



MacroSilicon Release For
深圳市视品数字技术有限公司
Internal Use Only
2023年03月13日

MS7210

软件开发用户手册

MacroSilicon Release For
深圳市视品数字技术有限公司
Internal Use Only
2023年03月13日

目录

MS7210 SDK.....	1
移植指南	1
APPLICATION NOTE	1
1. 概述	4
2. 总体设计	4
2.1 设计理念	4
2.2 API 列表	4
2.2.1 软件运行相关 API	4
2.2.2 操作视频类 API	5
3. 结构体类型说明	5
3.1 MS_TIMING_DEFINE	5
3.2 MS_HDMI_DEFINE	5
3.3 MS_HDMITX_DEFINE	9
3.4 MS_DVIN_DEFINE	10
4 API 使用说明	11
4.1 CHAR* MS7210_SDK_VERSION(VOID);	11
4.2 BOOL MS7210_CHIP_CONNECT_DETECT(UINT8 U8_CHIP_ADDR);	11
4.3 VOID MS7210_DVIN_INIT(DVIN_CONFIG_T *T_DVIN_CONFIG, UINT8 U8_SPDIF_IN);	12
4.4 VOID MS7210_DVIN_DATA_SWAP(UINT8 U8_SWAP_MODE);	12
4.5 VOID MS7210_DVIN_PHASE_ADJUST(BOOL B_INVERT, UINT8 U8_DELAY);	12
4.6 VOID MS7210_DVIN_TIMING_CONFIG(DVIN_CONFIG_T *T_DVIN_CONFIG, VIDEOTIMING_T *PTTIMING, HDMI_CONFIG_T *PT_HDMI_TX);	12
4.7 VOID MS7210_DVIN_VIDEO_CONFIG(BOOL B_CONFIG);	13
4.8 BOOL MS7210_HDMITX_HPD_DETECT(VOID);	13

4.9	BOOL MS7210_HDMITX_EDID_GET(UINT8 *U8_EDID);	13
4.10	BOOL MS7210_HDMITX_INPUT_TIMING_STABLE_GET(VOID);	13
4.11	VOID MS7210_HDMITX_OUTPUT_CONFIG(HDMI_CONFIG_T*PT_HDMI_TX);	13
4.12	VOID MS7210_HDMITX_SHUTDOWN_OUTPUT(VOID);	14
5	编译选项	14
5.1	平台相关编译选项	14
6	SDK 所使用数据类型	14
7	应用指南	15
7.1	上电复位操作	15
7.2	外部 API 实现	15
7.2.1	延时函数	15
7.2.2	I2C 读写操作	15
7.3	调用举例	17
7.3.1	芯片初始化	17
7.3.2	配置芯片输入相关模块初始化	17
7.3.3	检测输出状态变化并配置	17
附录：	版本历史	18

1. 概述

MS7210 芯片支持 24bit 数字视频和 I2S/SPDIF 音频信号输入, HDMI 信号输出, 芯片引脚示意图参考 MS7210 datasheet。

MS7210 SDK LIB 是一套针对 MS7210 视频处理芯片的软件开发包。采用 ANSI C 语言编写, 软件结构采用模块化设计, 可移植性好方便进行二次开发。SDK 正式 Release 中包含以下文件:

- ms7210.h
- ms7210_comm.h
- ms7210_typedef.h
- SDK 库文件

库文件命名及适用编译环境说明:

ms7200_sdklib_keil_c51_vx.x.x_202xxxxx.lib – 适用于 Keil C51;

ms7210_sdklib_keil_m0/m3/m4_vx.x.x_202xxxxx.lib – 适用于 Keil MDK, STM32 系列 MCU;

ms7210_sdklib_iar_stm8s_vx.x.x_202xxxxx.a – 适用于 IAR STM8;

ms7210_sdklib_stvd_stm8s_vx.x.x_202xxxxx.lib – 适用于 STVD STM8。

- 版本说明文件

2. 总体设计

2.1 设计理念

SDK 的接口上分为与软件运行相关和硬件操作读取相关两大类, 其中与软件相关的接口有:

- 获取 SDK 版本信息;
- 设置芯片 I2C Slave 地址, 获取芯片连接状态。

与 MS7210 硬件操作及读取相关的接口有:

- 模块初始化配置;
- 解密功能 HDCP KEY 设置;
- 配置输入通道切换;
- 配置输入相位调整;
- 配置输入时序;
- 获取输出端 HPD、EDID 状态;
- 获取输出端 Stable 状态;
- 配置输出使能、关闭。

2.2 API 列表

2.2.1 软件运行相关 API

- CHAR* ms7210_sdk_version(VOID);

- BOOL ms7210_chip_connect_detect(UINT8 u8_chip_addr);

2.2.2 操作视频类 API

- VOID ms7210_dvin_init(DVIN_CONFIG_T *t_dvin_config, UINT8 u8_spdif_in);
- VOID ms7210_dvin_data_swap(UINT8 u8_swap_mode);
- VOID ms7210_dvin_phase_adjust(BOOL b_invert, UINT8 u8_delay);
- VOID ms7210_dvin_timing_config(DVIN_CONFIG_T *t_dvin_config, VIDEOTIMING_T *ptTiming, HDMI_CONFIG_T *pt_hdmi_tx);
- VOID ms7210_dvin_video_config(BOOL b_config);
- BOOL ms7210_hdmitx_hpd_detect(VOID);
- BOOL ms7210_hdmitx_edid_get(UINT8 *u8_edid);
- BOOL ms7210_hdmitx_input_timing_stable_get(VOID);
- VOID ms7210_hdmitx_output_config(HDMI_CONFIG_T *pt_hdmi_tx);
- VOID ms7210_hdmitx_shutdown_output(VOID);

3. 结构体类型说明

3.1 MS_TIMING_DEFINE

typedef struct _T_MS7200_VIDEO_TIMING

{
 UINT8 u8_polarity;
 UINT16 u16_htotal;
 UINT16 u16_vtotal;
 UINT16 u16_hactive;
 UINT16 u16_vactive;
 UINT16 u16_pixclk; /*10000hz*/
 UINT16 u16_vfreq; /*0.01hz*/
 UINT16 u16_hoffset; /* h sync start to h active*/
 UINT16 u16_voffset; /* v sync start to v active*/
 UINT16 u16_hsyncwidth;
 UINT16 u16_vsyncwidth;
} VIDEOTIMING_T;

3.2 MS_HDMI_DEFINE

typedef enum _E_HDMI_VIDEO_CLK_REPEAT_

{
 HDMI_X1CLK = 0x00,

```
HDMI_X2CLK      = 0x01,  
HDMI_X3CLK      = 0x02,  
HDMI_X4CLK      = 0x03,  
HDMI_X5CLK      = 0x04,  
HDMI_X6CLK      = 0x05,  
HDMI_X7CLK      = 0x06,  
HDMI_X8CLK      = 0x07,  
HDMI_X9CLK      = 0x08,  
HDMI_X10CLK     = 0x09  
}HDMI_CLK_RPT_E;
```

```
typedef enum _E_HDMI_VIDEO_ASPECT_RATIO_  
{  
    HDMI_4X3      = 0x01,  
    HDMI_16X9     = 0x02  
}HDMI_ASPECT_RATIO_E;
```

```
typedef enum _E_HDMI_VIDEO_SCAN_INFO_  
{  
    HDMI_OVERSCAN    = 0x01,    //television type  
    HDMI_UNDERSCAN   = 0x02    //computer type  
}HDMI_SCAN_INFO_E;
```

```
typedef enum _E_HDMI_COLOR_SPACE_  
{  
    HDMI_RGB         = 0x00,  
    HDMI_YCBCR422    = 0x01,  
    HDMI_YCBCR444    = 0x02,  
    HDMI_YUV420      = 0x03  
}HDMI_CS_E;
```

```
typedef enum _E_HDMI_COLOR_DEPTH_  
{  
    HDMI_COLOR_DEPTH_8BIT    = 0x00,  
    HDMI_COLOR_DEPTH_10BIT   = 0x01,  
    HDMI_COLOR_DEPTH_12BIT   = 0x02,  
    HDMI_COLOR_DEPTH_16BIT   = 0x03  
}HDMI_COLOR_DEPTH_E;
```

```
typedef enum _E_HDMI_COLORIMETRY_  
{  
    HDMI_COLORIMETRY_601     = 0x00,  
    HDMI_COLORIMETRY_709     = 0x01,
```

```
HDMI_COLORIMETRY_656    = 0x02,
HDMI_COLORIMETRY_1120   = 0x03,
HDMI_COLORIMETRY_SMPTE  = 0x04,
HDMI_COLORIMETRY_XVYCC601 = 0x05,
HDMI_COLORIMETRY_XVYCC709 = 0x06
}HDMI_COLORIMETRY_E;

//HDMI vendor specific
typedef enum _E_HDMI_VIDEO_FORMAT_
{
    HDMI_NO_ADD_FORMAT,
    HDMI_4Kx2K_FORMAT,
    HDMI_3D_FORMAT
}HDMI_VIDEO_FORMAT_E;

typedef enum _E_HDMI_4Kx2K_VIC_
{
    HDMI_4Kx2K_30HZ = 0x01,
    HDMI_4Kx2K_25HZ,
    HDMI_4Kx2K_24HZ,
    HDMI_4Kx2K_24HZ_SMPTE
}HDMI_4Kx2K_VIC_E;

typedef enum _E_HDMI_3D_STRUCTURE_
{
    HDMI_FRAME_PACKING,
    HDMI_FIELD_ALTERNATIVE,
    HDMI_LINE_ALTERNATIVE,
    HDMI_SIDE_BY_SIDE_FULL,
    L_DEPTH,
    L_DEPTH_GRAPHICS,
    SIDE_BY_SIDE_HALF = 8
}HDMI_3D_STRUCTURE_E;

//HDMI audio
typedef enum _E_HDMI_AUDIO_MODE_
{
    HDMI_AUD_MODE_AUDIO_SAMPLE    = 0x00,
    HDMI_AUD_MODE_HBR              = 0x01,
    HDMI_AUD_MODE_DSD              = 0x02,
    HDMI_AUD_MODE_DST              = 0x03
}HDMI_AUDIO_MODE_E;
```

```
typedef enum _E_HDMI_AUDIO_I2S_RATE_
```



```
{
    HDMI_AUD_RATE_44K1  = 0x00,
    HDMI_AUD_RATE_48K   = 0x02,
    HDMI_AUD_RATE_32K   = 0x03,
    HDMI_AUD_RATE_88K2  = 0x08,
    HDMI_AUD_RATE_96K   = 0x0A,
    HDMI_AUD_RATE_176K4 = 0x0C,
    HDMI_AUD_RATE_192K  = 0x0E
}HDMI_AUDIO_RATE_E;

typedef enum _E_HDMI_AUDIO_LENGTH_
{
    HDMI_AUD_LENGTH_16BITS = 0x00,
    HDMI_AUD_LENGTH_20BITS = 0x01,
    HDMI_AUD_LENGTH_24BITS = 0x02
}HDMI_AUDIO_LENGTH_E;

typedef enum _E_HDMI_AUDIO_CHANNEL_
{
    HDMI_AUD_2CH   = 0x01,
    HDMI_AUD_3CH   = 0x02,
    HDMI_AUD_4CH   = 0x03,
    HDMI_AUD_5CH   = 0x04,
    HDMI_AUD_6CH   = 0x05,
    HDMI_AUD_7CH   = 0x06,
    HDMI_AUD_8CH   = 0x07
}HDMI_AUDIO_CHN_E;

typedef struct _T_HDMI_CONFIG_PARA_
{
    UINT8  u8_hdmi_flag;           // FALSE = dvi out;  TRUE = hdmi out
    UINT8  u8_vic;                 // reference to CEA-861 VIC
    UINT16 u16_video_clk;          // TMDS video clk, uint 10000Hz
    UINT8  u8_clk_rpt;             // enum refer to HDMI_CLK_RPT_E. X2CLK = 480i/576i, others = X1CLK
    UINT8  u8_scan_info;           // enum refer to HDMI_SCAN_INFO_E
    UINT8  u8_aspect_ratio;        // enum refer to HDMI_ASPECT_RATIO_E
    UINT8  u8_color_space;         // enum refer to HDMI_CS_E
    UINT8  u8_color_depth;         // enum refer to HDMI_COLOR_DEPTH_E
    UINT8  u8_colorimetry;         // enum refer to HDMI_COLORIMETRY_E. IT601 = 480i/576i/480p/576p,
    ohters = IT709
    //
    UINT8  u8_video_format;        // enum refer to HDMI_VIDEO_FORMAT_E
    UINT8  u8_4Kx2K_vic;          // enum refer to HDMI1.4 extented resolution transmission
```



```

UINT8  u8_3D_structure;          // enum refer to HDMI_3D_STRUCTURE_E
//
UINT8  u8_audio_mode;           // enum refer to HDMI_AUDIO_MODE_E
UINT8  u8_audio_rate;           // enum refer to HDMI_AUDIO_RATE_E
UINT8  u8_audio_bits;           // enum refer to HDMI_AUDIO_LENGTH_E
UINT8  u8_audio_channels;       // enum refer to HDMI_AUDIO_CHN_E
UINT8  u8_audio_speaker_locations; // 0~255, refer to CEA-861 audio infoframe, BYTE4
}HDMI_CONFIG_T;

```

3.3MS_HDMITX_DEFINE

```

//HDMI TX channel
typedef enum _E_HDMI_TX_CHANNEL_
{
    HDMI_TX_CHN0      = 0x00,
    HDMI_TX_CHN1      = 0x01,
    HDMI_TX_CHN2      = 0x02,
    HDMI_TX_CHN3      = 0x03
}HDMI_CHANNEL_E;

typedef struct _T_HDMI_HDCP_RI_
{
    UINT8 TX_Ri0;
    UINT8 TX_Ri1;
    UINT8 RX_Ri0;
    UINT8 RX_Ri1;
}HDMI_HDCP_RI;

//HDMI EDID
typedef struct _T_HDMI_EDID_FLAG_
{
    UINT8  u8_hdmi_sink;          //1 = HDMI sink, 0 = dvi
    UINT8  u8_color_space;        //color space support flag, flag 1 valid. BITS: YCBCR444 flag; BIT4:
YCBCR422 flag.(RGB must be support)
//
    UINT8  u8_edid_total_blocks;  //block numbers, 128bytes in one block
    UINT16 u16_preferred_pixel_clk; //EDID Preferred pixel clock rate, u16_preferred_pixel_clk * 10000Hz,

```

ERROR code is 0xFFFF

```
UINT32    u32_preferred_timing;        //EDID Preferred Timing (Hact*Vact)
UINT8     u8_max_tmds_clk;              //HDMI VSDB max tmds clock, u8_max_tmds_clk * 5 Mhz
UINT32    u32_max_video_block_timing;  //EDID max video block timing (Hact*Vact)
UINT8     u8_hdmi_2_0_flag;            //1 = HDMI 2.0
}HDMI_EDID_FLAG_T;
```

3.4MS_DVIN_DEFINE

typedef struct T_DVIN_CONFIG

```
{
    UINT8 u8_cs_mode; //refer to DVIN_CS_MODE_E
    UINT8 u8_bw_mode; //refer to DVIN_BW_MODE_E
    UINT8 u8_sq_mode; //refer to DVIN_SQ_MODE_E
    UINT8 u8_dr_mode; //refer to DVIN_DR_MODE_E
    UINT8 u8_sy_mode; //refer to DVIN_SY_MODE_E
}DVIN_CONFIG_T;
```

typedef enum _E_DVIN_CS_MODE_

```
{
    DVIN_CS_MODE_RGB,
    DVIN_CS_MODE_YUV444,
    DVIN_CS_MODE_YUV422
}DVIN_CS_MODE_E;
```

typedef enum _E_DVIN_BW_MODE_

```
{
    DVIN_BW_MODE_16_20_24BIT,
    DVIN_BW_MODE_8_10_12BIT
}DVIN_BW_MODE_E;
```

typedef enum _E_DVIN_SQ_MODE_

```
{
    DVIN_SQ_MODE_NONSEQ,
    DVIN_SQ_MODE_SEQ
}DVIN_SQ_MODE_E;
```

```
typedef enum _E_DVIN_DR_MODE_  
{  
    DVIN_DR_MODE_SDR,  
    DVIN_DR_MODE_DDR  
}DVIN_DR_MODE_E;
```

```
typedef enum _E_DVIN_SY_MODE_  
{  
    DVIN_SY_MODE_HVDE,  
    DVIN_SY_MODE_HSVS,  
    DVIN_SY_MODE_VSDE,  
    DVIN_SY_MODE_DEOL,  
    DVIN_SY_MODE_C656,  
    DVIN_SY_MODE_1120,  
    DVIN_SY_MODE_1004  
}DVIN_SY_MODE_E;
```

4 API 使用说明

4.1 CHAR* ms7210_sdk_version(VOID);

- * Function name: ms7210_sdk_version
- * Description: get the SDK version
- * Input parameters: None
- * Output parameters: None
- * Returned value: version string

4.2 BOOL ms7210_chip_connect_detect(UINT8 u8_chip_addr);

- * Function name: ms7210_chip_connect_detect
- * Description: get the chip connect status
- * Input parameters: u8_chip_addr: 0 = auto check chip addr, others = set i2c slave addr
- * Output parameters: None
- * Returned value: connect status, 0: disconnect 1: connect

4.3 VOID ms7210_dvin_init(DVIN_CONFIG_T *t_dvin_config, UINT8

u8_spdif_in);

- * Function name: ms7210_dvin_mode_config
- * Description: config dvin mode, only config when system init
- * Input parameters: t_dvin_config: dvin mode select
- * u8_spdif_in: 0 = i2s, 1 = spdif with melk, 2 = spdif
- * Output parameters: None
- * Returned value: None

4.4 VOID ms7210_dvin_data_swap(UINT8 u8_swap_mode);

- * Function name: ms7210_dvin_data_swap
- * Description: set data pin swap mode
- * Input parameters: u8_swap_mode: 0 = D0~D23 swap, 1 = rb swap for rgb/yuv444,
2 = yc swap for sequential 16-bit yuv422
- * Output parameters: None
- * Returned value: None

4.5 VOID ms7210_dvin_phase_adjust(BOOL b_invert, UINT8 u8_delay);

- * Function name: ms7210_dvin_phase_adjust
- * Description: output phase adjust
- * Input parameters: b_invert: output clk invert
- * u8_delay: 1ns of one step, max 3ns
- * Output parameters: None
- * Returned value: None

4.6 VOID ms7210_dvin_timing_config(DVIN_CONFIG_T *t_dvin_config,

VIDEOTIMING_T *ptTiming, HDMI_CONFIG_T *pt_hdmi_tx);

- * Function name: ms7210_dvin_timing_config
- * Description: regeneration hs&vs&de, must do when input is not hs+vs+de
- * Input parameters: t_dvin_config: use for timing regeneration coefficient
- * ptTiming: use for timing regeneration
- * pt_hdmi_tx: may change output repeat times
- * Output parameters: None
- * Returned value: None

4.7 VOID ms7210_dvin_video_config(BOOL b_config);

- * Function name: ms7210_dvin_video_config
- * Description: enable/disable video pad input
- * Input parameters: b_config: 1 = enable output, 0 = disable output
- * Output parameters: None
- * Returned value: None

4.8 BOOL ms7210_hdmitx_hpd_detect(VOID);

- * Function name: ms7210_hdmitx_hpd_detect
- * Description: get tx hpd status
- * Input parameters: None
- * Output parameters: None
- * Returned value: tx hpd status

4.9 BOOL ms7210_hdmitx_edid_get(UINT8 *u8_edid);

- * Function name: ms7210_hdmitx_edid_get
- * Description: get tx edid
- * Input parameters: None
- * Output parameters: u8_edid: tx edid
- * Returned value: tx edid get success or not

4.10 BOOL ms7210_hdmitx_input_timing_stable_get(VOID);

- * Function name: ms7210_hdmitx_input_timing_stable_get
- * Description: get tx timing status
- * Input parameters: None
- * Output parameters: None
- * Returned value: tx timing stable status

4.11 VOID ms7210_hdmitx_output_config(HDMI_CONFIG_T *pt_hdmi_tx);

- * Function name: ms7210_hdmitx_output_config
- * Description: set tx output infoframe
- * Input parameters: pt_hdmi_tx
- * Output parameters: None
- * Returned value: None

4.12 VOID ms7210_hdmitx_shutdown_output(VOID);

- * Function name: ms7210_hdmitx_shutdown_output
- * Description: set tx output shutdown
- * Input parameters: None
- * Output parameters: None
- * Returned value: None

5 编译选项

5.1 平台相关编译选项

MS7200 SDK 支持多种编译平台，以下是在不同平台编译时需要增加的预编译选项：

- _PLATFORM_ARM_ARM: 32bit ARM 编译器
- __KEIL_C__: C51 Keil 编译器
- __CSMC__: STVD + COSMIC STM8 编译器
- _IAR_: IAR STM8 编译器
- _PLATFORM_WINDOWS_: MFC WIN32 编译器

6 SDK 所使用数据类型

__CODE : const 类型

__XDATA: C51 xdata

__DATA: C51 data

__IDATA: C51 idata

__NEAR: STM8 @near

__IO: volatile

UINT8: 8bit 无符号整型

INT8: 8bit 符号整型

BOOL: 布尔型

CHAR: 字符型

UINT16: 16bit 无符号整型

INT16: 16bit 符号整型

UINT32: 32bit 无符号整型

INT32: 32bit 符号整型

7. 应用指南

7.1 上电复位操作

MS7210 需要上电复位操作后才能使用 SDK 进行编程操作，具体操作如下：

- 首先将与复位 PIN 连接的 GPIO 拉成低电平；
- 保持该 GPIO 低电平时间不低于 1ms；
- 最后将该 GPIO 电平拉高完成上电复位动作。

7.2 外部 API 实现

7.2.1 延时函数

- extern VOID mcuilib_delay_ms(UINT8 u8_ms);

函数名：mcuilib_delay_ms

参数：u8_ms，8bit 无符号整型，代表延时毫秒数。

返回参数：无。

用途：毫秒级软件延时。

注意事项：无。

- 微妙延时：extern VOID mcuilib_delay_us(UINT8 u8_us);

函数名：mcuilib_delay_us

参数：u8_us，8bit 无符号整型，代表延时微秒数。

返回参数：无。

用途：微秒级软件延时。

注意事项：无。

7.2.2 I2C 读写操作

MS7200 的 IIC 地址可以根据 SA 引脚进行选择。当 SA 引脚上拉到电源电压或者悬空时，地址为 0xB2。当 SA 引脚连接到 GND 时，地址为 0x56。

- extern UINT8 mcuilib_i2c_read_16bitx8bval(UINT8 u8_address, UINT16 u16_index);

函数名：mcuilib_i2c_read_16bitx8bval

参数：u8_address，8bit 无符号整型，代表 I2C 器件的从地址；

u16_index，16 位无符号整型，代表所要读取器件内部寄存器的地址。

返回参数：UINT8，读取寄存器的返回值。

用途：16bit 寄存器 I2C 器件的读操作。

注意事项：

(1) 如何确认 IIC 单读函数通讯正常？芯片复位之后，读取以下寄存器

UINT8 Value;

//验证 IIC 读

LOG("r");

LOG("Checkk IIC Read Funtion!");

Value = mculib_i2c_read_16bidx8bval(0x56,0x0000);//寄存器地址 0x0000, 正常返回值: 0xA2

LOG1("ms7210 chip reg0 = ", Value);

Value = mculib_i2c_read_16bidx8bval(0x56,0x0001);//寄存器地址 0x0001, 正常返回值: 0x20

LOG1("ms7210 chip reg1 = ", Value);

Value = mculib_i2c_read_16bidx8bval(0x56,0x0002);//寄存器地址 0x0002, 正常返回值: 0x0A

LOG1("ms7210 chip reg2 = ", Value);

Value = mculib_i2c_read_16bidx8bval(0x56,0x1000);//寄存器地址 0x1000, 正常返回值: 0x27

LOG1("ms7210 chip reg1000 = ", Value);

Value = mculib_i2c_read_16bidx8bval(0x56,0x1001);//寄存器地址 0x1001, 正常返回值: 0xA6

LOG1("ms7210 chip reg1001 = ", Value);

LOG("r");

- 16bit 寄存器地址写操作: extern BOOL mculib_i2c_write_16bidx8bval(UINT8 u8_address, UINT16 u16_index, UINT8 u8_value);

函数名: mculib_i2c_write_16bidx8bval

参数: u8_address, 8bit 无符号整型, 代表 I2C 器件的从地址;

u16_index, 16 位无符号整型, 代表所要写入器件内部寄存器的地址;

u8_value, 8bit 无符号整型, 代表所要写入目标寄存器的值。

返回参数: UINT8, 读取寄存器的返回值。

用途: 16bit 寄存器 I2C 器件的读操作。

(1) 如何确认 IIC 单写函数通讯正常? 读写以下寄存器:

//验证 IIC 写

LOG("Check IIC Wirte Funtion!");

mculib_i2c_write_16bidx8bval(0x56,0x0003,0x5A);//寄存器地址 0x0003, 写入值: 0x5A

Value = mculib_i2c_read_16bidx8bval(0x56,0x0003);//寄存器地址 0x0003, 正常返回值: 0x5A

LOG1("ms7210 chip reg3 = ", Value);

mculib_i2c_write_16bidx8bval(0x56, 0x1000, 0x5A);//寄存器地址 0x1000, 写入值: 0x5A

Value = mculib_i2c_read_16bidx8bval(0x56, 0x1000);//寄存器地址 0x1000, 正常返回值: 0x5A

LOG1("ms7210 chip reg1000 = ", Value);

LOG("r");

7.3 调用举例

以下以 24bit SDR 视频+I2S 信号输入, HDMI 输出的例子来举例说明如何用 SDK 搭建一个实例(详见 Sample Code):

- 芯片初始化;
- 配置芯片输入输出相关模块初始化;
- 检测输入变化并配置。

7.3.1 芯片初始化

初始化的过程, 首先是要对芯片进行上电复位; 其次可以通过调用 `API ms7210_chip_connect_detect` 完成对系统上 MS7210 芯片 I2C 连接的检查, 并设置正确连接的 I2C 从地址。

7.3.2 配置芯片输入相关模块初始化

配置 DVIN 初始化, 包括输出的色彩空间 (CS: RGB、YUV444、YUV422), 位宽 (BW: 16/20/24、8/10/12), Mapping 方式 (SQ: Sequential、Non Sequential), 速率 (DR: SDR、DDR), Sync (BT601、BT656、BT1120), 音频格式 (I2S、SPDIF)。

7.3.3 检测输出状态变化并配置

将输入 timing 的信息配置到 DVIN 模块, 用于系统内部的时钟选择和 Timing 生成。

HDMI_TX 支持实时检测模块内 timing stable 状态, 可根据变化情况, 打开或关闭 HDMI_TX 的输出。



附录：版本历史

文档版本	撰写时间	变更内容
V1.0	2020.12.8	初版
V1.01	2021.01.15	1. 添加 IIC Address 2. 添加 IIC 读写函数验证方法
V1.10	2021.01.15	1. 修正 IIC 读写函数验证方法 2. 删除 IIC 连读连写函数
V1.11	2021.02.26	1. 补充 IIC Address
V1.12	2021.11.19	更新模板