**Jetson nano 的I2C通信**

**一、命令行窗口：**

1、I2Ctool安装：

sudo apt-get update

sudo apt-get install -y i2c-tools

检查安装情况：apt-cache policy i2c-tools

输出如下即为安装成功：

i2c-tools:

已安装：4.0-2

候选： 4.0-2

2、列出安装的所有总线命令：sudo i2cdetect -l

效果：

i2c-3 i2c 7000c700.i2c I2C adapter

i2c-1 i2c 7000c400.i2c I2C adapter

i2c-6 i2c Tegra I2C adapter I2C adapter

i2c-4 i2c 7000d000.i2c I2C adapter

i2c-2 i2c 7000c500.i2c I2C adapter

i2c-0 i2c 7000c000.i2c I2C adapter

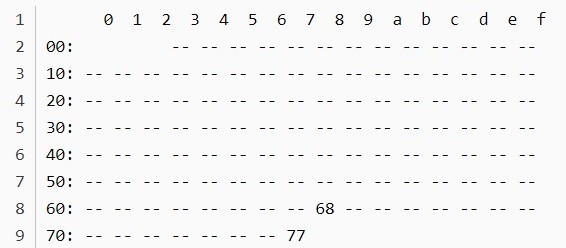
i2c-5 i2c 7000d100.i2c I2C adapter

扫描某一总线bus上所有i2c设备，并且打印出设备i2c总线地址

命令（其中1为i2c设备号）：sudo i2cdetect -y -r -a 1

参数: -y是无视交互问题直接执行，-r是SMBus read byte命令，-a是所有地址，1是指i2c-1。

效果：



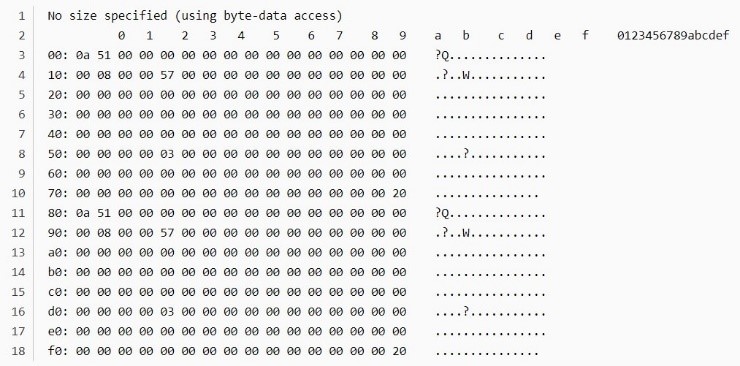
其中，0x68和0x77 为i2c设备在I2C\_1上的总线地址

3、寄存器数据导出。总线上i2c设备的数据是要存储在寄存器上。

命令：sudo i2cdump -y 1 0x68

含义为：导出I2C-1中地址为0x68的设备的数据

效果：左边是寄存器地址，右侧数据

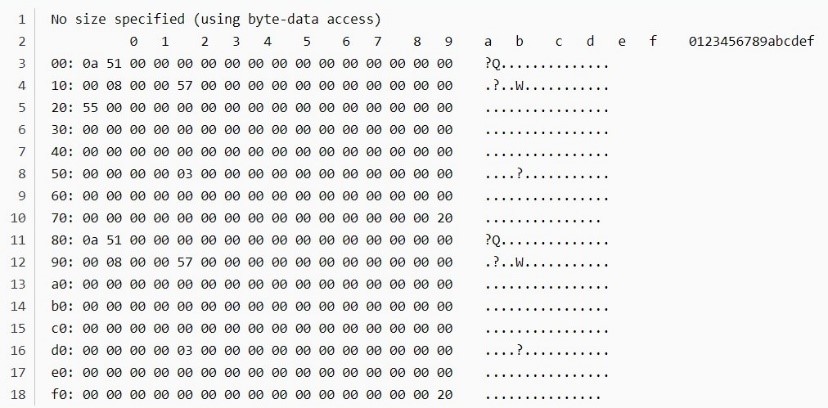


4、寄存器数据写入，即向寄存器中某一位地址上插入数据：

命令：(sudo i2cset -y 1 i2c设备地址 写入位置 写入内容)

sudo i2cset -y 1 0x68 0x20 0x55

效果：(寄存器地址0x20上被插入数据0x55)



5、读出总线地址为0x77设备的0x90寄存器地址的数据

命令（sudo i2cget -y 1 i2c设备地址 寄存器位置）

sudo i2cget -y 1 0x68 0x80

**二、pythen代码编写：**

1、安装 python2 和 python3 的 I2C 库,分别输入“sudo apt-get install python-smbus”指令和“sudo apt-get install python3-smbus”指令即可

2、使用指令“apt-cache policy i2c-tools”来检查安装情况

3、导入模块,实例化

import SMBus

i2c = SMBus(1)   # 1 代表一号总线

4、open() 打开一个i2c bus

i2c.open(1)  # 1 代表 /dev/i2c-1

5、close() 关闭一个i2c bus

i2c.close()

6、read\_byte //从设备读取一个字节

read\_byte(i2c\_addr, force=None)

参数:i2c\_addr (整形) – i2c 地址

返回值:读取的字节

7、write\_byte //向设备写入一个字节

write\_byte(i2c\_addr, value, force=None)

参数:

i2c\_addr (整形) – i2c 地址

value(整形) -要写入的值

8、read\_byte\_data //从指定的寄存器中读取一个字节

read\_byte\_data(i2c\_addr, register, force=None)

参数:

i2c\_addr (整形) – i2c 地址

register(整形)–读取的寄存器

返回值:读取的字节

9、write\_byte\_data //向指定的寄存器中写入一个字节

write\_byte\_data(i2c\_addr, register, value, force=None)

参数:

i2c\_addr (整形) – i2c 地址

register(整形)–读取的寄存器

value(整形) -要写入的值

10、read\_i2c\_block\_data //从给定的寄存器中读取字节数据块

read\_i2c\_block\_data(i2c\_addr, register, length, force=None)

参数:

i2c\_addr (整形) – i2c 地址

register(整形)–开始寄存器

length(整形)–所需的块长度

11、write\_i2c\_block\_data //将字节数据块写入给定寄存器

write\_i2c\_block\_data(i2c\_addr, register, data, force=None)

参数:

i2c\_addr (整形) – i2c 地址

register(整形)–开始寄存器

data(整形)–字节组成的列表

12、read\_word\_data //从给定的寄存器中读取一个字(2个字节)

read\_byte\_data(i2c\_addr, register, force=None)

参数:

i2c\_addr (整形) – i2c 地址

register(整形)–开始寄存器

返回值:整形

13、 write\_word\_data //将一个字(2个字节)写入给定的寄存器

write\_word\_data(i2c\_addr, register, value, force=None)

参数:

i2c\_addr (整形) – i2c 地址

register(整形)–开始寄存器

value(整形) -要写入的字

[例子](https://blog.csdn.net/finedayforu/article/details/116590855?utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-2.no_search_link&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-2.no_search_link)