OD2101 芯片应用详解

一、 概述

OD2101 芯片为 14 脚TSSOP封装的CMOS器件，是一款提供I^2C转UART接口方案的专用协议转换芯片。该芯片可以方便用户进行I^2C接口的扩展，将数据与RS232，RS485 总线进行透明传输。当设计中需要扩展微处理器的串口或希望在单独I^2C总线上与UART器件进行数据交换，使用该芯片可以简单的实现方案。

OD2101 芯片具有低功耗（关断状态下功耗可降到 10μA以下），内部复位，内部晶振等特点，用户可以用非常少的引脚和外围电路就可以实现芯片工作。一条I^2C总线可以最多支持 8 个OD2101 芯片。

二、 引脚及说明

采用 14 脚 TSSOP 封装，引脚图如图 1 所示。其管脚定义如下：

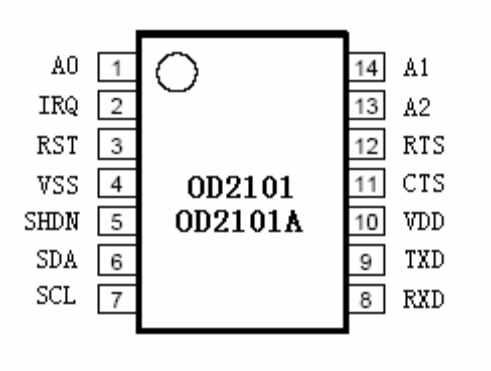

图 1 管脚配置-TSSOP

管脚描述

管脚号	符号	功能
1	A0	I^2C地址输入 0
2	IRQ	中断输出（开漏极）低有效
3	RST	复位管脚（需要通过上拉电阻 5~10K 接 VDD）
4	VSS	地
5	SHDN	硬件关断输入脚。当关断产生（SHDN 置高），芯片停止工作，所有接口变成高阻态，系统进入节电模式。正常工作需要接低电平。
6	SDA	串行数据线

7	SCL	串行时钟线
8	RXD	串行口输入
9	TXD	串行口输出
10	VDD	电源：2.4V~3.6V
11	CTS	低有效输入管脚。通过 CTS 寄存器读取。通常用在 RS-232 的清除输入功能。
12	RTS	低有效输出管脚。通过 RTS 寄存器控制。通常用在 RS-232 的输出请求或 RS-485 的驱动使能。
13	A2	I^2C地址输入 2
14	A1	I^2C地址输入 1

表 1 管脚描述

三、 功能详述

下面我们根据一个单片机与 OD2101 芯片通讯的实例，一步步说明如何使用这颗芯片。

电路及说明

本实例采用 8051 单片机与 OD2101 芯片进行通讯连接，相关电路如下图所示：

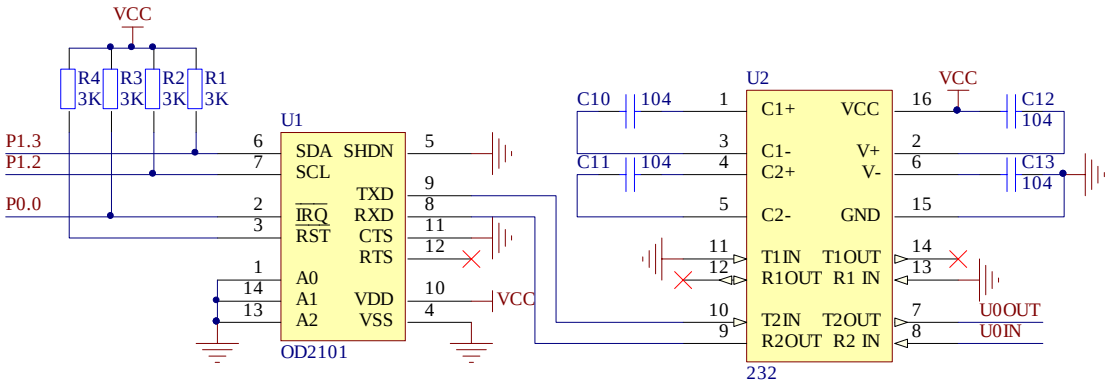

图 2 OD2101 电路连接图

电路连接图采用 8051 的P1.3 和P1.3 模拟I^2C接口，P0.0 作为中断输入信号。A0、A1、A2 管脚都接地，因此OD2101 的从机地址为 0x50。我们可以通过改变 A0、A1、A2 管脚的高低点平来改变OD2101 的从机地址，从而实现控制多个 OD2101 的目的。如A0、A1、A2 管脚高低点平为（100），OD2101 的从机地址即为（0x50+0x02=0x52）。

模拟I²C软件库说明

模拟I²C软件包共有两个文件，分别为VIRTUAL_I2C.C和VIRTUAL_I2C.H。此软件库主要针对 12M晶振的 8051 单片机设计，用户如果采用其他微处理器，可以修改相关部分进行改进。软件包的I²C接口定义如下：

```
sbit SDA=P1^3;                  /*模拟 I2C 数据传送位*/
sbit SCL=P1^2;                  /*模拟 I2C 时钟控制位*/
```

软件库集成了I²C主机模式下的函数集，由于OD2101 的操作需要从机地址和硬件子地址。我们在这里主要使用如下两个函数：

```
/*****************************************************************
                    有子地址发送多字节数据函数

功能:       从启动总线到发送地址，子地址,数据，结束总线的全过程,从器件
            地址 sla，子地址 suba，发送内容是 s 指向的内容，发送 no 个字节。
            如果返回 1 表示操作成功，否则操作有误。
*****************************************************************/
extern bit ISendStr(uchar sla,uchar suba,uchar *s,uchar no) ;
/*****************************************************************
                    有子地址读取多字节数据函数

功能:       从启动总线到发送地址，子地址,读数据，结束总线的全过程,从器件
            地址 sla，子地址 suba，读出的内容放入 s 指向的存储区，读 no 个
字节。
            如果返回 1 表示操作成功，否则操作有误。
*****************************************************************/
extern bit IRcvStr(uchar sla,uchar suba,uchar *s,uchar no);
```

寄存器读写

关于 OD2101 芯片寄存器设置，请参考芯片说明文档。程序中设置寄存器地址如下：

```
#define WR          0x00        /*数据读寄存器*/
#define RD          0x00        /*数据写寄存器*/
#define UARTBUF     0x01        /*UART 接收缓存接收字节数*/
#define I2CBUF      0x02        /*I2C 可加载字节数*/
#define CTRL        0x03        /*UART 接口控制寄存器*/
```

需要注意的是，我们在利用这些寄存器读写数据的时候，这些地址都为硬件子地址。在I²C主机控制读写数据时的顺序应为：（从机地址R/W） + 硬件子地

址 + 数据….。这种带有硬件子地址的I^2C读写顺序与普通只有从机地址的I^2C读写顺序有些不同，相关资料请查询I^2C总线协议。我们在利用I^2C库文件读写就比较方便，例如我们向I^2C接口发送“ABCDE”这几个字符，就可用如下方式编写：

```
ISendStr( OD2101 , WR , "ABCDE", 5);
```

通讯接口的设置

OD2101 通讯接口的设置主要包括：波特率的设置、CTS 和 RTS 的选择设置、数据清空设置选项。

波特率设置

OD2101 的 UART 波特率设置由控制寄存器的 0~4 位来决定，我们在写入控制字后，波特率立即生效，如我们设置 9600 的波特率如下所示：

```
IRcvStr( OD2101 , CTRL , &in_dat , 1 ) ;  /*读取 OD2101 控制寄存器*/
in_dat = ( in_dat & 0xf0 ) | UART_9600 ;  /*设置 OD2101 波特率为 9600*/
ISendStr( OD2101 , CTRL, &in_dat , 1) ;  /*发送控制字节*/
```

OD2101 的 UART 波特率为固定波特率选项，如下表所示：

波特率 B3 B2 B1 B0	波特率	波特率 B3 B2 B1 B0	波特率
0 0 0 0	9600	1 0 0 0	4800
0 0 0 1	300	1 0 0 1	7200
0 0 1 0	600	1 0 1 0	14400
0 0 1 1	900	1 0 1 1	19200
0 1 0 0	1200	1 1 0 0	28800
0 1 0 1	1800	1 1 0 1	38400
0 1 1 0	2400	1 1 1 0	57600
0 1 1 1	3600	1 1 1 1	115200

表 2：波特率对照表

程序中，波特率对照表如下所示：

```
/*定义 OD2101 波特率*/
enum{ UART_9600,UART_300,UART_600,UART_900,UART_1200,
      UART_1800,UART_2400,UART_3600,UART_4800,UART_7200,
      UART_14400,UART_19200,UART_28800,UART_38400,
      UART_57600,UART_115200 };
```

CTS 和 RTS 控制

CTS 和 RTS 是串行通讯中流控制的两个管脚，本身成对出现。一般连接方式如下图所示：

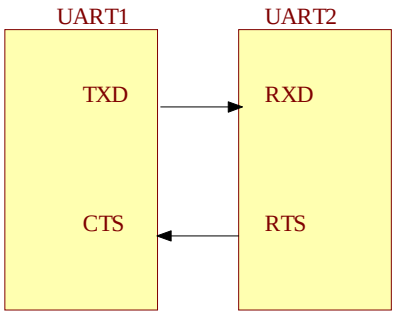

图 2 UART 流控制示意图。

具体地说，就是当接收方（UART2）认为可以接受 UART 数据时，将 RTS 置为有效位。发送方（UART1）的 CTS 管脚接收到此信号后，才将数据进行发送。一般来说，当 UART 数据接收缓存大于两个字节空余时，就可以将 RTS 置为有效接收。当然，OD2101 的 UART 接收缓冲区具有 64 位的字节，用户可以按照自己数据流量的需求定义 RTS。CTS 和 RTS 的置位和读取如下代码所示：

```
IRcvStr( OD2101 , CTRL , &in_dat , 1 ) ;  /*读取 OD2101 控制寄存器*/
if( in_dat & (1<<4) )       /* CTS 管脚为低    */
else                        /* CTS  管脚为高  */
```

```
IRcvStr( OD2101 , CTRL , &in_dat , 1 ) ;  /*读取 OD2101 控制寄存器*/
in_dat |= (1<<5)   ;   /*设置 RTS 为低，置高为 in_dat &= (~(1<<5)) ;*/
ISendStr( OD2101 , CTRL, &in_dat , 1) ;  /*发送控制字节*/
```

数据清空设置

数据清空设置比较简单，只要把控制寄存器相应位置位并发送，相关接口的数据缓冲区的数据立即被清空。

关断功能

当SHDN管脚置高时，芯片进入关断功能状态。这时所有通讯接口均进入高阻态，系统进入低功耗状态。除了SHDN，其他触发都不能改变芯片状态。当SHDN重新置低后，系统恢复正常工作，所有寄存器恢复到关断前状态（I^2C和UART数据缓冲区数据全部清空）。

例程试验

本试验例程首先设置OD2101 的UART波特率为 9600，然后发送“good\n”5个字符到I^2C接口，用户可以通过OD2101 的UART口接收观察数据是否正确发送出去。在大循环中，程序轮询IRQ管脚是否产生中断（建议在开发中采用硬件中断方式），当中断产生时，接收I^2C数据，并进行大小写转换，然后将传唤后的数据发送回OD2101。相关试验程序如下：

```
IRQ = 1 ;          /*将 IRQ 管脚设置成输入状态*/

IRcvStr( OD2101 , CTRL , &in_dat , 1 ) ;      /*读取 OD2101 控制寄存器*/
in_dat = ( in_dat & 0xf0 ) | UART_9600 ;      /*设置 OD2101UART 波特率为 9600*/
ISendStr( OD2101 , CTRL, &in_dat , 1) ;       /*发送控制字节*/

ISendStr( OD2101 , WR , "good\n", 5);         /*发送"good"字节到 OD2101*/

while(1)
{
      if( !IRQ)                               /*判断是否有接收中断产生*/
      {
            DelayNs(10) ;                     /*延时过滤杂波*/
            if( !IRQ )
            {
                IRcvStr(OD2101,UARTBUF,&in_dat,1) ;/*UART 接收缓存接收字节*/
                for( i = 0 ; i < in_dat ; i++ )          /*将接收缓冲区的数据读出*/
                {
                    IRcvStr( OD2101 , RD , &in_dat1 , 1 ) ;  /*读取一个字节数据*/
                    /*将读取出的数据进行大小写转换*/
                    if( (in_dat1>='A')&&(in_dat1<='Z') )
                    {
                       in_dat1 += 'a'-'A' ;
                    }
                    else if( (in_dat1>='a')&&(in_dat1<='z') )
                    {
                       in_dat1 -= 'a'-'A' ;
                    }
                ISendStr(OD2101, WR, &in_dat1, 1);/*将转换后的数据发送给 I2C 总线*/
                }
            }
      }
  }
```

总结

OD2101 是一款简单实用的I²C转UART协议转换芯片，用户可以在缺少UART或需要扩展多串口的场合下使用此芯片。考虑到I²C接口的地址重复问题，如果OD2101 地址被占用，客户也可以选择OD2101A（从机地址为 0x70）使用。本次试验采用的是模拟I²C接口编程，实际使用中强烈建议客户采用硬件I²C接口进行，因为毕竟UART的软件模拟要比I²C容易的多，何必多此一举呢!:）

技术支持：support@firstpower.cn