



慕华音频芯片寄存器读写说明(CJC8988)

一、寄存器 I2C 读写对比

不同家的音频芯片寄存器地址存在差异, 可查看下列表格, 这里通过 I2C 通信读写寄存器举例说明:

品牌	寄存器地址	寄存器数据值	备注
慕华音频芯片	7 位	9 位	I2C 通信中以字节为单位发送
其他音频芯片	8 位	8 位	I2C 通信中以字节为单位发送

1.1 芯片中 8 位寄存器地址+8 位寄存器数据写寄存器时序:

Start+设备地址+Write (0) +ack+寄存器地址 (8bit) +ack+寄存器数据 (8bit) +ack+stop

驱动中对应 i2cwrite(寄存器地址 (8bit) , 寄存器数据 (8bit))

写 8 位寄存器+8 位寄存器数据, 应用程序中:

R0 = 0x34 i2cwrite(0x0, 0x34) i2cwrite(寄存器地址 (8bit) , 寄存器数据 (8bit))

R1 = 0x46 i2cwrite(0x1, 0x46) i2cwrite(寄存器地址 (8bit) , 寄存器数据 (8bit))

R9 = 0x57 i2cwrite(0x9, 0x57) i2cwrite(寄存器地址 (8bit) , 寄存器数据 (8bit))

写 8 位寄存器地址+8 位寄存器数据读寄存器时序:

Start+0x1A (001 1010) +Write (0) +ack+寄存器地址(8bit)+ack+

Start+0x1A (001 1010) +Read (1) +ack+寄存器数据(8bit)+nack+stop

1.2 慕华音频芯片中 7 位寄存器+9 位数据地址写寄存器时序:

Start+设备地址+Write (0) +ack+[寄存器地址 (7bit) +寄存器数据最高位 (1bit)]+ack+寄存器数据低 (8bit) +ack+stop

驱动中对应 i2cwrite(寄存器地址 (7bit) +寄存器数据最高位 (1bit) , 寄存器数据低 (8bit))



7 位寄存器+9 位数据地址，应用程序中：

R0 = 0x134 i2cwrite(0x1, 0x34) i2cwrite(寄存器地址 (7bit) + 寄存器数据最高位 (1bit) , 寄存器数据低 (8bit))

R1 = 0x046 i2cwrite(0x2, 0x46) i2cwrite(寄存器地址 (7bit) + 寄存器数据最高位 (1bit) , 寄存器数据低 (8bit))

R9 = 0x157 i2cwrite(0x13, 0x57) i2cwrite(寄存器地址 (7bit) + 寄存器数据最高位 (1bit) , 寄存器数据低 (8bit))

7 位寄存器+9 位数据地址读寄存器时序：

Start+0x1A (001 1010) +Write (0) +ack+寄存器地址(8bit)+ack+

Start+0x1A (001 1010) +Read (1) +ack+寄存器数据(8bit)+nack+stop

读寄存器从时序上看没什么区别，因 i2c 通信基本上以字节为单位传输，读取 8 位寄存器地址 8 位寄存器数据时，只需要读取一次。而慕华音频芯片寄存器数据为 9 位，占用两个字节地址，所以需要读两次应用程序读比较。

写读 8 位寄存器+8 位寄存器数据 ，应用程序中：

R0 = 0x34 i2cread(0x0, ®value) 读出后 regvalue = 0x34

R1 = 0x46 i2cread(0x1, ®value) 读出后 regvalue = 0x46

R9 = 0x57 i2cread(0x9, ®value) 读出后 regvalue = 0x57

写 7 位寄存器+9 位数据地址 ，应用程序中：

R0 = 0x134 i2cread(0x0, ®value) 读出后 regvalue = 0x34

i2cread(0x1, ®value) 读出后 regvalue = 0x1 读取寄存器需要读取两次

R1 = 0x046 i2cread(0x2, ®value) 读出后 regvalue = 0x46

i2cread(0x3, ®value) 读出后 regvalue = 0x0 读取寄存器需要读取两次

R9 = 0x157 i2cread(0x12, ®value) 读出后 regvalue = 0x57

i2cread(0x13, ®value) 读出后 regvalue = 0x1 读取寄存器需要读取两次



二、Linux 系统下调试时如何读取修改慕华音频芯片寄存器

以 CJC8988 为例说明

很多时候调试时需要动态修改寄存器，那么 Linux 系统下 i2ctool 中的 i2cdump 命令把所有寄存器打印出来，如下图(R0 = 0x130 R1= 0x130 R2= 0x179 R3= 0x179 R7= 0x020 R8 = 0x0)

我们可以把 i2cdump -f -y 3 0x1a 读出所有寄存器值，其中：3 是总线号，0x1a 是设备地址。

```

45] 3204 root@RK3588-Tronlong:~#
46] 3205 root@RK3588-Tronlong:~# i2cdump -f -y 3 0x1a
46] 3206 No size specified (using byte-data access)
46] 3207      0 1 2 3 4 5 6 7 8 9 a b c d e f      0123456789abcdef
46] 3208 00: 30 01 30 01 79 01 79 01 00 00 00 00 00 02 00      070?y?y?.....?
46] 3209 10: 00 00 00 00 ff 01 ff 01 0f 00 0f 00 00 00 00      .....?.?.?.?.....
46] 3210 20: 00 00 7b 00 00 00 32 00 00 00 c3 01 c3 01 c0 00      ..{...2...?????.
46] 3211 30: 04 00 7d 01 00 00 00 00 00 00 00 00 00 00 00      ?.)?.....
46] 3212 40: 00 00 00 00 52 01 50 00 52 00 50 01 50 00 50 00      ....R?P.R.P?P.P.
46] 3213 50: 79 01 79 01 79 00 00 00 00 00 00 00 00 00 00      y?y?y.....
46] 3214 60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3215 70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3216 80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3217 90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3218 a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3219 b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3220 c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3221 d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3222 e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
46] 3223 f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      .....
35:46] 3224 root@RK3588-Tronlong:~#

```

如需读取单个寄存器值我们可以使用 i2cget，其中：3 是总线号，0x1a 是设备地址。

具体命令如下：

i2cget -y -f 3 0x1a 0x14 是获取 R10 寄存器 低 8 位的值

i2cget -y -f 3 0x1a 0x15 是获取 R10 寄存器 最高位的值

i2cget -y -f 3 0x1a 0x0 是获取 R0 寄存器 低 8 位的值

i2cget -y -f 3 0x1a 0x1 是获取 R0 寄存器 最高位的值



如需修改寄存器值我们可以使用 i2cset, 具体命令如下:

i2cset -y -f 3 0x1a 0x01 0x17 是设置 R0 = 0x117

i2cset -y -f 3 0x1a 0x00 0x17 是设置 R0 = 0x017

i2cset -y -f 3 0x1a 0x03 0x17 是设置 R1 = 0x117

i2cset -y -f 3 0x1a 0x02 0x17 是设置 R1 = 0x017

i2cset -y -f 3 0x1a 0x0e 0x42 是设置 R1 = 0x042

i2cset -y -f 3 0x1a 0x0f 0x42 是设置 R1 = 0x142