

## UART 技术和 MAX3100

Wettroth/Suppan Maxim Integrated Products 栾成强 译

摘要: 本文回顾了 UART 的发展过程及应用, 分析了 UART 的市场需求, 详细介绍了 MAXIM 公司的新型 UART-MAX3100 及其应用。附录给出了软件清单。

关键字: UART, 接收, 发送, 接口, 速率

### 1. 言引

通用异步接收/发送器 (UART) 产生于 70 年代, Intel 8250 和 Intersil6402 是现代 CMOS 标准 UART (如国半的 16550 和 ZILOG8630) 的早期产品。一般来说, UART 制造讲究, 几年后才重新研制。直到今天其通用性、管脚、寄存器仍很少改变。

96 年的市场上有 40 种 UART, 而 MAXIM 却推出了另一种 UART—MAX3100。为什么象 MAXIM 这样精明的公司会花时间和资金研制新的 UART? 为什么 40 种还不能满足要求? 让我们首先回顾一下 UART 的历史、20 年来的变化, 对现代 UART 的特性及其新的应用领域作一深入研究, 即可得出问题答案。

### 2. UART 应用历史

电子学在过去二十年里发生了深刻的变化, UART 是第一个大规模集成电路, 在单片微处理器出现前几年就已经产生了 UART, 目前的 UART 与二十年前相比, 结构基本相似。PC 机的发展对 UART 技术起到了重大推动作用, 下面简要介绍 UART 的发展历史和现代的市场观念。

PC 机中的 UART:

1981 年, 在 IBM PC 机主板上用 8250 UART 与 MODEM 或串行打印机进行通信, 由于 8250 的应用包含对 PC 机中 BIOS 的支持, 因而确定了它的结构和特性。几年后, 8250 的结构有所扩展, 如速率增加, 可驱动快速 MODEM 和“Laplink”应用软件, 支持 115k 和 230k 速率, 其总线接口也有所改善。在此基础上产生了 16450 UART, 它是 8250 的直接扩展, 但是, 即使以这样的速度, PC 机的中断和软件响应时间仍显得不够, 在 115k 速率时, 每字节近 100us, 20us 的中断和 30us 的缓存, 需占用 50% 的 CPU 时间, 这在实际应用中是难以接受的。

下一个发展阶段就是在 UART 中增加硬件缓存, 其代表性产品 16550 在功能上比 8250 多 16 字节的接收、发送 FIFO, 后来 FIFO 增加到 32 字节 (16C650) 和现在的 64 字节 (16C750), 因为 FIFO 总是最后服务, 大容量的 FIFO 可减少返回次数和提高速度。新一代的 UART 将实现智能通信, 目前已经开始在 PC 卡上出现。

由于价格的压力、VLSI 的出现, 80 年代产生了超级 I/O, 这类器件包括两个 UART, 并行打印口和其它标准的 PC I/O, 速率可达到 490k 甚至 920k。从内部结构看, 它实际上是在 16550 的基础上增加了寄存器。最新的超级 I/O 带有 IrDA 时间模型用于 IR 串行通信, IrDA 最初用于掌上型设备, 目前已在许多简单的无线接口设备 (如打印机、付费电话等) 中得到应用。尽管其它高速接口 (USB) 已经出现, 但标准的带有 RS-232 接口的 UART 在相当长的时间里是不会从 PC 机中消失的, 因此, 关注下一步 UART 在 PC 机中的应用技术是非常有意义的事情。

非 PC 系统中的 UART:

PC 是 UART 的主要市场, 在非 PC 系统中, 一般其主要组成部分仍然是 PC, 系统间的通信也由 PC 间接带动, 而且要求 UART 与 PC 保持同步, 因此, 非 PC 系统在价格、功耗等方面对 UART 也提出了新的要求, MAXIM 公司发现 UART 在这方面的应用并不完善, 例如: 电信及工业设备等大的非 PC 系统中, 非常普遍的做法是采用标准 PC UART, 为满足要求, 8250 和 Philips2691 已扩展到 2 个、4 个甚至 8 个 UART, Zilog8630 因其较高的速度而成为这一市场的主导产品, MOTOROLA 的 683 系列微控制器也有一部分市场, 这类器件内部包括不同的

外围电路、68000 内核等，有些器件对于复杂通信非常灵活，目前，在用户 IC 中也有采用 UART 的趋势。UART 具有相对综合的逻辑功能，并用现代的 EDA 工具集成在硅片上，因而，这种片上系统模型对于大规模数字系统越来越有吸引力。

对于小的非 PC 设备（如：MODEM、工业设备等），UART 必须与无处不在的 PC 以及小的工业网络通信，早在 80 年代初期，全功能的微控制器（80186、8051、68HC11、Z8 等）已包含内部 UART，这些内部 UART 可满足中低速的应用，尽管制造商增加了时钟速度、ROM 容量及其它功能，但微控制器中 UART 的速度和功能没有多大改善，仅有少数产品除外，象 Dd11as 的 80C320 包含两个 UART，Intel 的 80C51FA 具有 9 位网络地址（后面描述）。

当需要采用多个 UART 时，设计者有两种选择：①如果需要低性能的 UART，并且 ROM 容量和 CPU 时间允许，可由软件实现 UART，与硬件交替使用，在后面章节将进一步讨论这一问题。②另加一个 UART，大多情况下，这个 UART 是无法接受的，它们需要大尺寸、大功耗、高价格，更重要的是需要复杂的软件以获得好的性能。如果真正需要 IrDA 功能，设计者可以在 PAL 内实现 IrDA 时间发生器，并输出到微控制器外部的 UART（支持 115K 波特的 IrDA 速率），IrDA 时间片出现后即取代了 PAL，但仍需要外部 UART。

对低功耗系统（手持工业设备，条形码阅读器，测试设备和消费产品）经常用很小的微控制器。低价格低功耗的微控制器（Microchip 的 PIC16C54 和 Motorola 的 68HC05J2）都没有 UART，这些系统实现串行通信由软件完成，这占去了 CPU 的大部分时间。如果由于带宽和性能的原因，需要一个硬件 UART，可能会用一个高档微控制器带有 UART，如果微控制器内的 UART 功能不够完善，将不得不用一个全功能的 UART。这类系统设计很难选择。如果量大，设计者会研制适合自己的 UART，如果量少这一功能会被取消。

市场上的 40 种 UART 对另一类应用也不满足。很多 DSP（TMS320C10）没有 UART，用软件实现有一个严重的问题，DSP 一般为同步系统而对异步串行中断的响应很困难。

现代小型 UART 的市场需求

MAXIM 看到了这一市场需求和机会。虽然 UART 基本上是数字电路，而 MAXIM 是一家线性和混合集成电路的公司，但 MAXIM 在串行接口方面有很多经验（如 MAX232, MAX485），MAXIM 认识到对 UART 的需求主要有以下几方面：

- 支持高速（微控制器不能达到这一目标）
- 支持低功耗、低电压。
- 小尺寸且带有波特率发生器等功能。
- 需功耗关闭模式并能被唤醒。
- 支持 IrDA 通信。
- FIFO 接收缓冲减轻微控制器的通信负担。
- 高驱动和施密特触发器输入与隔离光耦直接接口。
- 低价格。

依照这些要求，MAXIM 努力的结果就是 MAX 3100，一个全功能的 UART（后面详细讨论）。按照小尺寸的要求，MAX 3100 采用了同步串行接口，串口到串口听起来是矛盾的，但这正是 MAX 3100 的特殊之处，它将一个全功能的 UART 封装在 8 脚的 SO（QSOP-16）内，与 SPI 兼容的串行接口极易与微控制器接口，以极小的尺寸、较低的价格和功耗即可获得高性能（这些后面还有描述）。

### 3. 软件实现 UART 的缺点

内部没有 UART 的微控制器进行通信的简单方法是通过软件实现，不需外部硬件，软件实现不存在任何问题，真正的花费是 CPU 的时间。除了很简单情况，软件实现 UART 要比硬件实现价格高。

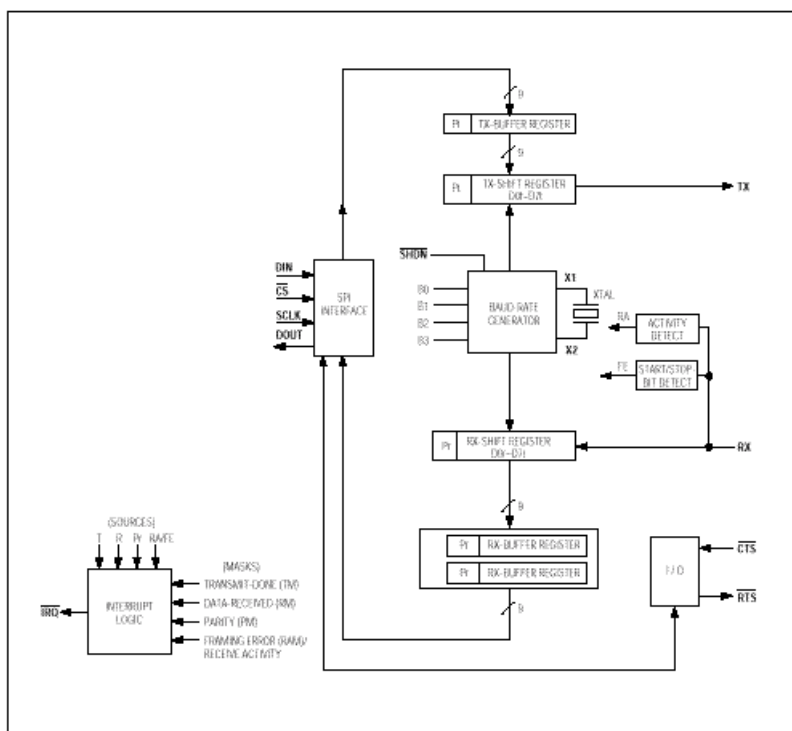
软件实现 UART 时，利用计数器/时间发生器将串行数据位分开，时间发生器是微控制器的宝贵资源。串行输入输出至少需要两个 I/O 口，串行输入 (RX) 应有中断能力，以使软件同步数据，这一中断使系统的设计更为复杂，因最大中断时间必须小于半个数据位时间才能可靠接收，高波特率时，将使微处理器无法工作，当需要握手信号时，还需要多条 I/O 口线，此外，依照处理器和 UART 的复杂程度还需 200-500 字节的编码，而很多小控制器仅有 500 字节。有些情况下，唤醒时间大于波特周期而导致处理器连续运行，还可造成较大的功率损耗。

相比之下，MAX3100 仅需要 4 条口线实现握手接口，有一条中断线可选，多数情况下这条中断线不需选用。此外，MAX3100 仅需要 50 字节编码，处理器和 UART 在等待串行事件时休眠，可以节省大量功耗，不需随波特率而改变时间。MAX3100 的 FIFO 还可缓解很多小系统中的突发信息。

#### 4. MAX3100 使用说明

MAX3100 UART 实现 uP 的同步串行数据转换到异步串行通信口 (RS-232、RS-485、IrDA)，图一是 uP 的四条同步串行接口 (与 SPI、QSPI、Microwire 兼容) 和 MAX3100 的通用异步接收发送器 (UART)，附录给出了 SPI 的描述。

MAX3100 包括一个简单的 UART，带有 SPI 接口的波特率发生器和一个中断发生器，通过“写结构寄存器”完成波特率、字长、校验、8 字节接收 FIFO 的设定并选择通用 UART 或 IrDA，控制关闭状态和四个中断任务。



图一 MAX3100 UART 框图

图一为功能框图，片上时钟 X1 可以是 1.8432MHZ, 3.6864MHZ 或由占空比为 45%—55%的方波驱动 X1，片上时钟除以 16 是波特率时钟，B0—B3 四位确定波特率系数 (BRD)，波特率时钟除以 BRD 即为波特率，波特率的范围是 300—230K。

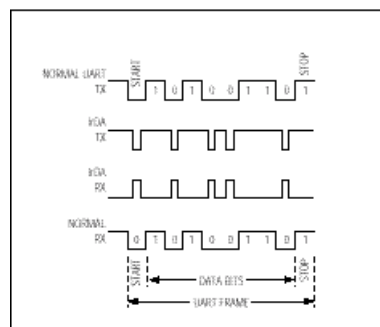
发送部分接收 SPI 数据，通过 TX 按异步串行发送出去，SPI 的数据进行发送缓存，MAX3100 加上起始位和停止位，并按要求的波特率发送。

接收部分接收异步串行数据，MAX3100 确定由高变低为起始位，确定方法是以 16 倍波特率时钟采样，按第 7、8、9 次的结果确定起始位，接收器在停止位的中间开始寻找下一个起

始位。

MAX3100 允许直接光耦输入输出，接收器有一个施密特触发器，发送器可以吸收 25mA，还有两个光耦用于握手信号（RS-232）或控制（RS-485）。

CPU 的计算时间包括 MAX3100 的 8 字节 FIFO 和中断逻辑。CPU 的传输时间被缓冲，当 CPU 服务接收中断时 FIFO 准备 8 字节开始传送。MAX3100 有一个中断由 4 个源设置：校验（Pr），接收数

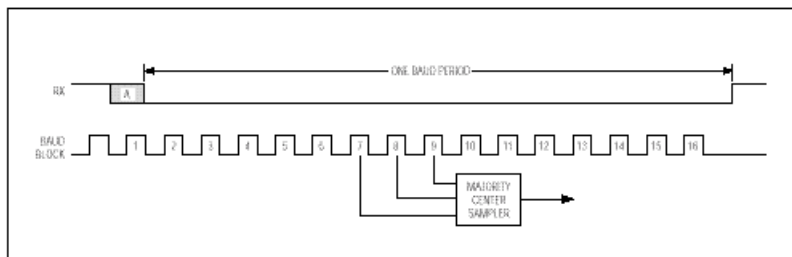


图二、起始位

据（R），接收开始/帧错（RA/FA）和发送缓存空（T），每个中断可以被屏蔽。

这个 SPI UART 的另一特性是 IrDA 模式，可以与 SIR 兼容的设备通信，或者在光隔离应用中降低功耗，MAX3100 设计为直接驱动光隔离设备，所以驱动串行 IR（象 HP 的 HDSSL-1000）时需倒相。

IrDA 模式时，一位的周期缩短到 3/16 波特周期，数据零对于发送引脚就是逻辑低，对于接收引脚就是逻辑高，见图 3。



图三 IrDA 时间

接收模式时，接收信号的采样是在发送高电平的一半时进行，采样一次完成，而普通模式是采样三次完成。

IrDA 器件与 MAX3100 通信必须设置发送脉冲为波特周期的 3/16，MAX3100 忽略小于波特周期性 1/16 的脉冲。

## 5. MAX3100 开辟的新应用。

### A. 能够使小微控制器实现 IrDA 或高速。

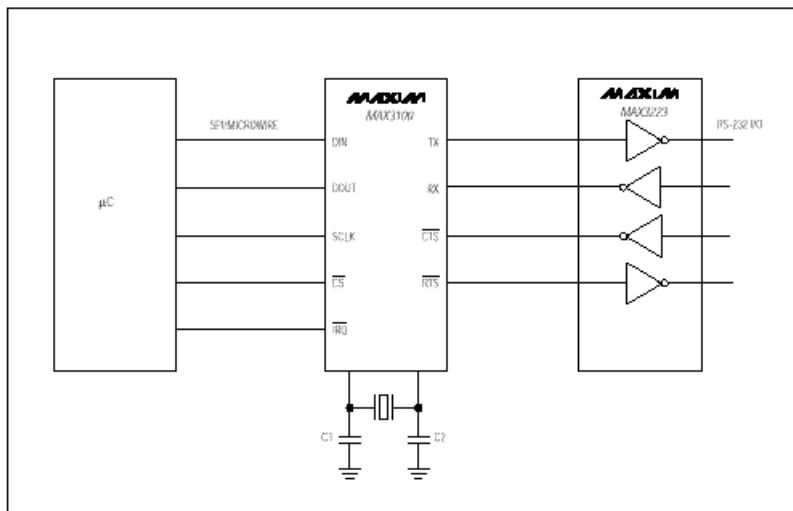
图 4 的电路用 MAX3100 UART 和常用的一种微控制器实现 IrDA SIR 方式或高速方式通信。微控制器到 UART 的接口用 SPI 串行接口，附录里给出三种微处理器的程序例子。虽然它们内部有 UART，但不与 IrDA 兼容，而且没有办法使微控制器实现这一功能。

MAX3100 的最高速率是 115KBD，但由于光耦而限制在 4800BD，如果采用高性能的光耦（HP HDSSL-1000），很容易达到 SIR 的满速率，但应知道很多 IrDA 设备支持低到 2400BD，而且这样的器件很便宜，几乎所有的光电二极管都可以用，但要滤除光电二极管周围的可见光，或者用外部 IR 滤波器放在光电二极管的前端。

处理器的工作电压从 2.7V 到 6.0V，很多小的微处理器每 1MHZ 耗电 1mA，如果实时处理由 UART 完成，处理器时钟可以不讲究，只要 UART 有一个稳定的时钟，而 MAX3100 耗电仅 1mA。

这个电路有几个特点，由软件完成的 IrDA UART 程序，活动时占 100% 的 CPU 时间，而且不能高于 2400BD 的速率。分离逻辑 PAL 可以产生 IrDA 时间，但其价格很贵，功耗大，需要外部波特率发生器时钟源。图 4 简单的电路能够提供 IrDA 通信，而且价格低，

编程简单，功率低，另请注意 MAX3100 也能够实现 RS-232 通信。



图四 系统框图

附录是 MAX3100 对 6805, PIC 和 8051 的源程序。

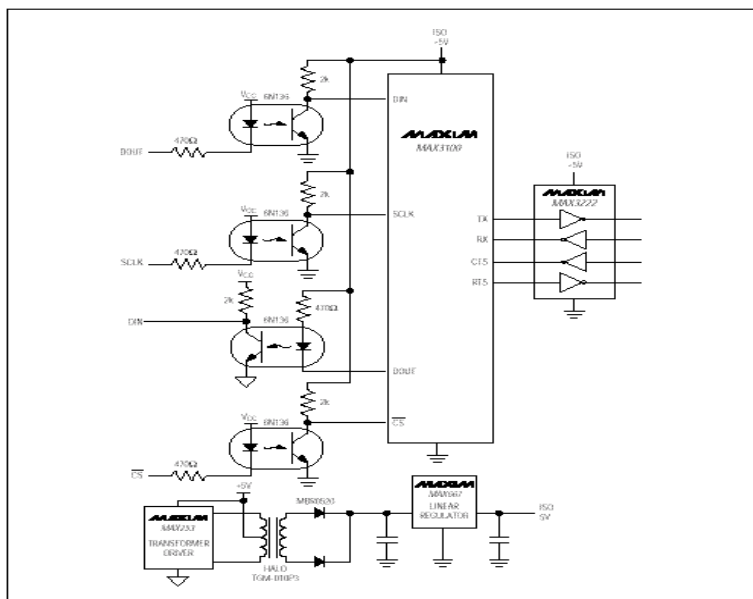
B. 9 位网

MAX3100 支持普通多点通信称为 9 位模式。奇偶位用来指示地址信息，这种情况 MAX3100 可以产生中断，这种方法可以使从控制器不用处理很多与自己无关的信息。这可以让远端处理器处理更多有用的工作。

9 位模式时，MAX3100 设置为 8 位加校验，校验位设置为地址信息，网络中，主机发送带有地址信息，所有节点收到后判断地址信息，是自己的信息就处理。

奇偶位/第 9 位中断只受接收寄存器控制，不受 FIFO 内数据的影响，所以第 9 位中断的有效应用是 FIFO 失效，FIFO 失效时，收到的非地址字可以忽略掉，不用从 UART 读出。

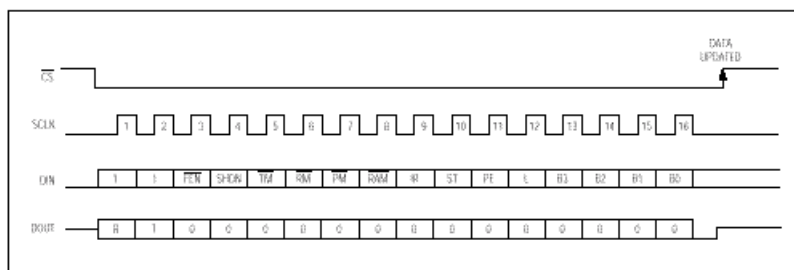
B. 隔离应用



图五 驱动光耦

图 5 是 MAX3100 用在隔离串行接口，MAX3100 的施密特触发器输入直接由光耦输出驱动，这个电路的一个特点是光耦不影响异步串行输出的时间，只需考虑 SPI 接口的建立和保持时间。

如果需要双向隔离仅需两个光耦，用 IrDA 的 3/16 宽波特周期可以节约 81% 的功率。



图六 SPI 时间图

附件：串行接口

MAX3100 是串行结构，这可以减少管脚，它与 Motorola 的 SPI 和 QSPI 以及国半的 Microwire 接口兼容，下面简单描述这些接口。

Microwire 和 Microwire Plus (国半的专利)。

Microwire 是用在国半 COP 控制器上的串行接口，包括时钟线，数据进出线和片选线。最大时钟速率 250KHZ，数据定义在时钟的下降沿，片选不用时为高。Microwire Plus 用在国半的 HPC 控制器上，将时钟相位对输入数据和输出数据倒相，可以加速接口时间。

SPI 和 QSPI (Motorola 的专利)

SPI 与 Microwire 非常相似，用在 Motorola 的控制器上，它也包括时钟线，数据输入输出线，片选线。最大时钟比 Microwire 高可达 2M bps，对 QSPI 超过 10M bps。QSPI 内有 16 级硬件控制，自动片选，和 SPI 处理。对外部器件，两种器件相同，数据时钟输入输出由 CPHA 位控制，CPOL 位控制时钟相位和希望的极性。

图一 MAX3100 UART 框图

图二 起始位

图三 IrDA/通用时间

图四 系统框图

图五 驱动光耦

图六 SPI 时间图

### 8051 软件清单

- ； 基于 8051 的 UART—MAX3100 实现 IrDA 通信的代码
- ； MAX3100 实现双向通信，IrDA 到 RS-232 和 RS-232 到 IrDA。
- ； 8051 和 MAX3100 两侧均为 9600BD 的速率。IrDA 由掌上型的 HP-100 完成，
- ； PC 运行终端程序。

PCON EQU 87H

； 口线定义

DOUT BIT P1.0 ;数据从 UART 输出  
 DIN BIT P1.1 ; 数据从 UART 输入  
 SCLK BIT P1.2 ; 串行时钟  
 CS BIT P1.3 ; 片选，低电平有效  
 IRQ BIT P3.2

```

;RAM
TX1    EQU    10H;    发送
TX2    EQU    11H
RX1    EQU    12H;    接收
RX2    EQU    13H
; *****
        ORG    0H
BEGIN:   MOV    SP, #70H    ; 初始化堆栈
        CLR    SCLK        ; 清时钟
; 初始化 8051 内部 UART
        MOV    TMOD, #20H    ; t1 波特率
        MOV    TH1, #253     ; 9600BD
        MOV    SCON, #50H    ; UART—ml, tx, rx
        MOV    PCON, #80H
        MOV    TCON, #40H
; 初始化 MAX3100—UART—IrDA 模式—9600BD
        MOV    TX1, #0E4H
        MOV    TX2, #0CAH
        CALL   UTLK          ; 发送到 UART
; *****主循环程序*****
LOOP:   JNB    IRQ, URCV     ; 数据有效?
NRECV:  JBC    RI, RCV51     ; 检测 8051 的接收
        JMP    LOOP
; 从 3100 收到的字节
URCV :   MOV    TX1, #0       ; 读数据
        MOV    TX2, #0       ; 读数据
        CALL   UTLK          ; 发送到 8051
        MOV    A, RX2        ; 读数到累加器
        MOV    SBUF, A       ; 发送到 RS-232
        JMP    LOOP         ; 从头开始
; 收到 8051 的字节
RCV51:  MOV    A, SBUF        ; 读取 8051 的 UART
        MOV    TX1, #80H     ; 发送数据
        MOV    TX2, A        ; 数据到 IrDA
        CALL   UTLK          ; 发送到 UART
        JMP    LOOP         ;
; *****
; 子程序 UTLK
UTLK:   CLR    CS            ; 使能 CS
        MOV    A, TX1        ; 读高字节
        CALL   BYT8          ; 发送
        MOV    RX1, A        ; 读接收 1
        MOV    A, TX2        ; 读高字节
        CALL   BYT8          ; 发送

```

```

        MOV     RX2, A           ; 读接收 2
        SETB   CS               ; CS 取反
        RET

; *****
; BYT8—8 位位移
BYT8:   MOV     R4, #8          ; 8 位发送
        SETB   DIN            ; 确定 DIN 是输入
B8LP:   RLC     A              ;
        MOV     DOUT, C        ;
        SETB   SCLK           ; 确定时钟高
        MOV     C, DIN         ; 时钟变高读数据
        CLR    SCLK           ; 清时钟
        MOV     ACC.0, C       ;
        DJNZ   R4, B8LP       ; 8 位循环
        RET
        END

; *****

```