极其敏锐的战略眼光推出了PL/M高级语言.它是开发各种Intel微处理器和微控制器的理想高级开发语言.特别是对于大型应用系统,开发的程序量将比汇编语言少几倍至十几倍,开发效率显著提高,开发周期明显缩短,并且具有极佳的可维护性;另一方面,PL/M语言也是一种能直接操纵微处理器的类"低级"语言.它的地址分配是静态的,不具备微机上C语言那样的可浮动性;外部接口的地址必须准确给出;对片内集成的串行口和定时器的操作等均带有汇编语言的性质.而这些都是软硬件兼备的单片机系统对开发语言PL/M的"低级"要求之处,也正是这种特色,给开发人员提供了很强的灵活性.这种灵活度丝毫不亚于汇编语言,因而PL/M是一种既具有汇编语言的灵活,又具有高级语言开发效率的一种成功的、很值得推广的微处理器高级开发语言.

PL/M语言的特点主要体现在以下几个方面:

(1) 具有程序块结构和控制构造,有助于结构化编程;

(2) 具有结构数组以及有基指针动态变量等高级数据结构;

(3) 具有类型化特色的高级语言,即程序在编译时做数据类型检查以帮助查找程序中的逻辑错误;

(4) 具有系统编程表达良好的逻辑式数据结构服务程序和控制语句;

(5) 具有确保程序正确无误的清晰控制结构;

(6) 具有Intel微处理器与微控制器之间的良好兼容性与可移植性;

(7) 具有对微处理器全系统资源进行最佳利用的系统易操作特性.

正是上述优良的特性使得PL/M语言目前已成为Intel微处理器与微控制器的首选开发语言.

随着现代智能信息处理与智能控制技术的迅猛发展,要使通用的微处理器与微控制器展现出强大的生命力,就必须将先进的智能信息处理与智能控制技术融入通用微处理器与微控制器中.本书是在我们积累多年实际开发经验的基础上撰写的,书中除介绍通用的微处理器I/O接口实用程序之外,还介绍了当代最先进的智能信息处理与智能控制技术,如模糊逻辑控制、神经网络信息处理等.

本书共给出数十个实用程序,而且每个程序都给出了算法概要、完整的PL/M语言程序及应用示例.程序用PL/M-96或PL/M-51高级语言编写,并均通过了上机调试.这些程序几乎不用作修改便可直接用于实际项目开发中.本书共分十章.第一章介绍PL/M语言的特点及实用编程技巧,它是以后各章的基础;第二章介绍人机接口的PL/M语言实用程序设计;第三章介绍串口通信技术的PL/M实用程序设计;第四章介绍图像处理技术的

PL/M 程序设计；第五章介绍信号滤波技术的 PL/M 程序设计；第六章介绍时频域分析的 PL/M 程序设计；第七章介绍电力电子控制的 PL/M 实用程序设计；第八章介绍模糊逻辑推理与控制技术的 PL/M 实用程序设计；第九章介绍神经网络智能信息处理的 PL/M 程序设计；第十章介绍应用系统设计实例. 以上几章内容在实际应用中既密切相关，又相对独立，读者可根据工作需求灵活选择.

在本书出版过程中，中国科学院院士、清华大学信息科学技术学院院长李衍达教授以及王普副教授、李崇荣副教授给予我们很多指导与帮助，谨在此向他们表示衷心的感谢！在我们从事微处理器系统的开发工作期间，清华大学自动化系毕业生刘军、唐欣、张军、沈海颖等参与了部分工作，自动化系硕士生李庆华、孙冠军及本科生孙亚彬、曹恒、毛志宏、苏青等亦为本书提供了许多素材. 在此致以诚挚的谢意！

书中的不妥之处，敬请广大读者批评指正！

作　者

1994 年 7 月 16 日写于清华园

目　　录

第一章　PL/M 语言的特点及实用编程技巧

1.1　PL/M 语言及特点

1.1.1　PL/M 语言的特点

PL/M 程序设计语言是美国 Intel 公司推出的一种用于微处理器开发的高级语言. 它是开发各种 Intel 微处理器和微控制器的理想高级开发语言. 特别是对于大型应用系统，开发的程序量将比汇编语言少几倍甚至十几倍，开发效率显著提高，开发周期亦明显缩短，并且具有极佳的可维护性. PL/M 语言主要包含 PL/M-51、PL/M-86、PL/M-96 三种类型，分别针对不同类型的微处理器. 一般来讲，PL/M-51 适用于 MCS8X51 系列单片机开发，PL/M-86 适用于 8X86 系列微处理器开发，PL/M-96 适用于 MCS-96 系列单片机的开发. 因为每种单片机需自己特定形式的机器代码，这些代码可通过 PL/M 高级开发语言经编译、连接后产生.

另外，PL/M 语言亦是一种能够直接操纵微处理器的类“低级”语言. 它的地址分配是静态的，不具备微机上 C 语言那样的可浮动性；外部接口地址必须由硬件准确给出；对片内集成的串行口和定时器的操作等均带有汇编语言的性质. 而这些均是软硬件兼备的单片机系统对开发语言 PL/M 的“低级”要求之所在. 亦正是这种特色，给开发人员提供了很强的灵活性，这种灵活性丝毫不亚于汇编语言，因而我们说 PL/M 是一种既具有汇编语言的灵活，又具有高级语言开发效率的一种成功的、很值得推广的微处理器高级开发语言.

概括来讲，PL/M 语言的特点主要体现在以下几个方面：

1) 具有程序块结构和控制构造，有助于结构化编程；

2) 具有结构数组以及有基指针动态变量等高级数据结构；

3) 具有类型化特色的高级语言，即程序在编译时做数据类型检查以帮助查找程序中的逻辑错误；

4) 具有系统编程表达良好的逻辑式数据结构服务程序和控制语句；

5) 具有确保程序正确无误的清晰控制结构；

6) 具有 Intel 微处理器与微控制器之间的良好兼容性与可移植性；

7) 具有对微处理器全系统资源进行最佳利用的系统易操作特性；

8) 由 PL/M 语言生成的目标文件可与汇编语言及其它高级开发语言(如 C)进行统一的连接.

1.1.2　PL/M 语言基础

1. PL/M 语言的字符集

PL/M 语言程序是由以下合法字符组合而成，除字符串常数外，英文字母大小写不加区分，这些合法字符为

英文字母：

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z

数字： 0 1 2 3 4 5 6 7 8 9

算术运算符： + − * / MOD

关系运算符： < > = <= >= <>

逻辑运算符： NOT AND OR XOR

分界符： . ( ) , : ; '

特殊字符： $ _ := /* */

空字符： Space(空格) Tab(制表)

Carriage-Return(回车) Line-feed(换行)

在 PL/M 语句中,这些字符的名称和作用如表 1.1 所示.

表 1.1 PL/M 语言特殊字符作用表

符号	名称	作 用
@	at 符	引用地址(仅在 PL/M-86 中使用)
$	美元符号	数和标识符中加入 $,仅是为了改善可读性,而不改变任何含义
:=	赋值号	内嵌赋值时的赋值号
_	下划线	可作标志符中的有效符号
/*	注释前分界符	每个注释以此分界符开始
*/	注释后分界符	每个注释以此分界符结束
=	等号	(1)赋值运算符 (2)关系测试符
.	点	(1)十进制小数点 (2)结构成员限定 (3)地址运算符
/	斜杠	除法运算符
(	左括号	表、下标及表达式的左分界符
)	右括号	表、下标及表达式的右分界符
+	加	加运算符
−	减	减运算符
'	小撇	串分界符
*	星号	乘运算符
<	小于	关系测试运算符

续表

符号	名称	作　用
>	大于	关系测试运算符
>=	大于等于	关系测试运算符
<=	小于等于	关系测试运算符
<>	不等于	关系测试运算符
:	冒号	标号分界符
;	分号	标号分界符
,	逗号	表元素分界符

除字符串常数外，任何空符号作用相同，任何未分开的一串空符号与单个空符号相同. 除字符串常数外，大小写字母意义相同. 值得提醒的是，如果在 PL/M 程序中出现了上述未列出的任何字符，编译程序将其识为错误.

2. PL/M 语言的保留字及标识符

标识符用来命名变量、过程、宏说明和标号等. 一个标识符最长可有 31 个字符. 每个标识符第一个字母必须是英文字母，其余字符可以是字母、数字或下划线，但在标识符中不允许用美元符号 $ 作为第一个字符.

保留字不能用作标识符，它们已有确定的意义，已构成 PL/M 语言不可分割的一部分，其中的 EOF 标识符作为 PL/M-96 的一条特殊语句，表示一个模块的结束，但对编译结果不产生影响.

预说明的标志符是 PL/M 语言提供的内部过程和内部变量. 预说明的标识符与保留字不一样之处在于可以用说明语句重新说明这些标识符，且在这个说明的作用域内不能再使用相应的内部过程和内部变量. 现仅以 PL/M-96 语言与 PL/M-51 语言为例，将保留字及标志符进行整理.

(1) PL/M-96 的保留字如下：

ADDRESS	EOF	OR
AND	EXTERNAL	PLUS
AT	FAST	PROCEDURE
BASED	GO	PUBLIC
BY	GOTO	REAL
BYTE	IF	REENTRANT
CALL	INDIRECTLY_CALLABLE	RETURN
CASE	INTEGER	SHORTINT
DATA	INTERRUPT	SLOW
DECLARE	INTERRUPT_CALLABLE	STRUCTURE
DISABLE	LABEL	THEN
DO	LITERALLY	TO
DWORD	LONGINT	WHILE
ELSE	MINUS	WORD

ENABLE	MOD	XOR
END	NOT	

PL/M-96 语言的内部过程和预说明变量的标志符如下(大小写无区别)：

ABS	IABS	SETB
BITASR	INIT $ REAL $ MATH $ UNIT	SET $ REAL $ MODE
BITCLK	LAST	SETW
BITCPL	LENGTH	SHL
BITSET	LOW	SHORT
BITTST	MEMORY	SHR
CARRY	MOVB	SIGN
CMPB	MOVE	SIGNED
CMPW	MOVW	SIZE
DOUBLE	OVERFLOW	SKIPB
EXTEND	OVERFLOW $ TRAP	SKIPW
FINDB	RESTORE $ REAL $ STATUS	STACK $ PTR
FINDW	ROL	STICKY $ BIT
FIX	ROR	TIME
FLOAT	SAL	UNSIGN
GET $ REAL $ ERROR	SAR	ZERO
HIGH	SAVE $ REAL $ STATUS	

(2) PL/M-51 语言的保留字如下：

ADDRESS	ENABLE	NOT
AND	END	OR
AT	EXTERNAL	PLUS
AUXILIARY	GO	PROCEDURE
BASED	GOTO	PUBLIC
BIT	IDATA	REGISTER
BY	IF	RETURN
BYTE	INDIRECTLY_CALLABLE	STRUCTURE
CALL	INTERRUPT	THEN
CASE	LABEL	TO
CONSTANT	LITERALLY	USING
DECLARE	MAIN	WHILE
DISABLE	MINUS	WORD
DO	MOD	XOR
ELSE		

PL/M-51 语言的内部过程和预说明变量的标志符如下(大小写无区别)：

BOOLEAN	INPUT	PROPAGATE	SHL
DEC	LAST	ROL	SHR
DOUBLE	LENGTH	ROR	SIZE
EXPAND	LOW	SCL	TESTCLEAR
HIGH	OUTPUT	SCR	TIME

3. PL/M 语言的数据类型

PL/M 语言中的数据类型主要有常数和变量两种. 其中常数包含数值常数、字符串常数. 数值常数又包含纯数常数、浮点常数等.

所谓常数，是指在程序执行期间不改变的值，纯数常数是指不包含小数点的数值常数，它可表示成二进制、八进制、十进制或十六进制，以后分别缀 B,O 或 Q,D,H 表示，而

不带后缀的数被默认为十进制数.

纯数常数根据其类型可划分成节型(BYTE)、字型(WORD)、双字型(DWORD)、短整型(SHORTINT)、整型(INTEGER)、长整型(LONGINT)、地址型(ADDRESS)等 7 种. 浮点常数是由十进制小数点标志的十进制数.

纯数常数和浮点常数的取值范围如表 1.2 所示.

表 1.2 PL/M 数据类型取值表

数据类型	取值范围	所占存储单元数(字节)
BYTE 型	0～255	1
WORD 型	0～65535	2
DWORD	0～4294967295	4
SHORTINT	−128～+127	1
INTEGER	−32768～+32767	2
LONGINT	−2147483648～+2147483647	4
ADDRESS		2
浮点数(如 REAL 型)	1.17×10^{-38}～3.4×10^{38}(绝对值)	4

变量是其值在程序执行或运算过程中可以改变的量,它不写成具体数的形式,而是用一个名字即标志符来表示. 变量可以是标志变量(标量)、数组或结构. 数组或结构是一组标量,但可以使用同一个标志符来引用. 在同一作用域内,某一标志符被说明为标量后,将不能再被说明成数组. PL/M 语言中所有变量和数据均应属于某一类型. 即在 PL/M-96 中有表 1.2 所示的 8 种类型,此时 ADDRESS 型与 WORD 型不加区别.

一般来讲,同种数据类型可直接进行运算,但亦不排除隐含类型转换后的异种数据类型进行运算. 如无法完成隐含类型转换,则应使用内部过程将多种数据类型进行转换.

4. PL/M 语言的数组和结构

数组是同类型的一组标量,用一个标志符来引用一组标量,借助于下标来区分各个标量. 数组中单个标量称为数组元素. 一个数组用一个维数区分符来说明. 维数区分符是一个括在括号中的纯数常数,常数的值描述了构成数组的数组元素的个数,且数组元素是依次相邻存储的,如:

```
declare A(100) byte;
```

上式说明数组 A 有 100 个元素,且每个元素均为字节型数,值得说明的是,A(0)～A(99)才是数组 A 的 100 个元素.

结构是用一个结构名(标识符)来引用一组结构成员,这些结构成员可以有不同的类型. 如:

```
declare human structure(hair byte,height real,weight real);
```

从上面的说明语句可以看出，structure 为其关键词，结构成员为 human. hair，human. height，human. weight，显然此种类型的结构等价于一个一维数组.

结构数组是指结构名为数组的结构，其成员名是标量变量名. 结构数组的说明如下：

declare human(20) structure(hair byte,height real,weight real);

此语句说明了与标识符 human 相联系的 20 个结构，下标从 0 到 19，每个结构有 2 个实型变量成员，1 个字节型变量成员. 整个结构数组共分配了 40 个实型变量的存储单元，20 个字节型变量的存储单元，总共为 180 个字节存储单元.

另一类结构是结构内数组和结构数组内数组，仅分别给出说明示例如下：

declare A structure(B(5)real,C(6) byte,D word);

declare X(10) structure(y(10) real,Z(6)word,XZ byte);

值得说明的是，还用一种数组的隐含长度说明，其隐含长度说明符在说明语句中的格式示例为

declare t(*) byte data(0,1,2);

总的看来，数组与结构在实际应用系统开发程序设计中扮演着十分灵活的角色，这将在后面的介绍中逐步呈现出来.

表达式在计算时遵照运算符的优先级，其优先级规则如下：

1) 若一个运算对象仅有一个直接相邻的运算符，它束缚于这个运算符.

2) 若一个运算对象在它的两边有直接相邻的运算符，它束缚于较高优先级的运算符.

3) 若运算符优先级相同，则遵守从左到右的规则. PL/M 语言的运算符优先级如表 1.3所示.

表 1.3 运算符优先级表

类别	运算符	优先级
括号	( )	1(最高级)
减(加)	−(+)	2
算术运算符	* / MOD + −	3 4
关系运算符	< <= <> = >= >	5
逻辑运算符	NOT AND OR XOR	6 7 8(最低级)

PL/M 语言的表达式运算规则小结如表 1. 4 所示.

表 1.4　PL/M 语言表达式运算规则小结表

变量类型	运算类型	运算数类型	算术运算	结果	备注
DWORD WORD BYTE	无符号运算	BYTE 与 BYTE	+或-或/或 MOD *	BYTE WORD	范围：0～255 范围：0～65535
		BYTE 与 WORD 变成 WORD 与 WORD	任意	WORD	首先将字节型运算数高 8 位补 0 扩展成字型值
		BYTE 与 DWORD 变成 DWORD 与 DWORD	任意	DWORD	首先将字节型运算数高 24 位补 0，扩展成双字型值
		WORD 与 DWORD 变成 DWORD 与 DWORD	任意	DWORD	首先将字形运算数高 16 位补 0，扩展成双字型值
		BYTE 与小于 256 的纯数常数	+或-或/或 MOD *	BYTE WORD	常数作为字节型 常数作为字型
		BYTE 或 WORD 与小于 65536 的纯数常数	任意	WORD	常数作为字型
		BYTE 或 WORD 与大于 65536 的纯数常数	任意	DWORD	常数作为双字型
		DWORD 与小于 4294967295 的纯数常数	任意	DWORD	常数作为双字型
SHORTINT INTEGER LONGINT	有符号运算	SHORTINT 与 SHORTINT	+或-或/或 MOD *	SHORTINT INTEGER	范围：-128～+127
		INTEGER 与 INTEGER	任意	INTEGER	范围：-32768～+32767
		LONGINT 与 LONGINT	任意	LONGINT	范围：-2147483648～+2147483647
		SHORTINT 与 INTEGER 变成 INTEGER 与 INTEGER	任意	INTEGER	SHORTINT 型值的符号位扩展到高 8 位成为 INTEGER 型
		SHORTINT 与 LONGINT 变成 LONGINT 与 LONGINT	任意	LONGINT	SHORTINT 型值的符号位扩展到高 24 位成为 LONGINT 型