SigmaStar Camera
I2C 使用参考

© 2019 SigmaStar Technology Corp. All rights reserved.

SigmaStar Technology makes no representations or warranties including, for example but not limited to, warranties of merchantability, fitness for a particular purpose, non-infringement of any intellectual property right or the accuracy or completeness of this document, and reserves the right to make changes without further notice to any products herein to improve reliability, function or design. No responsibility is assumed by SigmaStar Technology arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others.

SigmaStar is a trademark of SigmaStar Technology Corp. Other trademarks or names herein are only for identification purposes only and owned by their respective owners.

REVISION HISTORY

Revision No.	Description	Date
{000001}	• {Initial release}	{07/28/2018}
{000002}	• {i6e update}	{01/11/2019}

Copyright © 2019 SigmaStar Technology Corp. All rights reserved.

TABLE OF CONTENTS

Copyright © 2019 SigmaStar Technology Corp. All rights reserved.

1. 概述

1.1. 概述

Table 1: 表 1-1

I2C Group	SCL	SDA	DEV
HW I2C group0	PAD_I2C0_SCL	PAD_I2C0_SDA	/dev/i2c-0
HW I2C group1	PAD_I2C1_SCL	PAD_I2C1_SDA	/dev/i2c-1,
HW I2C group2	PAD_I2C2_SCL	PAD_I2C2_SDA	/dev/i2c-2,

提供 3 组 HW I2C，第一组 HW I2C 对应 pad 是 PAD_I2C0_SCL/ PAD_I2C0_SDA,对应节点是/dev/i2c-0；第二组 HW I2C 对应 pad 是 PAD_I2C1_SCL/ PAD_I2C1_SDA,对应节点是/dev/i2c-1。第三组 HW I2C 对应 pad 是 PAD_I2C2_SCL/ PAD_I2C2_SDA,对应节点是/dev/i2c-1。

这是所有 i2c 全开的情况，如果实际情况中没有打开这么多 i2c，要根据 mhal_iic.h 中"#define HAL_HWI2C_PORTS 3"的定义和 infinity6e.dtsi 中 i2c device node 的定义情况，而对应的节点一般看 i2c-group = <0>;定义的值是多少。

```
i2c0@0{
    compatible = "mstar,i2c";
    status = "ok";
    reg = <0x1F226800 0x200>,<0x1F204c00 0x200>,<0x1F206600 0x200>;
    //clocks = <&CLK_miic0>;
    i2c-group = <0>;
    i2c-dma = <0>;
    /*
     * padmux: 1 -> PAD_I2C0_SCL, PAD_I2C0_SDA
    */
    i2c-padmux = <1>;
};
```

Figure 2: 图 1-1

下面所有的描述都是以第一组 HW I2C 为例，其他 I2C 请参照第一组的使用方法。

Copyright © 2019 SigmaStar Technology Corp. All rights reserved.

2. 使用 I2C

2.1. 读写 I2C

通过标准的 ms_i2c_xfer 函数来读写 I2C，以下是一个使用的小例子。

```
#include <stdio.h>
#include <linux/types.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ioctl.h>
#include <errno.h>
#include <assert.h>
#include <string.h>
#include <linux/i2c.h>
#include <linux/i2c-dev.h>

#define FILE_NAME "/dev/i2c-0"

static int i2c_write(int fd,unsigned char slave_addr, unsigned char reg_addr, unsigned char value)
{
    unsigned char outbuf[2];
    struct i2c_rdwr_ioctl_data packets;
    struct i2c_msg messages[1];

    messages[0].addr   = slave_addr;
    messages[0].flags = 0;
    messages[0].len    = sizeof(outbuf);
    messages[0].buf    = outbuf;

    /* The first byte indicates which register we ‘ll write */
    outbuf[0] = reg_addr;

    /*
     * The second byte indicates the value to write.   Note that for many
     * devices, we can write multiple, sequential registers at once by
     * simply making outbuf bigger.
     */
```


Copyright © 2019 SigmaStar Technology Corp. All rights reserved.

```
    outbuf[1] = value;

    /* Transfer the i2c packets to the kernel and verify it worked */
    packets.msgs  = messages;
    packets.nmsgs = 1;
    if(ioctl(fd, I2C_RDWR, &packets) < 0)
    {
        perror("Unable to send data");
        return 1;
    }

    return 0;
}

static int i2c_read(int fd, unsigned char slave_addr, unsigned char reg_addr, unsigned char *value)
{
    unsigned char inbuf, outbuf;
    struct i2c_rdwr_ioctl_data packets;
    struct i2c_msg messages[2];

    /*
     * In order to read a register, we first do a "dummy write" by writing
     * 0 bytes to the register we want to read from.  This is similar to
     * the packet in set_i2c_register, except it 's 1 byte rather than 2.
     */
    outbuf = reg_addr;
    messages[0].addr  = slave_addr;
    messages[0].flags = 0;
    messages[0].len   = sizeof(outbuf);
    messages[0].buf   = &outbuf;

    /* The data will get returned in this structure */
    messages[1].addr  = slave_addr;
    messages[1].flags = I2C_M_RD/* | I2C_M_NOSTART*/;
    messages[1].len   = sizeof(inbuf);
    messages[1].buf   = &inbuf;

    /* Send the request to the kernel and get the result back */
    packets.msgs      = messages;
    packets.nmsgs     = 2;
    if(ioctl(fd, I2C_RDWR, &packets) < 0)
    {
        perror("Unable to send data");
        return 1;
```


Copyright © 2019 SigmaStar Technology Corp. All rights reserved.

```
    }
    *value = inbuf;

    return 0;
}

int main(int argc, char **argv)
{
    int fd;
    unsigned int slave_addr=0, reg_addr=0, value = 0;

    if (argc < 4){
        printf("Usage:\n%s r[w] start_addr reg_addr [value]\n",argv[0]);
        return 0;
    }

    fd = open(FILE_NAME, O_RDWR);
    if (!fd)
    {
        printf("can not open file %s\n", FILE_NAME);
        return 0;
    }

    sscanf(argv[2], "%x", &slave_addr);
    sscanf(argv[3], "%x", &reg_addr);

    if(!strcmp(argv[1],"r"))
    {
        i2c_read(fd, slave_addr, reg_addr, (unsigned char*)&value);
    }
    else if(argc>4&&!strcmp(argv[1],"w"))
    {
        sscanf(argv[4], "%x", &value);
        i2c_write(fd, slave_addr, reg_addr, value);
    }

    close(fd);
    return 0;
}
```


Copyright © 2019 SigmaStar Technology Corp. All rights reserved.