

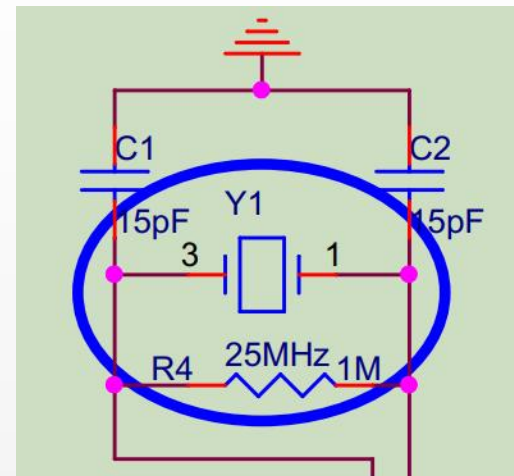
LT8912B

MIPI 转 HDMI & LVDS

软/硬件方面调试

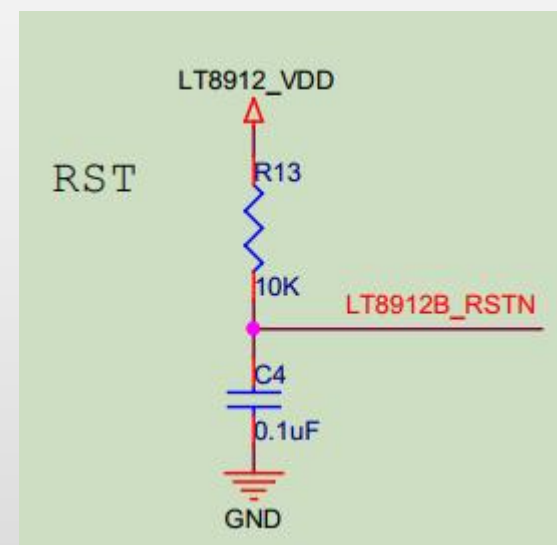
1.1、硬件review:

1、晶振选择25M，中间并一个1M的反馈电阻，反馈电阻不能省。



2、LT8912B的复位脚，需要接到主控SOC（或者单片机）的GPIO去控制，低电平复位。

3、LT8912B的slave IIC 只支持1.8V 电平，如果主控SOC的IIC bus不是1.8V 电平，需要做电平转换。

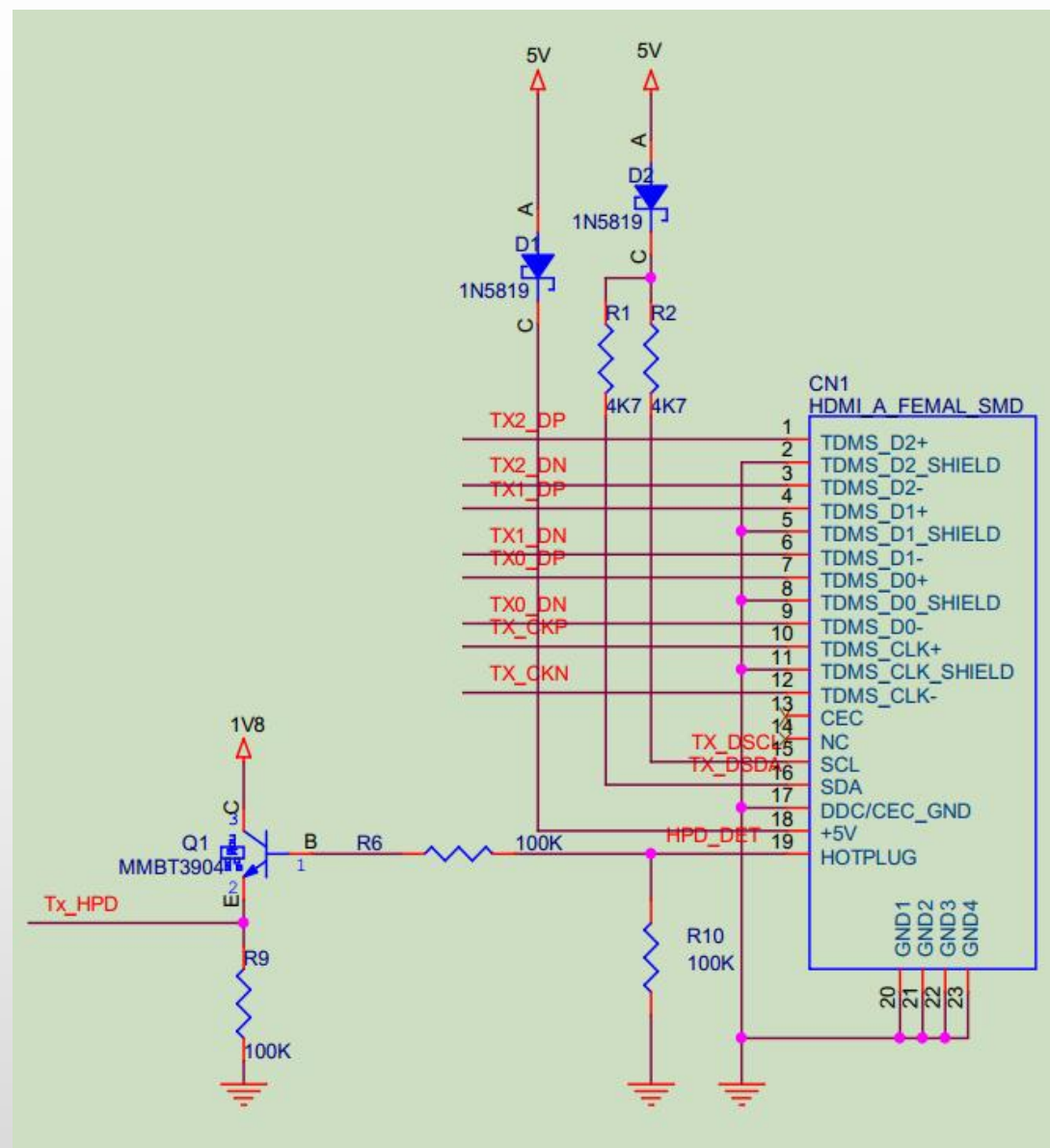


1.2、硬件review:

4、HDMI 座子的DDC接口的上拉电阻和HDMI 5V接二极管到5V，防止后端电压反串到系统5V。

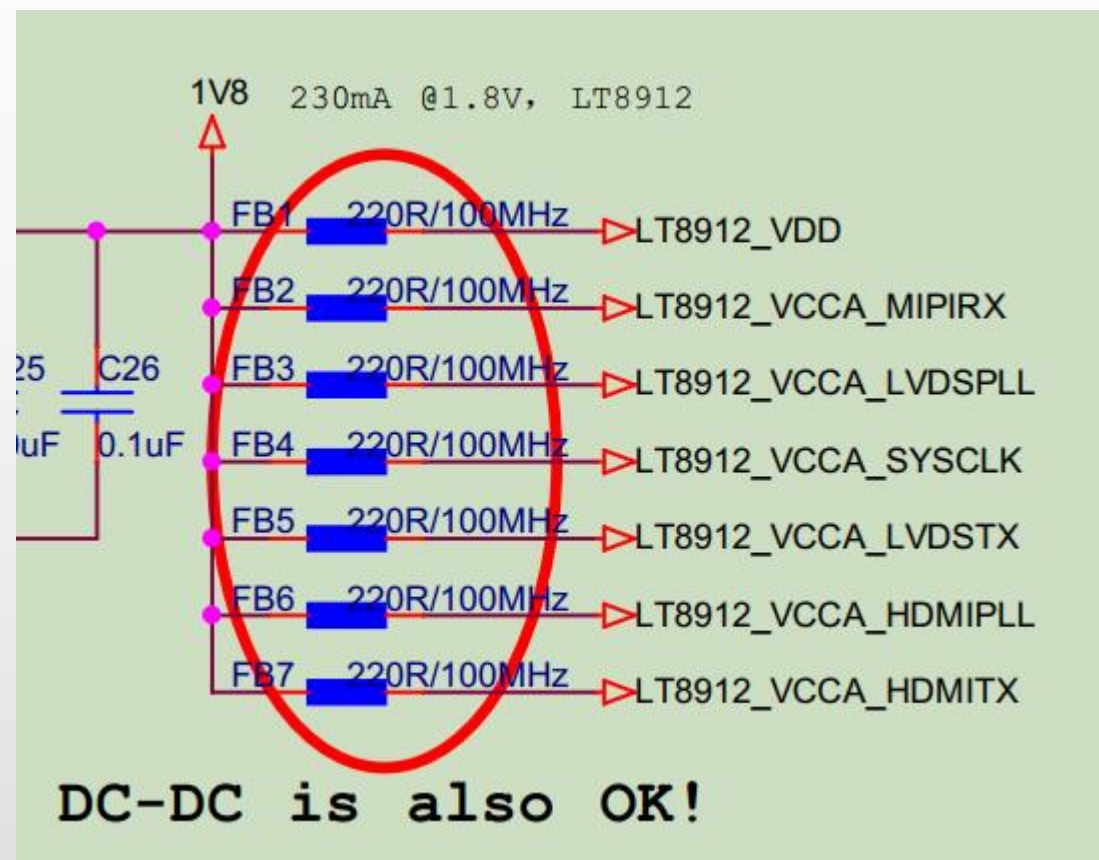
如果要读sink端的EDID，需要将主控SOC的IIC bus接到HDMI DDC，IIC 地址是0xA0。

5、LT8192B的HPD 输入只支持1.8V电平。



1.3、硬件review:

6、LT8912B的不同电源输入，需要用磁珠隔开。
不建议用0欧姆电阻。



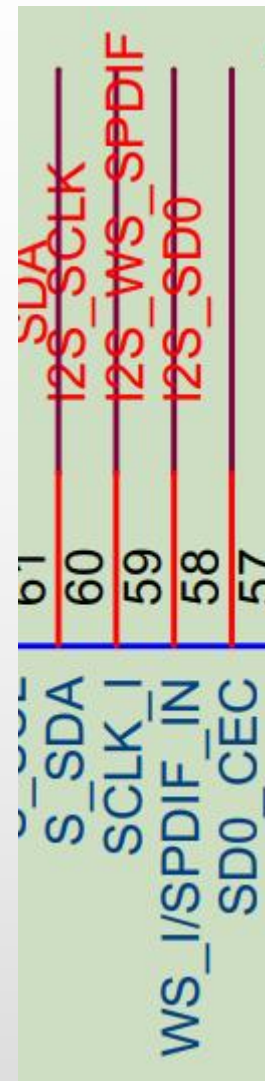
1.4、硬件review:

7、对于IIS音频输入，LT8912B是slave。

- 关于IIS:

- SCLK 串行时钟，也叫位时钟 (BCLK)，即对应数字音频的每一位数据，SCLK都有一个脉冲， $SCLK = 2 * \text{采样频率} * \text{采样位数}$ 。
- WS 也叫帧时钟 (LRCK)，用于切换左右声道的数据，WS的频率等于采样频率。
- SD0 是串行数据，就是用二进制补码表示的音频数据。
- MCLK (主时钟/系统时钟)，这个是为了使系统间能够更好的同步，如果有就接上，不接也没影响；(LT8912B 没有MCLK输入脚)

- 也可以选择SPDIF 音频输入。
- 如果输出的HDMI不需要音频 (DVI)，IIS和SPDIF可以不接，管脚悬空就可以。



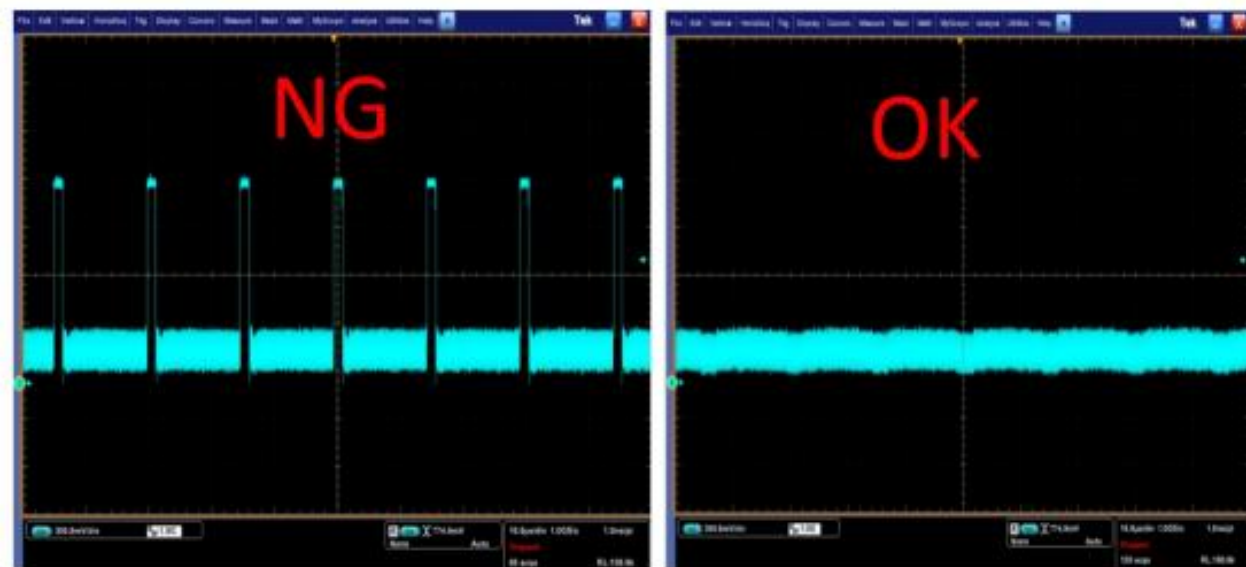
2.1、LT8912B 对输入MIPI 信号的要求1

- LT8912B只支持DSI 的MIPI 输入。
 - Compliant with D-PHY1.1 and DSI1.02
 - 不支持CSI的MIPI。
 - 不支持通过MIPI 下command。

- LT8912B 对输入MIPI 信号的要求：

- MIPI 信号需要关闭展频（SSC）。
- MIPI CLK 不能进LP（见右图）。

LT8912B 要求MIPI Clock Lane 是连续的，不要进入LP mode。如下图所示：



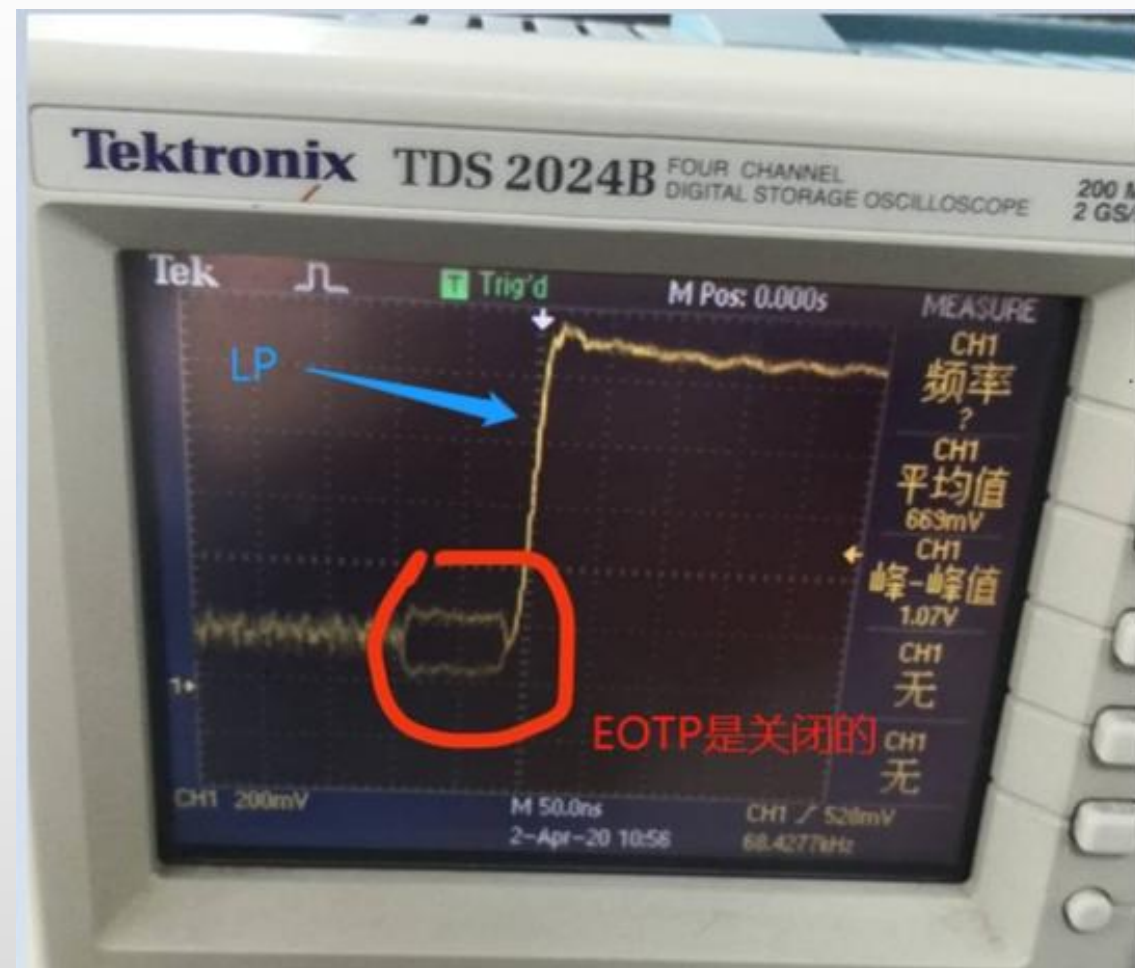
2.2、LT8912B 对输入MIPI 信号的要求2

- MIPI data 的一个采样周期不能有两个LP。



2.3、LT8912B 对输入MIPI 信号的要求3

- MIPI 信号需要关闭EOTP。
 - 用示波器测量MIPI D0+ 或者MIPI D0- 的波形，如果下图圈起来的地方（进入LP之前）电平一直在变动，说明EOTP是关闭的，如果保持为高，或者保持为低，说明EOTP是打开的。
 - LT8912B 要求EOTP是**关闭**的。
- 进入LP之前的那一段是叫trail的东西，是按照最后一个数据包的状态决定1还是0，trail的高电平或者低电平就是最后一个byte的bit7的反相。
- Eotp是一个短包，内容是固定的（0x08、0x0F、0x0F、0x01），分配到每条lane（P lane）后每次进入EOT波形就是固定的，最后一个byte的bit7都是0，反相之后，trail是固定高电平。
- 如果没有打开eotp，那么最后传输的那个byte就是随机的，trail的高电平/低电平就是最后一个byte的bit7的反相，所以示波器看到的trail就是跳动的。



3.1、LT8912B 输出信号

- HDMI 和 LVDS 可以同时输出，也可以分别关闭，只输出HDMI 或者LVDS。
- HDMI 输出：
 - Support HDMI1.4 standard
 - 输出最大支持1080P60 HDMI / DVI 分辨率。
 - MIPI 转 HDMI 是bypass的，不支持分辨率缩放。
 - HDMI 输出没有DDC，不支持HDCP 加密。
- LVDS输出：
 - LVDS 输出只有单port (6bit/8bit) .
 - MIPI 转 LVDS 输出支持分辨率缩放（最大到1080P60 分辨率）。

4.1、软件调试--复位控制

8.6 Power-up and Reset Sequence

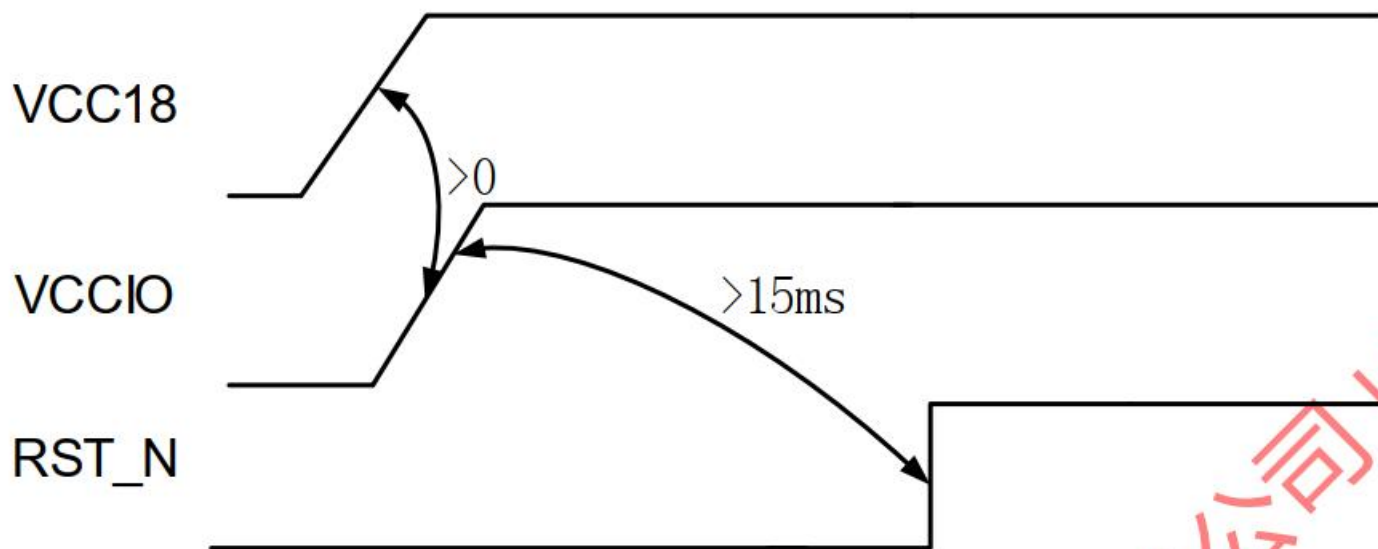


Figure 8.6.1 Power-up and Reset Sequence

上电时序，1.8V先上电或者同时上电，
上电后务必Reset执行低(>15ms)->高 (10ms) ->加载驱动；

4.2、软件调试--IIC

- LT8912B内部有三个IIC 模块，IIC 地址分别是0x90/0x92/0x94，分别负责不同的功能配置。
- 如果SOC是Linux系统，bit7作为读写标志位，SOC端的IIC bus地址需要右移一位，既是0x48/0x49/0x4A。
- IIC 速率不要超过**100K**Hz。

LT8912B 调试问题:			
1、IIC 不通:	A、电源（1.8V）、晶振（25MHz）、复位（低电平复位，正常工作时为高）是否正常；IIC是否有接上拉电阻到1.8V；芯片底部的EPAD要接地。	如果IIC bus上挂的IIC从器件比较多，IIC上的上拉电阻不要太大，1K ~ 2K 之间比较合适。	
	B、LT8912B的IIC地址是 0x90/0x92/0x94 ，如果SOC是Linux系统，bit7作为读写标志位，SOC端的IIC bus地址需要右移一位，既是 0x48/0x49/0x4A 。	// 读LT8912B ID值的操作 如下： IIC_Addr = 0x90; //bit0 是读写标志位； Temp_ID0 = HDMI_ReadI2C_Byte(0x00); // 读 0x00 寄存器的值，正常值是0x12 Temp_ID1 = HDMI_ReadI2C_Byte(0x01); // 读 0x01 寄存器的值，正常值是0xB2 // HDMI_ReadI2C_Byte(Reg_Addr); 是标准 IIC 读操作函数， IIC 速率不要超过100KHz 。	
	C、IIC 速率不要超过100KHz。		
	D、可以用逻辑分析仪抓IIC 波形，可以看到IIC bus上写的是什么数据，有没有Ack。		

4.2.1、软件调试--检测MIPI输入信号 (video check)

读0x90 IIC地址的0x9d ~ 0x9F 寄存器值，可以知道输入MIPI 信号的V total值，通过计算可以知道H total值。

```
// for debug
void LT8912B_Video_Check( void )
{
    u16 H_total_Cnt=0x0000;
    u16 V_total = 0x0000;

    I2CADDR = 0x90;

    H_total_Cnt = LT8912B_I2C_Read_Byte(0x9d)*256 + LT8912B_I2C_Read_Byte(0x9c);

    V_total = ( LT8912B_I2C_Read_Byte( 0x9f ) & 0x07 ) * 256 + LT8912B_I2C_Read_Byte( 0x9E );

    if( V_total >= 0x200 && V_total <= 0x21d )    // 0x20D
    {
        videoformat = VESA_720x480_60;           /**/
    }else
    if( V_total >= 0x2E0 && V_total <= 0x2Fe )    //0x2EE
    {
        videoformat = VESA_1280x720_60;           /**/
    }else
```



4.2.2、软件调试--检测MIPI输入信号 (video check)

如果读video check不对:

- 1、确保输入的MIPI 信号波形是对的 (参考2.1 ~ 2.3 LT8912B 对输入MIPI 信号的要求) 。
- 2、MIPI 信号输入线序不对、或者P/N反了。
- 3、芯片虚焊, MIPI 信号没有给到LT8912B。
- 4、信号质量不好, 需要设置LT8912B EQ, (0x90 IIC地址, 0x3e寄存器 (Mipi lp rx trigger point select) , 0x90 IIC的地址, 0x41寄存器 (Mipi hs rx current select))
- 5、LT8912B的MIPI RX time settle 设置太大, (0x92 IIC 地址, 0x11寄存器 (寄存器配置是0x08, 改成0x04试试), 是控制在term 打开以后, 多久开始接受高速数据)
- 6、上电没有对芯片进行复位 (需要用GPIO控制复位)

当video check 能读到稳定的值, 说明MIPI已经稳定的接受信号了, 接下来需要确认是否能稳定的恢复video的时钟。

读(IIC addr:0x92) 0x0c ~ 0x0f 4个寄存器 (0x0f寄存器是最高位, 一般都是0x00), 如果MIPI源发生的信号的pixel clk是148.5M, 那么理论上读到的值应该是0x32/0x33/0xD3, 允许低位跳动, 既是0x0c寄存器值跳动, 其它三个寄存器保持稳定, 那么就说明成功恢复了pixel clk。

如果pixel clk的值一直不稳定,

- 1、请确认MIPI源的屏参 和 LT8912B的MIPI timing配置完全相同。
- 2、关闭MIPI CLK的展频, 关闭EOTP。

4.3、LT8912B的IIS/SPDIF 输入音频配置

```
void Audio_Config( void )
{
    // Audio config
    I2CADR = 0x90;
    LT8912B_I2C_Write_Byte( 0xB2, 0x01 );    // 0x01:HDMI; 0x00: DVI

#ifdef 1    // IIS input
    // AudioIISEn(); // IIS Input
    I2CADR = 0x94;
    LT8912B_I2C_Write_Byte( 0x06, 0x08 );    // 0x09
    LT8912B_I2C_Write_Byte( 0x07, 0xF0 );    // enable Audio: 0xF0; Audio Mute: 0x00

    LT8912B_I2C_Write_Byte( 0x09, 0x00 );
    // 0x00 :Philips spec; default
    // 0x04 :Left justified;
    // 0x06 :Right justified;(Japan spec)
#else // SPDIF input
    // AudioSPDIFEn(); // SPDIF Input
    I2CADR = 0x94;
    LT8912B_I2C_Write_Byte( 0x06, 0x0e );
    LT8912B_I2C_Write_Byte( 0x07, 0x00 );
    /*
#endif

    LT8912B_I2C_Write_Byte( 0x0f, 0x0b + Sample_Freq[_48KHz] );

    LT8912B_I2C_Write_Byte( 0x37, (u8)( IIS_N[_48KHz] / 0x10000 ) );
    LT8912B_I2C_Write_Byte( 0x36, (u8)( ( IIS_N[_48KHz] & 0x00FFFF ) / 0x100 ) );
    LT8912B_I2C_Write_Byte( 0x35, (u8)( IIS_N[_48KHz] & 0x0000FF ) );

    LT8912B_I2C_Write_Byte( 0x34, 0xD2 );    // 32 bit的数据长度
    // LT8912B_I2C_Write_Byte( 0x34, 0xE2 ); // 16 bit的数据长度

    LT8912B_I2C_Write_Byte( 0x3c, 0x41 );    // Null packet enable
} // end Audio_Config
```

1、0x90 IIC地址，0xB2寄存器的bit0是设置选择HDMI / DVI 输出的。

2、0x94 IIC地址，0x09寄存器是设置IIS 格式的，有。

3、0x94 IIC地址，0x34寄存器，不同的数据长度，设置的值不一样，如果设置值不对，HDMI会没有声音。

4、0x94 IIC地址，0x35 ~ 0x37 根据不同的采样率设置不同的值。

通过测量SCLK 和 WS的频率，可以算出音频数据长度。

$SCLK = WS * 2 * \text{数据长度}$

$MCLK = SCLK * 4$

常见采样率是：32K、44.1K、48K、88.2K、96K、176K、196K

4.4.1、LT8912B--HDMI swing设置

0x90 IIC 地址, 0x31 ~ 0x33 寄存器, 设置HDMI 信号swing;

0x33 寄存器的bit0 是HDMI TX的使能位, 为0是关闭HDMI 输出, 为1是打开HDMI 输出。

```
// TxAnalog();  
I2CADDR = 0x90;  
#ifdef _TurnOff_HDMI_DVI_  
    LT8912B_I2C_Write_Byte( 0x33, 0x00 ); // bit0/bit1 =1 Turn On HDMI Tx;   bit0/bit1 = 0 Turn Off HDMI Tx  
#else  
    LT8912B_I2C_Write_Byte( 0x31, 0xa1 );  
    LT8912B_I2C_Write_Byte( 0x32, 0xbf );  
    LT8912B_I2C_Write_Byte( 0x33, 0x17 ); // bit0/bit1 =1 Turn On HDMI Tx;   bit0/bit1 = 0 Turn Off HDMI Tx  
#endif  
LT8912B_I2C_Write_Byte( 0x37, 0x00 );  
LT8912B_I2C_Write_Byte( 0x38, 0x22 );  
LT8912B_I2C_Write_Byte( 0x60, 0x82 );
```

4.4.2、LT8912B--HDMI swing设置

0x90 IIC地址, 0x31 寄存器 (HDMI CLK swing 控制), 寄存器配置的值是 0xa1, 这个寄存器值可以调整一下试试看

1.2.2 HDMI/MHL TxPHY control register list

Device Address:0x90

0x31				
Bit	Name	Description	Default	Mode
7-1	RG_HDMITX_CLK_SWING[6:0]	Hdmi/mhl tx clock swing control. hdmi clk: 0000000~1111111, 0~800mV, 6.25mV per bit. mhl clk: 00000xx~11111xx, 0~870mV, 28mV per bit.	7'h50	R/W
0	RG_HDMITX_HDMI_CLK_EN	Hdmi tx tmds clock enable. 0: disable.	1'b0	R/W

4.4.3、LT8912B--HDMI swing设置

0x90 IIC地址，0x32 寄存器（HDMI data tap0 swing控制），寄存器配置的值是 0xbf，这个寄存器值可以调整一下试试看。

Device Address:0x90				
0x32				
Bit	Name	Description	Default	Mode
7-1	RG_HDMITX_TAP0_SWING[6:0]	Hdmi tx data tap0 swing control. 0000000~1111111, 0~800mV, 6.25mV per bit.	7'h50	R/W
0	RG_HDMITX_HDMI_TAP0_EN	Hdmi tx tap0 enable. 0: disable. 1: enable.	1'b0	R/W

Default Value: 0xA0

4.4.4、LT8912B--HDMI swing设置

0x90 IIC地址, 0x33 寄存器 (HDMI data tap1 swing 控制) , 寄存器配置的值是 0x17, 这个寄存器值可以调整一下试试看。

Device Address:0x90

0x33				
Bit	Name	Description	Default	Mode
7-6	RESERVED		2'h0	R/W
5-3	RG_HDMITX_TAP1_SWING[2:0]	Hdmi/mhl tx data tap1 swing control. 000~111, 0~350mV, 50mV per bit.	3'h0	R/W
2	RG_HDMITX_HDMI_TAP1_EN	Hdmi tx data pre-emphasis enable. 0: disable. 1: enable.	1'b0	R/W
1	RG_HDMITX_HDMI_EN	Hdmi tx enable. 0: disable. 1: enable.	1'b0	R/W
0	RG_HDMITX_HVCOM_SNESE_EN	Hdmi tx vcom sense enable. 0: disable. 1: enable.	1'b1	R/W

Default Value: 0x01

4.5、LT8912B HDMI输出 没显示

- 1、检查芯片电源、晶振、复位等外围硬件是否正常。
 - a、LT8912B的35脚接2.2nF电容到1.8V --- 确认电容值是否正常。
 - b、确认有给HDMI座子的18脚提供+5V电源。
- 2、确保IIC 通讯正常，寄存器配置正常写入。
 - a、如果IIC总线上挂了几个IIC 从设备，IIC bus上的上拉电阻需要减小，比如1K。
- 3、可以配置LT8912B 输出自测试的HDMI 信号，看是否能显示。
- 4、确认MIPI 信号线序、P/N 的硬件连接跟寄存器配置是对应的，如果线序或者P /N 跟参考设计不一样，记得改芯片寄存器配置。
- 5、读LT8912B的Video check，看是否检测到MIPI信号输入，如右图，读0x90 IIC 地址，0x9C ~ 0x9F 4个寄存器。
- 6、测量MIPI CLK和data的波形，看是否符合要求。

4.6、LT8912B HDMI输出闪屏

- 1、先设置成DVI输出，看显示是否稳定。
 - 如果设置DVI输出，显示是稳定的，那应该是信号timing的问题，需要调整输入信号的timing。
 - 因为音频信号是在视频信号的消隐区里面传输的，所以，如果一个视频信号的H/V blanking 过小（或者是H/V BP过小，而H/V FP过大也会有影响），导致音频传输时间不够，会影响到视频的显示。
- 2、如果可以，最好是按VESA 标准timing配置MIPI输入信号，因为LT8912B只负责转换，MIPI输入，HDMI输出是bypass的；如果输入信号分辨率是非标，那么输出的HDMI也是非标，不保证后端sink支持这种非标的HDMI信号。

4.7、LT8912B HDMI 图像是斜的，而且左右移动

- 确保MIPI data的EOTP是关闭的。
 - EOTP是否关闭，请参考2.3 里面的波形.
 - a、MTK平台 的 `dis_eotp_en` 的值，改成 `TRUE`； LK和kernel都需要修改成 `TRUE`
 - b、展讯平台的 `tx_eotp` 的值要置 0 。
 - c、高通 `cfg->append_tx_eot = false.`

4.8、LT8912B有图像没声音

- 1、确保IIS/SPDIF 信号有给到LT8618SXB，SCLK 和 WS 没接反。
- 2、有正确IIS/SPDIF 输入音频配置。
- 3、确定是设置HDMI输出，而不是DVI。
- 4、确保IIS格式设置是对的（IIS格式有三种：飞利浦格式（常用），左对齐（较少使用），右对齐（日本格式，也叫普通格式）），包括采样率、IIS 数据宽度的设置。
- 5、0x94 IIC地址，0x34寄存器设置的值跟输入对应（32bit数据长度，寄存器设置0xD2；16bit，设置0xE2）；测IIS的SCLK和WS的频率，数据位宽= $SCLK/(WS*2)$ 。
- 6、采样率设置跟输入的对应。

5.1、关于视频信号屏参的常用公式：

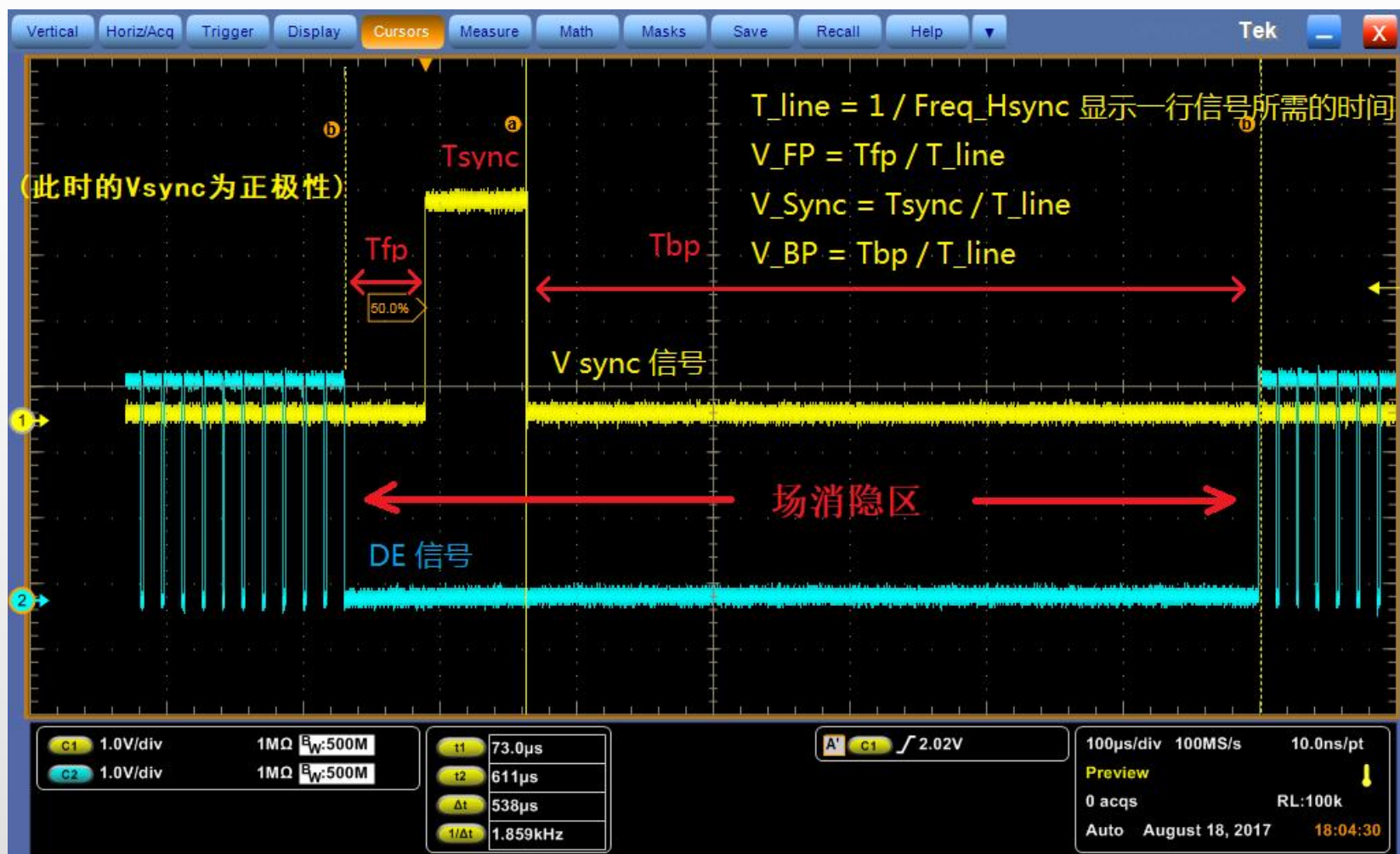
- $H_{total} = H_{active} + H_{BackPorch} + H_{FrontPorch} + H_{syncWidth} = H_{active} + H_{Blanking}$
- $V_{total} = V_{active} + V_{BackPorch} + V_{FrontPorch} + V_{syncWidth} = V_{active} + V_{Blanking}$
- $H_{Blanking} = H_{BackPorch} + H_{FrontPorch} + H_{syncWidth}$
- $V_{Blanking} = V_{BackPorch} + V_{FrontPorch} + V_{syncWidth}$

- $Pixel_CLK = H_{total} * V_{total} * \text{帧率(Hz)} = H_{total} * \text{行频(KHz)}$
- $\text{行频(KHz)} = V_{total} * \text{帧率(Hz)}$
- $H_{total} = Pixel_CLK / \text{行频(KHz)}$
- $V_{total} = \text{行频(KHz)} / \text{帧率(Hz)}$

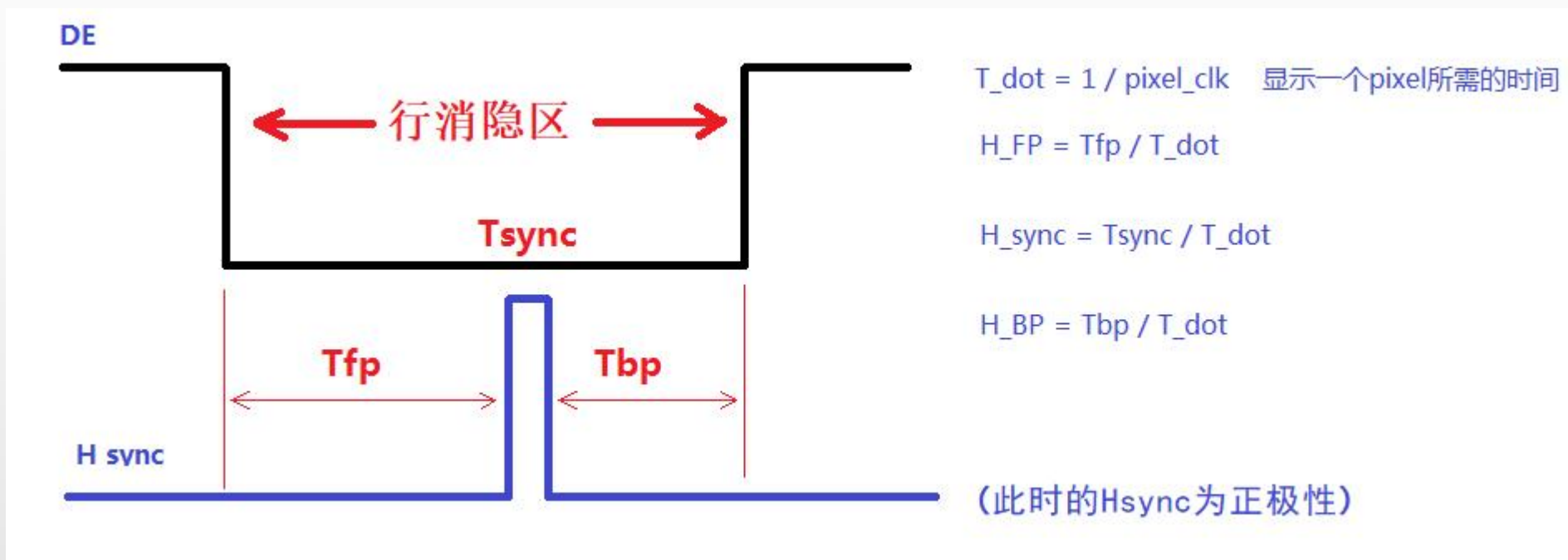
5.2、关于视频信号屏参的常用公式（MIPI）：

- $\text{MIPI_CLK} = \text{Pixel_CLK} * 3 * 8\text{bit} / (2 * 4 \text{ Lane}) // \text{RGB888}$
- $\text{MIPI_bit_rate} = \text{MIPI_CLK} * 2;$
- $\text{MIPI_byte_CLK} = \text{MIPI_bit_rate} / 8 = \text{MIPI_CLK} / 4;$

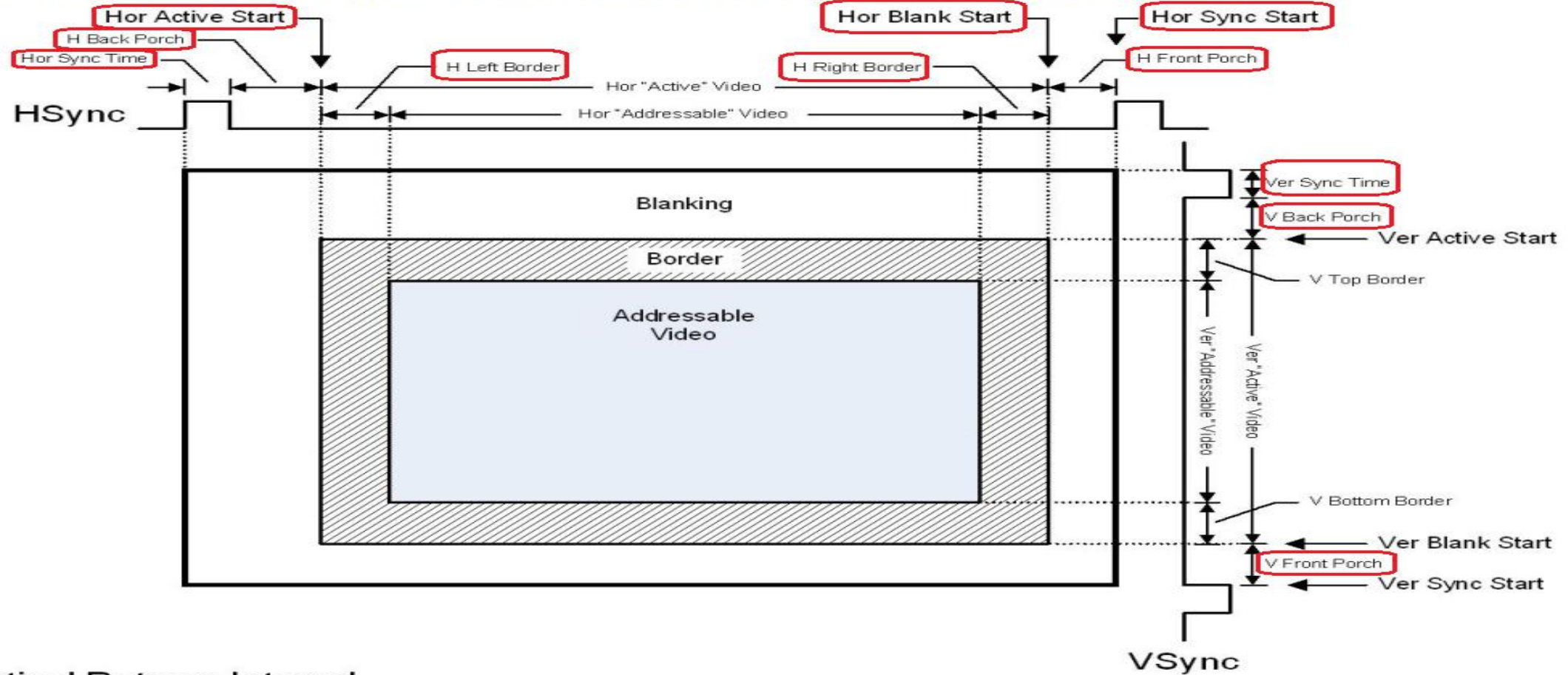
5.3、V sync & DE



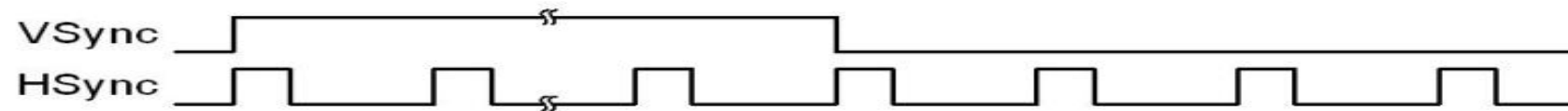
5.4、Hsync & DE



3.5 DMT Video Timing Parameter Definitions - Total Frame Timing:



Vertical Retrace Interval



Note:

All syncs shown as active high. For active low invert the waveform as shown below

