优胜电子科技
USEN ELECTRONIC TECHNOLOGY

USB 串口转 I2C Master 使用手册

USB Serial Port to I2C Master

适用产品: SER2I2C/HID2I2C

2017.12.25

目 录

前言

简介

前　言

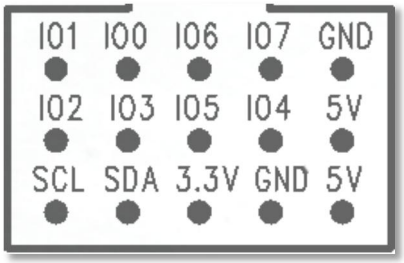
转换器是一款I2C Master产品，可通过USB串口读写控制I2C Slave，通过本转换器可以直接在任意串口助手软件上和I2C设备通信，作为I2C Master控制I2C总线所有时序。在主机端只需要几个简单的操作命令就可以读写I2C设备，控制起来简单容易。同时，本转换器还支持GPIO扩展，支持8个GPIO，可通过USB配置GPIO的状态，达到控制外围设备的目的。

简　介

SER2I2C 线上购买地址：https://item.taobao.com/item.htm?id=563493747947

HID2I2C 线上购买地址：https://item.taobao.com/item.htm?id=563248564507

转换器尾部插针信号对应图如下（背面有丝印标注）



转换器GPIO接口说明：

信号	类型	说明
IO0~IO7	输入/输出	GPIO,可配置为输入或输出
GND	电源	电源地
5V	电源	5V电压输出，取决于USB接口供电能力

转换器I2C接口说明：

信号	类型	说明
SCL	输出	I2C总线时钟
SDA	输入/输出	I2C总线数据
3.3V	电源	3.3V电压输出，约100mA
GND	电源	电源地
5V	电源	5V电压输出，取决于USB接口供电能力

一、 驱动

转换器支持在Windows, Linux, Mac OS , Android等各大主流系统中使用。

驱动程序下载：

1、Windows系统，包含WIN10 -32/64位
下载地址：http://www.wch.cn/downfile/65

2、 安卓免驱应用库API
下载地址：http://www.wch.cn/downfile/195

3、 Liunx系统
下载地址：http://www.wch.cn/downfile/177

4、 苹果MAC OS
下载地址：http://www.wch.cn/downfile/178

二、 指令

转换器支持的串口格式为：1 起始位，8 数据位，1 停止位，无校验位。初始上电，默认波特率为 9600bps，其它波特率可以通过调整内部寄存器值来配置。

在主机端，可以通过 RTS 复位转换器，复位后，转换器会连续发送两个字节的数据给串口显示复位状态，这两个字节分别是 0x4F 0x4B，即 ASCII 字符 OK.(标准版不支持)

转换器支持的 ASCII 命令如下表：

ASCII command	Hex value	Command function
S	0x53	I²C-bus START
P	0x50	I²C-bus STOP
R	0x52	read internal register
W	0x57	write to internal register
I	0x49	read GPIO port
O	0x4F	write to GPIO port

在电脑串口助手中使用这些命令的 HEX 值来操作，串口助手软件收发数据在 HEX 模式下进行。

为防止因为命令输入不完整，转换器有超时功能，主机发送命令数据两个字节延迟必须在655ms以内，否则超时自动清除接收区，等待新的命令输入。

三、 操作实例

读写操作实例主要在与I2C Slave设备YS201X之间的通信进行。YS201X是一款可以转成串口的I2C Slave设备，采用原装进口FTDI芯片设计，可以访问http://usendz.taobao.com购买。

3.1 写N字节数据到Slave设备

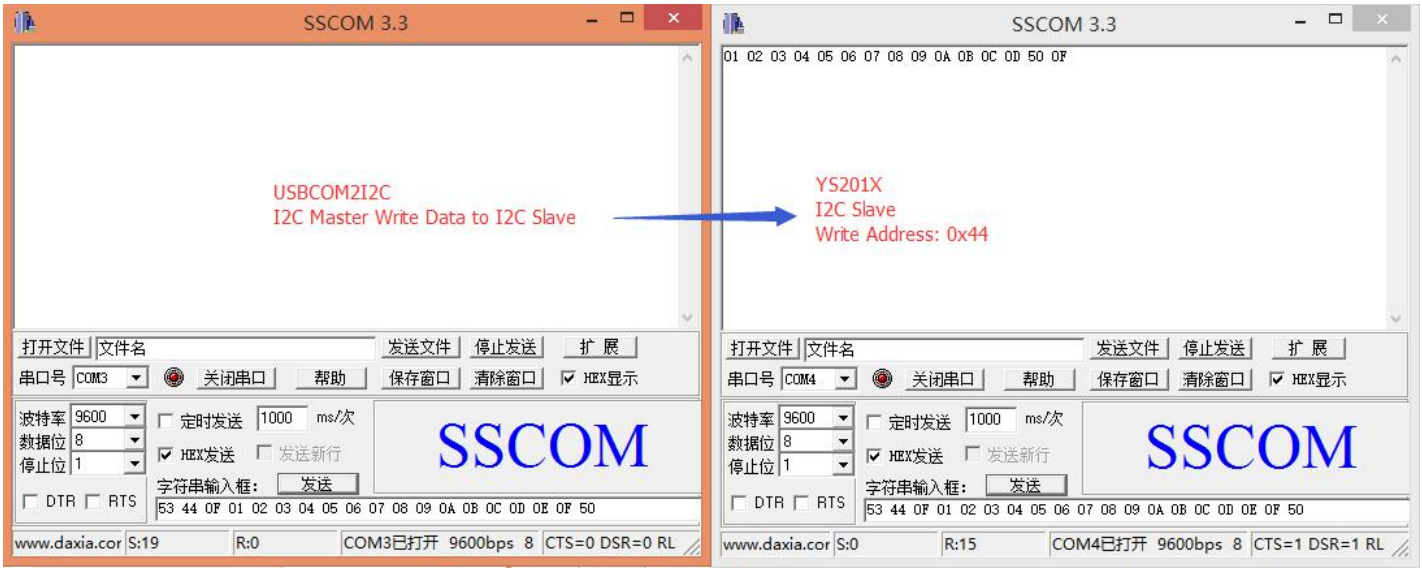
主机发送命令格式如下：

YS201X的从设备地址为0x44，那么要写 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0F 共15个字节数据到从机YS201X操作如下：(注：0x前缀表示该数据是16进制，在串口助手中操作时不写)

```
53 44 0F 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 50
```

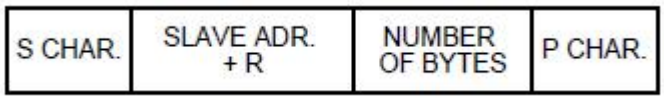
这里 53 为 S 命令的 HEX 值， 44 为 YS201X 的写设备地址（读写位=0），0F 为要写的字节数（15 的十六进制），01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 为要写出去的 15 个数据，50 为 P 命令的 HEX 值。

点击发送即可，如下图：

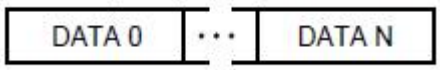


3.2 从Slave设备读取N字节数据

主机发送命令格式如下：



主机收到数据形式如下：



同样以 YS201X 为例:

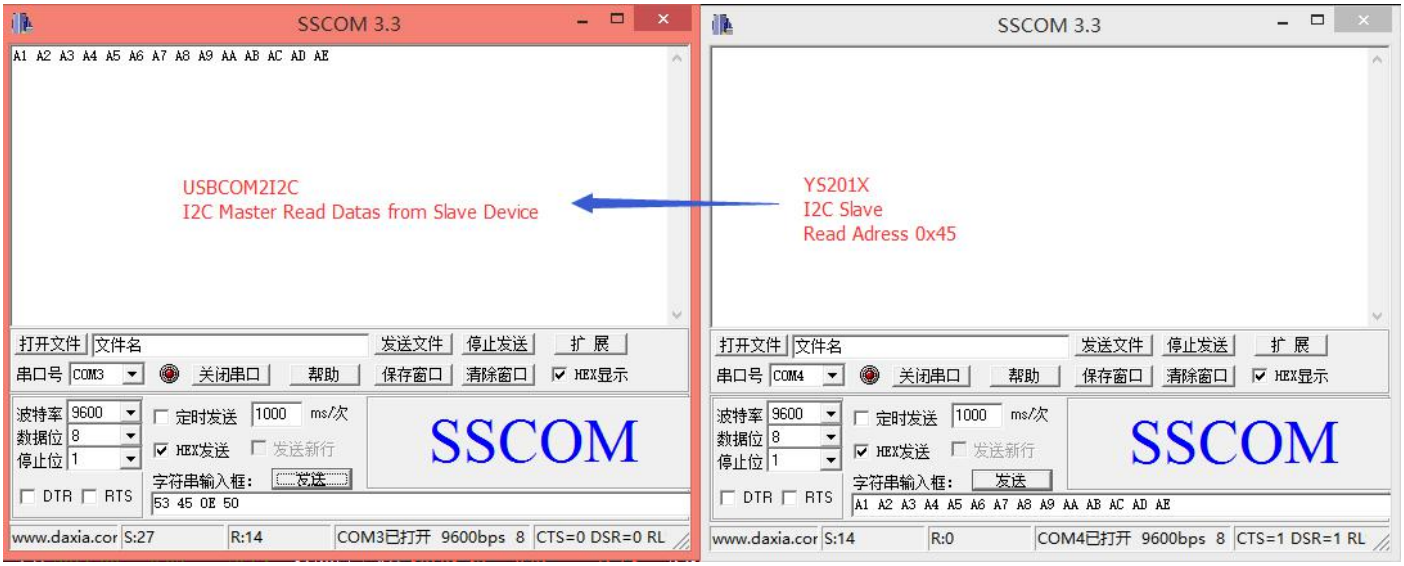
先把数据写到 YS201X 内部缓存，待 I2C 主机读取。

发送 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE 共 14 个数据到 YS201X，转换器从 YS201X Slave 读取这 15 个数据的命令如下：

```
53 45 0E 50
```

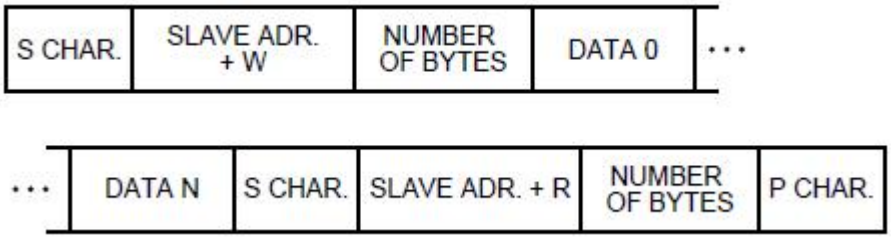
这里 53 为 S 命令的 HEX 值， 45 为 YS201X 的读设备地址（读写位=1），0E 为要读的字节数（14 的十六进制），50 为 P 命令的 HEX 值。

点击发送，结果如下图：

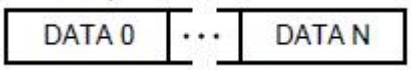


3.3 写了读

主机发送命令格式如下：



主机收到数据格式如下：

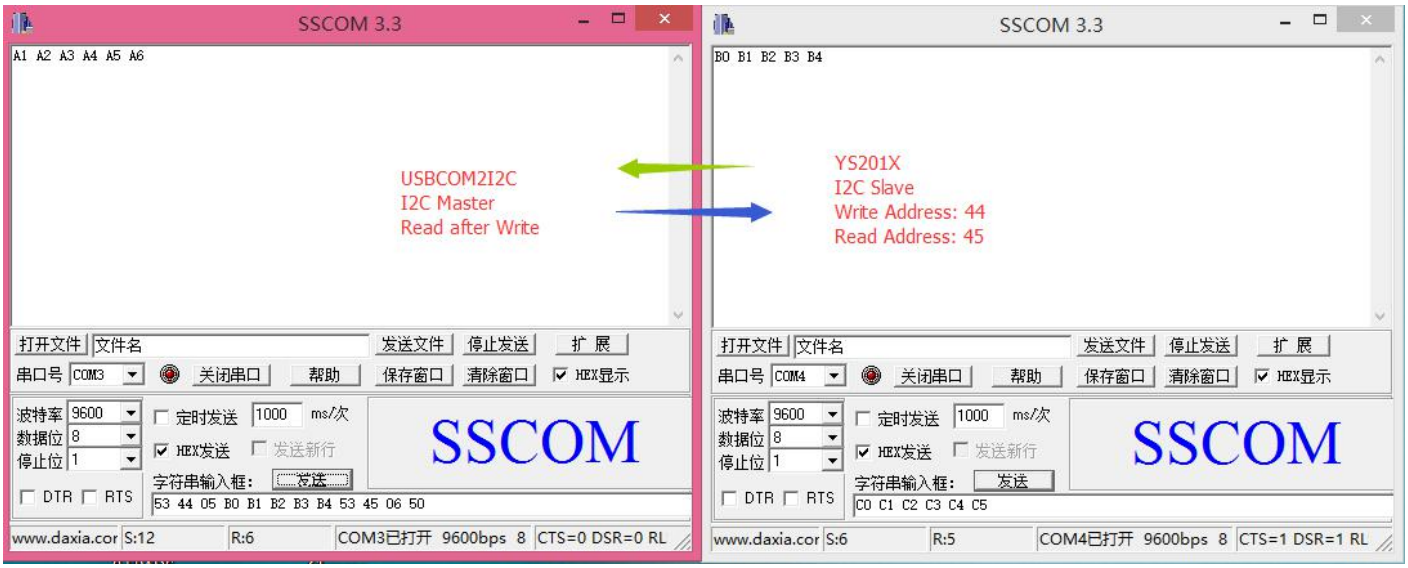


还是以与YS201X通信为例：发送数据0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 六个数据到YS201X备Master读取；Master向Slave写5字节数据0xB0 0xB1 0xB2 0xB3 0xB4后不停止接着读，命令操作如下：

```
53 44 05 B0 B1 B2 B3 B4 53 45 06 50
```

这里 53 为 S 命令的 HEX 值， 44 为 YS201X 写设备地址，05 为写的数据字节数，B0…B4 为写出的数据，53 第二个 S 命令，45 为 YS201X 的读设备地址，06 为要读的字节数，50 为 P 命令的 HEX 值。

如下图所示：



3.4 连续写

主机发送命令格式如下：

S CHAR.	SLAVE ADR. + W	NUMBER OF BYTES	DATA 0	...

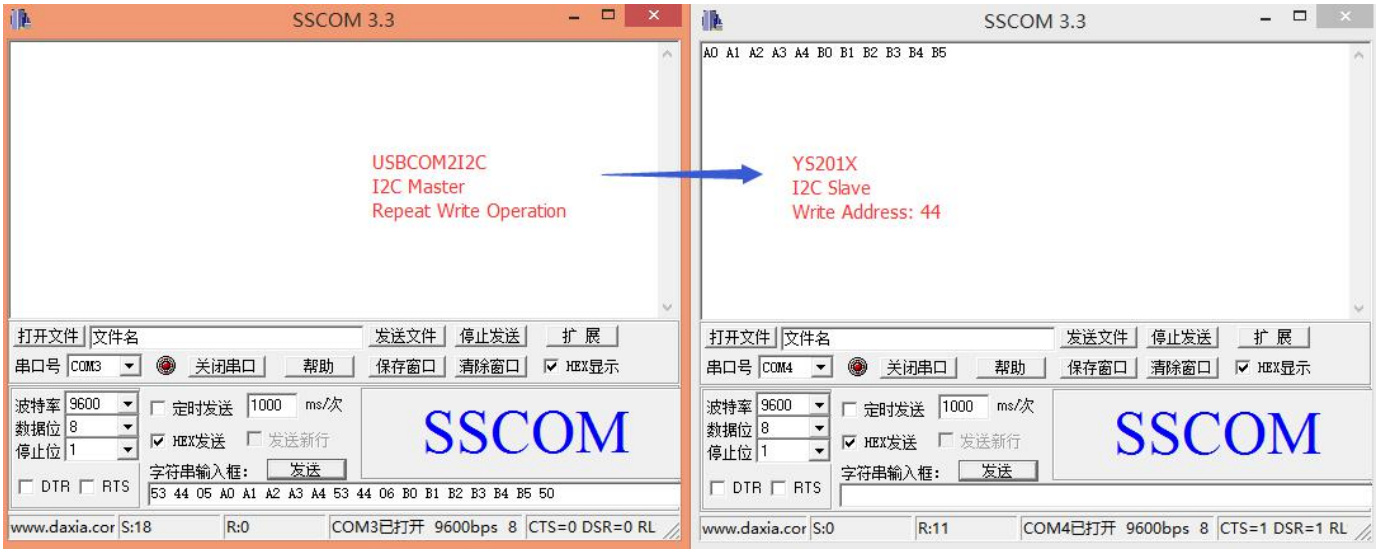
...	DATA N	S CHAR.	SLAVE ADR. + W	NUMBER OF BYTES	DATA 0	DATA N	P CHAR.

主机在两个命令里发2组数据给从机，中间无停止，如下图：
发送命令为：

```
53 44 05 A0 A1 A2 A3 A4 53 44 06 B0 B1 B2 B3 B4 B5 50
```

这里第一个53为第一个S命令，44为YS201X从机写地址，05为写入数据字节数，A0…A4为5个字节数据，紧跟第二个S命令，发送第二批数据, 44为从机写地址，06为写入数据字节数，B0…B5为第二批写入的数据 50为P I2C_STOP命令。

操作结果如下图所示：



3.5 读写含二级地址的I2C Slave

以读写EEPROM为例，写地址为A0 ，读地址为A1。

在EEPROM内部0x11位置，写入0xB1,0xB2,0xB3,0xB4,0xB5,0xB6,0xB7,0xB8共8个字节数据，写入命令如下：

```
53 A0 09 11 B1 B2 B3 B4 B5 B6 B7 B8 50
```

要读取刚才写入的8个字节数据，命令如下：

```
53 A0 01 11 53 A1 08 50
```

EEPROM内部数据按序列编程后的位置图：

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000010	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
00000020	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
00000030	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
00000040	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
00000050	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
00000060	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
00000070	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F
00000080	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
00000090	90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
000000A0	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
000000B0	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
000000C0	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
000000D0	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
000000E0	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
000000F0	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF

读取B0行的16个字节数据的命令如下：

```
53 A0 01 B0 53 A1 10 50
```

每次最多能读49个字节（0x31h）。

四、 内部寄存器

转换器可以通过串口配置内部参数寄存器，主要有以下几种：

Register address	Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	R/W	Default value
General register set											
0x00	BRG0	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W	0xF0
0x01	BRG1	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W	0x02
0x02	PortConf1	GPIO3.1	GPIO3.0	GPIO2.1	GPIO2.0	GPIO1.1	GPIO1.0	GPIO0.1	GPIO0.0	R/W	0x55
0x03	PortConf2	GPIO7.1	GPIO7.0	GPIO6.1	GPIO6.0	GPIO5.1	GPIO5.0	GPIO4.1	GPIO4.0	R/W	0x55
0x04	IOState	GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0	R/W	0x0F
0x05	reserved	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	-	0x00
0x06	I2CAdr	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W	0x26
0x07	I2CClkL	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W	0x13
0x08	I2CClkH	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W	0x13
0x09	I2CTO	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TE	R/W	0x66
0x0A	I2CStat	1	1	1	1	I2CStat[3]	I2CStat[2]	I2CStat[1]	I2CStat[0]	R	0xF0

4.1 波特率发生器【0x00h,0x01h】

通过写入内部寄存器0x00 0x01来配置波特率，计算公式如下：

$$Baud\ rate = \frac{7.3728 \times 10^6}{16 + (BRG1,\ BRG0)}$$

在计算波特率时，先要把BRG寄存器的值从16进制转换为10进制后再带入公式计算。BRG寄存器的值按BRG0，BRG1的顺序写入，写完BRG1寄存器的值后新的波特率立即生效。

Baud rate (bps)	0x01H BRG1 (Hex)	0x00H BRG0 (Hex)
600	0x2F	0xF0
1200	0x17	0xF0
2400	0x0B	0xF0
4800	0x05	0xF0
9600 （Default）	0x02	0xF0
14400	0x01	0xF0
19200	0x01	0x70
38400	0x00	0xB0
56000	0x00	0x73
57600	0x00	0x70
115200	0x00	0x30
128000	0x00	0x29
256000	0x00	0x0C

默认波特率为9600bps。

4.2 I2C总线地址寄存器【0x06h】

转换器 自己的I2C地址，在操作I2C设备时不用，如果与I2C总线上的设备地址冲突，则需要设置。

4.3 I2C时钟速率【0x08h,0x07h】

I2C接口SCL的频率由这两个寄存器来设置，计算公式如下：

$$bit\ frequency = \frac{7.3728 \times 10^6}{2 \times (I2CClkH + I2CClkL)}$$

I2CClkH决定SCL高电平时间，I2CClkL决定SCL低电平时间。

SCL频率设置如下表：

I2CClk(I2CClkH+I2CClkL)	I2C Clock Frequency	I2CClkH=I2CClkL(hex)
10(minimum)	369kHz	0x05
15	246kHz	0x07
25	147kHz	0x0C
30	123kHz	0x0F
38	97kHz	0x13
50	74kHz	0x19
60	61kHz	0x1E
100	37kHz	0x32

注意： 表中第一列数据为十进制，第三列为I2CClkH=I2CClkL的十六进制值，可直接写入寄存器。

4.4 I2C总线超时寄存器【0x09h】

超时寄存器决定允许SCL为低的最大时间，这个时间在I2C总线每个传输状态发生改变后都会重新加载。寄存器的最低位用于超时功能的开启或者关闭：Bit0为1，开启超时功能，Bit0为0，关闭超时功能。

Bit	Symbol	Description
7:1	TO[7:1]	time-out value
0	TE	enable/disable time-out function logic 0 = disable logic 1 = enable

时间设置可以由下面公式计算：

$$time\text{-}out\ period = \frac{I2CTO[7:1] \times 256}{57600}\ seconds$$

4.5 I2C总线状态寄存器【0x0A】

转换器传输或接收状态报告寄存器，可以反馈I2C读写数据发送或者接收是否成功。

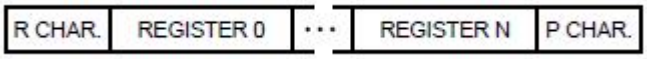
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	I^2C-bus status description
1	1	1	1	0	0	0	0	I2C_OK
1	1	1	1	0	0	0	1	I2C_NACK_ON_ADDRESS
1	1	1	1	0	0	1	0	I2C_NACK_ON_DATA
1	1	1	1	1	0	0	0	I2C_TIME_OUT

五、 内部寄存器的读写

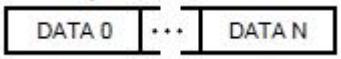
5.1 读内部寄存器

读内部寄存器命令R，后面跟寄存器地址，以P结尾

主机发送命令格式：



主机收到数据格式：

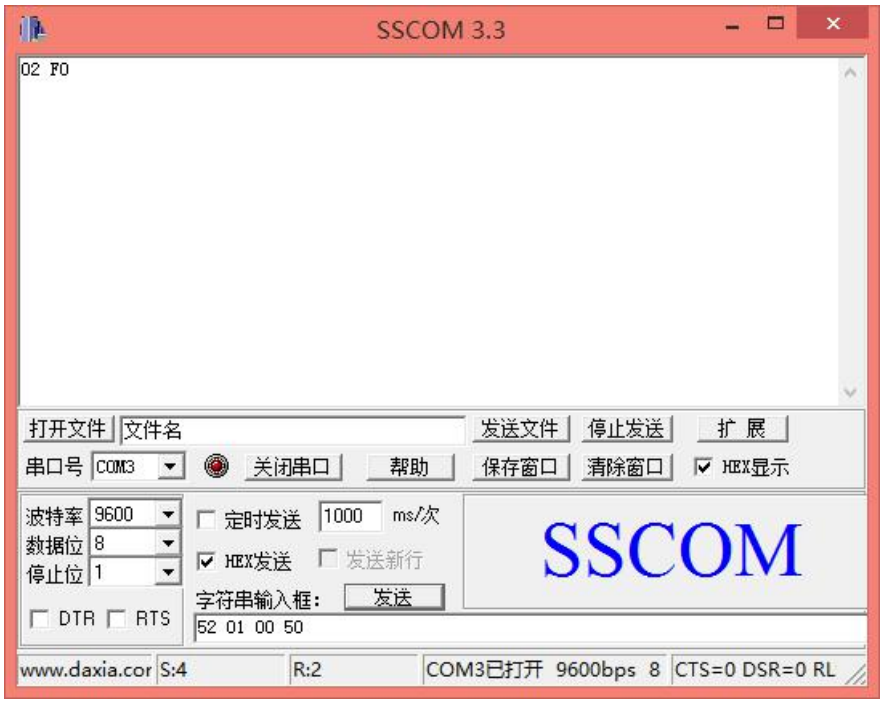


例如读取波特率寄存器BRG的值，在串口助手界面，以HEX模式输入：

```
52 01 00 50
```

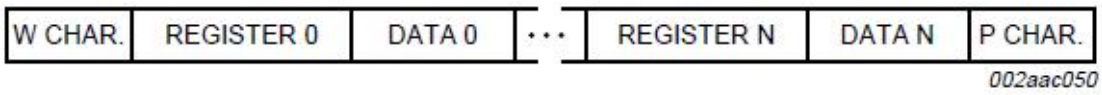
这里52为R命令的HEX值 01 00 为BRG 寄存器地址，50为P命令的HEX值。

返回结果02 F0，如下图：



5.2 写内部寄存器

写内部寄存器的命令为R，HEX值为0x57，写命令格式如下：



以配置波特率为例：

默认波特率是9600bps，如果要改为115200bps，BRG寄存器的值BRG0= 0x30 BRG1=0x00,在串口助手端输入如下命令：

```
57 00 30 01 00 50
```

这里57为W写命令的HEX码，00 01 为BRG寄存器地址。

左图是在9600bps下发送修改BRG寄存器命令设置波特率为115200bps；右图在115200bps下读取BRG寄存器0x00 0x01的值，与前面写入的值0x30 0x00一致，波特率设置成功。

5.3 休眠模式（Power Down Mode）

转换器转换器可以通过命令Z让转换器进入省电休眠状态，该状态下对主机发送的任何读写操作指令都不做响应。

进入休眠模式的命令格式如下：

Z CHAR.	0x5A	0xA5	P CHAR.

002aac056

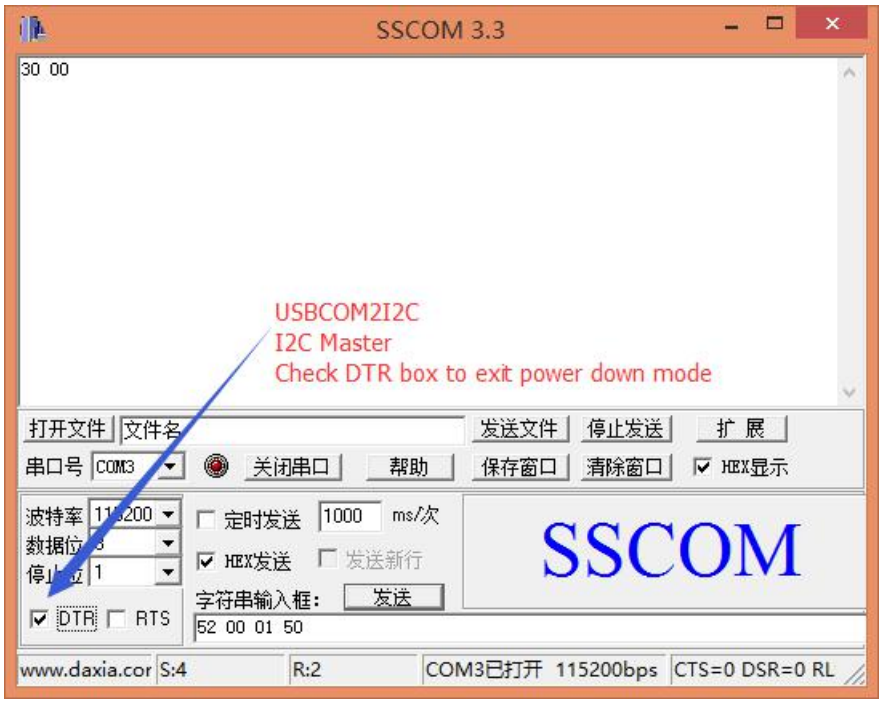
主机发送指令

```
5A 5A A5 50
```

I2C模组进入休眠状态，不再对主机发送的任何指令做反应。

5.4 退出休眠模式 （此功能标准版不支持）

转换器转换器在I2C模组休眠状态下，可以通过串口助手的DTR信号唤醒，在串口助手界面，选中DTR然后再取消即可唤醒I2C模组，退出休眠模式。



5.4 复位I2C模组 （此功能标准版不支持）

A. 主机可以通过串口的RTS来复位转换器中的I2C模组，复位后波特率为默认9600bsp，串口接收到OK字符代表I2C模组复位成功。

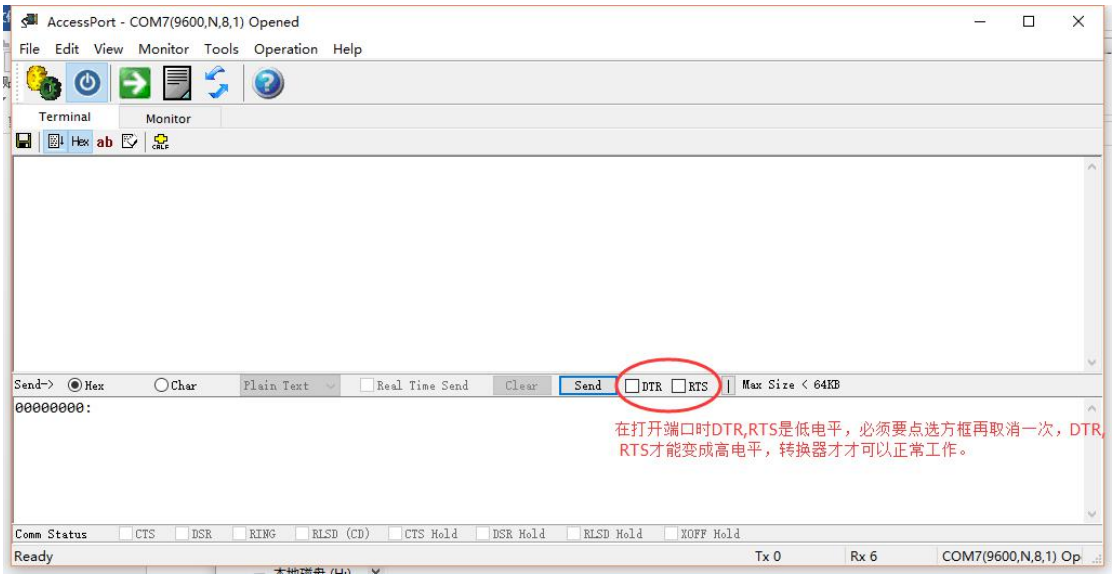
B. 在非当前波特率发送命令，转换器不能正确接收，而会自动复位，返回到默认波特率9600bps。
例如：
在9600bps状态下，写BRG寄存器，设波特率为115200bps后，界面还停留在9600bps波特率，然后读0x01 0x02寄存器值，不能读出，而是收到字符OK，再读就能读到寄存器值了，这时内部BRG寄存器又恢复为默认9600bps时的值了。

重要说明：

由于各版本串口助手打开时DTR RTS初始状态不一样，对于增强版，在打开串口助手后，先勾选再取消DTR RTS，使其恢复高电平状态，否则会影响转换器正常工作。

例如说明书中SSCOM3.3版本，在打开串口时，RTS由低变高，相当于转换器复位一次，会收到OK字符；DTR一直为低，需要点选再取消才会变高。（如下图）

在AccessPort软件中，打开端口DTR RTS默认为低电平，必须要点选再取消前面的方框一次后，DTR RTS才变成高电平，转换器才能正常工作。【勾选方框，DTR RTS输出低电平，取消勾选，DTR RTS输出高电平】


AccessPort - COM7(9600,N,8,1) Opened
File Edit View Monitor Tools Operation Help
Terminal
Monitor
Send-> Hex Char Plain Text Real Time Send Clear Send DTR RTS Max Size < 64KB
00000000:
在打开端口时DTR,RTS是低电平，必须要点选方框再取消一次，DTR,
RTS才能变成高电平，转换器才才可以正常工作。
Comm Status CTS DSR RING RLSD (CD) CTS Hold DSR Hold RLSD Hold XOFF Hold
Ready
Tx 0
Rx 6
COM7(9600,N,8,1) Op

关于GPIO的操作实例

1. 写 GPIO

主机发送命令格式：

O CHAR.	DATA	P CHAR.

详见芯片手册。（见附录）

2. 读 GPIO

主机发送命令格式：

I CHAR.	P CHAR.

主机收到数据：

DATA

详见芯片手册。（见附录）

例子：要配置 IO6 为高电平，其它 IO 为低电平

上电默认 IO 配置寄存器 0x02 0x03 值是 0x55，为输入状态； 需要先修改 02 03 寄存器值为 0xAA

配置 IO 为**推挽**状态：

```
57 02 AA 03 AA 50        //配置 IO 为推挽状态
4F 40 50                 //配置 IO6 为高，其他为低
52 02 03 04 50           //查看 02 03 04 寄存器值，为 AA AA 40
```

配置 IO 输出电平表

高电平 IO	0x04 寄存器	十六进制	说明
IO7	*1000 0000*	*0x80*	*IO7 为高电平，其它 IO 为低电平*
IO6	*0100 0000*	*0x40*	*IO6 为高电平，其它 IO 为低电平*
IO5	*0010 0000*	*0x20*	*IO5 为高电平，其它 IO 为低电平*
IO4	*0001 0000*	*0x10*	*IO4 为高电平，其它 IO 为低电平*
IO3	*0000 1000*	*0x08*	*IO3 为高电平，其它 IO 为低电平*
IO2	*0000 0100*	*0x04*	*IO2 为高电平，其它 IO 为低电平*
IO1	*0000 0010*	*0x02*	*IO1 为高电平，其它 IO 为低电平*
IO0	*0000 0001*	*0x01*	*IO0 为高电平，其它 IO 为低电平*
同时配置全部 IO 为高电平	*1111 1111*	*0xFF*	
同时配置全部 IO 为低电平	*0000 0000*	*0x00*	

附1 YS201X

YS201X I2C Slave to USB Adapter
https://item.taobao.com/item.htm?id=41124499102

附2 SSCOM3.3

串口助手SSCOM3.3下载
https://yunpan.cn/Ocq4g9WY8eXFsn 访问密码 6301