

Step1

Gamma 调整

1、Gamma 是为了矫正显示器件显示的亮度和输入的信号之间的关系，以达到人眼对亮度感知的均匀性，所以 gamma 曲线的调整应该满足 PANEL 的显示亮度与输入信号之间成二次关系的曲线：

$L=L_{\max} * (\text{输入信号值}/\text{最大信号值})^a$;

L_{\max} 是 PANEL 的最大亮度， a 是乘方的系数，也就是我们常说的 gamma 的系数；

对于大多数的 LCD PANEL 我们推荐使用 $a=2.2$;

具体可使用附件的 gamma 计算的工具；

2、因为 PANEL 都会有自己的 gamma buffer，而且 panel 是信号流程的最后端，所以 Panel 本身的 gamma 对我们调整 gamma 有至关重要的影响，可是使用我们的 internal pattern 来量测 panel 本身的 gamma 曲线，如果 panel 本身的 gamma 曲线与我们要得到的曲线偏差非常大，一般很难通过 chip 端来矫正，而且有可能会带来 contour；

3、Gamma 除了可以用来矫正 panel 的亮度曲线，还可用来矫正 Panel 色温；

可以首先使用我们的 internal pattern 来测量一下 panel 本身的色温，如果 panel 本身的色温与我们的目标值相差不是很大，并且色温的一致性也比较好，可使用三条曲线完全相同的 gamma 曲线，而色温可以直接通过调整 gain 和 offset 来满足色温的要求；

如果是 panel 本身色温的一致性不是很好，可以通过调整 gamma 来矫正；

如果 panel 本身的色温距离我们的目标值相差太大，很难通过调整 gamma 来实现，因为这样比较容易造成某一条曲线的不连续而造成 contour；

通过 gamma 矫正色温时，如果 panel 的最大亮度的色温与我们的目标值相差较大，需要降低 gamma 最大值的方法来实现；

4、Gamma 的量测和调试条件：

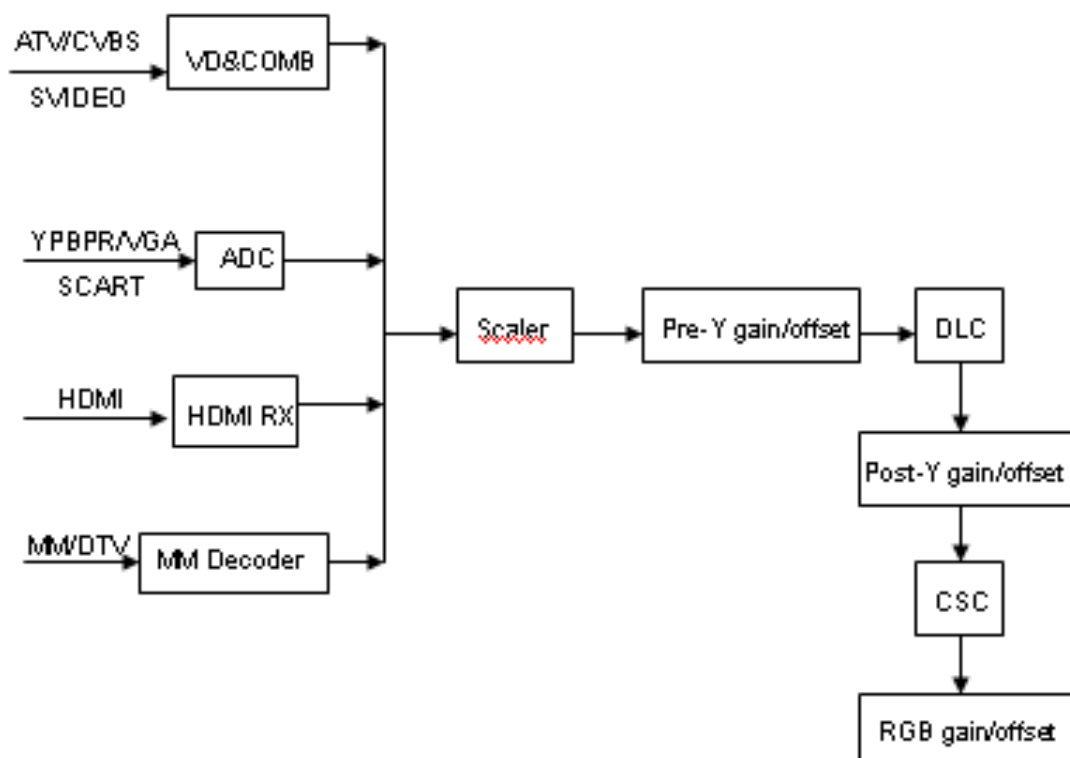
- (1) 在 HDMI source 下；
- (2) 关掉 DLC/BLE/WLE；
- (3) 将亮度和对比度都设置到中间值 0x80；
- (4) 将 RGB Gain/Offset 设置为中间值 0x80；

Step2

亮度的调整

1、亮度调整的前提需要先将 gamma 确定下来,因为 gamma 处于整个信号处理的最后端,gamma 重新调整会影响最后亮度的整体表现,如果调试后期 gamma 更改且变化比较大,亮度调整需要重新确认,所以需要先将 gamma 确定下来再来做亮度方面的调整;

2、在 YUV 的彩色空间里面,亮度调整需要满足 ITU-R BT. 601/BT. 709 的要求,即是 Y level 的 Range 是 16-235,这样才能保证信号的对比度要求,调试方法按照信号流程从前往后逐级进行调整,有关亮度的相关信号流程图如下:



我们可以使用 Chip 新增加了一个 debug mode 来读出我们每个 block 的 Y、CB、CR 的 data, 具体说明如下:

BK1A_D0[0]: Show pixel value for debug (only color engine block);

BK1A_D0[7-4]: debug location;

0000 : input data//0001 : noise mask top//0010 : v_noise_mask_filter//0011 : FCC//0100 : IHC//0101 : ICC//0110 : Y_switch//0111 : Y_curve_fit//1000 : UV compensate//1001 : Output data;

BK1A_D2/D3[10-0]: Debug pixel H-position;

BK1A_D4/D5[10-0]: Debug pixel V-position;

BK1A_D6/D7[9-0]: Debug pixel Y value;

BK1A_D8/D9[9-0]: Debug pixel Cb value;

BK1A_DA/DB[9-0]: Debug pixel Cr value;

将 BK1A_D0[0]=1, 屏幕左上角会出现一个“+”符号, 改变 BK1A_D2/D3[10-0]和 BK1A_D4/D5[10-0]将“+”挪到你希望读出 data 的地方, BK1A_D6/D7 就可以读出这个 pixel

的 Y data, BK1A_D8/D9[9-0] 可以读出这个 pixel 的 Cb 的 data, BK1A_DA/DB[9-0] 可以读出这个 pixel 的 Cr 的 data, 我们还可以用 BK1A_D0[7-4] 来确定我们读出 data 的位置, 需要转化为 8bit 的话将 Cb 和 Cr data 除以 4;

我们首先来确认各通道信号前端送进来的 Y level:

(1)、CVBS 信号, 因为要经过 VD 和 Comb filter, 所以首先需要确定 VD AGC 的方式, fixed 或者 auto, 一般而言我们 RF 使用 fixed, 而 AV 用 auto, 同时确认一下 BK_COMB_80、81、82=0xC8、0x96、0x6A, 然后分别输入 0IRE 和 100IRE 的信号, 将 BK1A_D0 设置为 0x01, 读出 Y 的 data, 通过调整 BK_COMB_74 和 BK_COMB_73 确保信号的输出范围是 16-235;

(2)、Componet 和 VGA, 因为 YPBPR 和 RGB 需要经过 ADC, 所以需要首先进行 ADC 校正, 完成后分别输入 0IRE 和 100IRE 的信号, 用工具读出 Y data, YPBPR 的范围应该是 16-235, 而 RGB 应该是 0-255, 大家也可以用这种方法来确认我们的 ADC 校正是否准确; 因为是模拟信号的缘故一般而言会有些偏差但是不应该太大;

(3)、HDMI 和 DTV, 因为是数字信号所以可以不用确认, 但是可以用来确认信号源的范围是否正确;

下一步需要进入 DLC 的 Y Level:

因为前端送进来的信号是 16-235, 而我们 DLC 的 curve 是 0-255, 所以需要进行转换再进 DLC 才能避免 DLC 做错;

首先把 debug mode 打开, 将 BK1A_D0 设置为 0x61, 这时候读出来的 Y Data 的值就是经过了 pre gain 和 offset 后进 DLC 的值, 我们有部分 IC 是先过 gain 再过 offset, 造成我们计算不能正好转换为 0-255, 但是配合 DLC 的参数还是能解决这个问题:

直接将 16-235 乘以 1 减 16, 也就是将 BK1A_2C 设置为 0x40, BK1A_1E 设置为 0xF0, 这样就 0-219 进 DLC, 将 DLC 的 Luma_limit 设置为 3, 这样就能保证信号的最亮和最暗不动, 而经过 DLC 之后转换成 RGB 时 Y 值不变, 不需要再减 16 进矩阵;

而有部分 Chip 是先过 offset 再过 gain, 这样就可以比较容易转换为 0-255, 但是注意出 DLC 后再乘 219/255, 这样保证进矩阵的 range 是 0-219, 不会因为后端矩阵造成饱和;

经过上述确认, 我们可以保证信号的完整, 不会出现饱和的现象, 然后再根据实际的需求调整 DLC 的参数即可。

Step3

彩色的调整

彩色调整因为和 PANEL 的相关性非常大，不同的 PANEL 彩色表现不同，而有些差异比较大，所以彩色的调整最好是在相同 PANEL 的基础上和样机对比。即便 panel 一样，如果在调整彩色之前没有将白平衡调好，色彩看上去也会千差万别。因此白平衡参数对彩色的影响也是至关重要的。

在 gamma 和白平衡都校准好后，我们才可以开始色彩的调整。

彩色应首先保证图像整体饱和度不要过低或过饱和，同时需要首先保证前端信号的饱和度：

(1)、CVBS 信号可调整 BK_COMB_75 (一般在 0x90-0xB0 之间)，BK_COMB_72 (一般用 0x80 或 0x90)，同时确认 BK_COMB_81、BK_COMB_82 为 0x96、0x6A，一般情况下请不要将前端 Comb 信号的饱和度调太大，过大的话会引起 cross color 比较严重的问题；

(2)、YPBPR 信号需要做 ADC 校正，请在 100% 的 color bar 信号下进行；VGA 信号也需要做 ADC 校正，请在黑白棋盘格信号下进行。

后端饱和度可调的寄存器有 C gain：即 BK_DLC_29，也就是说当此值为 0x40 时，表示乘以 1，这个值只有 6bit 可调。还有就是用户菜单的饱和度 (saturation) 的调整。色饱和度的调整最好也遵循从前往后逐级调整的原则。

在保证整体饱和度的前提下可以使用 MACE 提供的 FCC、ICC、IBC 和 IHC 实现对 Red、Green、Blue、Cyan、Magenta、Yellow、Flesh 七种颜色的独立调整而不影响其他的颜色；

FCC：对应寄存器在 BK_ACE_20；

以肤色校正为例说明 FCC 的调整原理，打开 BK_ACE_20[0]，首先我们需要确定 Cb 和 Cr 的 target，这个是我们调的 Cb 和 Cr 的目标值，对应 BK_ACE_30 和 BK_ACE_31，然后我们需要定义一个 range 来定义我们要校正的颜色范围，用 BK_ACE_48[1-0]、BK_ACE_48[3-2] 定义 Cr 向下和向上的 range，用 BK_ACE_48[5-4]、BK_ACE_48[7-6] 和定义 Cb 向下和向上的 range，定义以后调整 BK_ACE_40[3-0] 确定 strength；

以图为例：



ICC(Independent color control): enable 对应寄存器在 BK_ACE_60[6];
 ICC 的调整相对比较简单, BK_ACE_62[3-0]到 BK_ACE_68[3-0]即是对应 RGBCMYF 七种颜色的饱和度 gain; 一般我们推荐值为 0X08。
 参考如图:

■ **ICE: Independent color control**

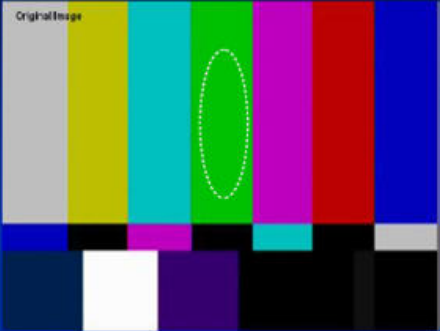
ICE means independent color enhancement, and you can adjust the difference color saturation for 7 different hues (RGBCMY)

Base Address	Bank	Register Index	Default Value	New Value	Description
0x2F	0x18	0x31	0x0001	0x6644	ICC saturation adjustment of R/G
0x2F	0x18	0x32	0x0002	0Xcc99	ICC saturation adjustment of B/C
0x2F	0x18	0x33	0x0003	0xc000	ICC saturation adjustment of M/Y
0x2F	0x18	0x34	0x0004	0x0088	ICC saturation adjustment of F

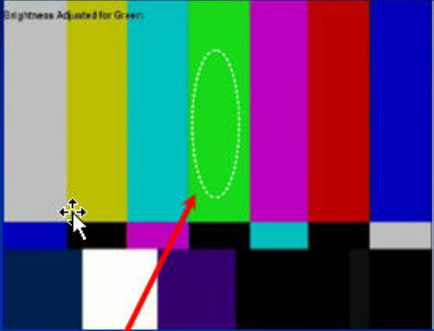
IBC(Independent brightness control): enable 对应寄存器在 BK_ACE_80[7];
 IBC 的调整和 ICC 类似, BK_ACE_82 到 BK_ACE_88 对应 RGBCMYF 七种颜色的亮度的调整, 我们一般推荐值为 0X20。
 如果值小于 0x20, 亮度相对原来的亮度是减小, 值越小亮度越低;
 如果值大于 0x20, 亮度相对原来的亮度是增加, 值越大亮度越高;
 参考如图:

■ IBC: Independent Brightness Control

Original Image



Brightness Adjusted for Green

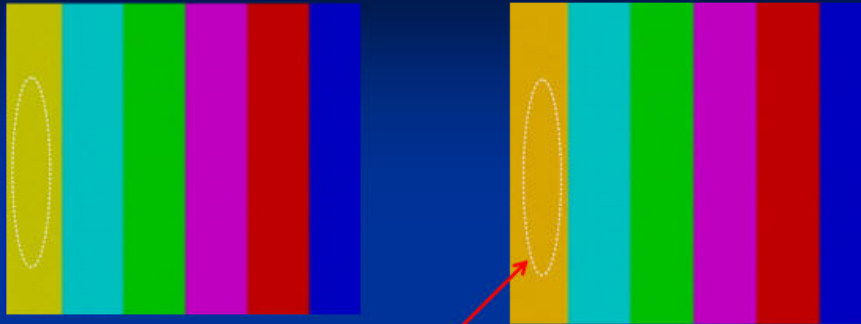


Increase green brightness independent

0x2F	0x18	0x41	0x2020	0x2010	IBC Y adjustment of R/G
0x2F	0x18	0x42	0x2020	0x2020	IBC Y adjustment of B/C
0x2F	0x18	0x43	0x2020	0x2018	IBC Y adjustment of M/Y
0x2F	0x18	0x44	0x2020	0x0018	IBC Y adjustment of F

IHC(Independent hue control): enable 对应寄存器在 BK_ACE_C0[7];
 IHC 的调整, BK_ACE_C2 到 BK_ACE_C8 对应 RGBCMYF 七种颜色的色调的调整, 这些值的 BIT6 是 hue 的方向, 而 BIT[5-0]是 hue 的值, 值越大表示偏离原来越大;
 参考如图:

■ IHC: Independent Hue Control



Adjust yellow hue independent

0x2F	0x18	0x61	0x0000	0x0038	IHC hue adjustment of R/G
0x2F	0x18	0x62	0x0000	0x3838	IHC hue adjustment of B/C
0x2F	0x18	0x63	0x0000	0x2000	IHC hue adjustment of M/Y
0x2F	0x18	0x64	0x0000	0x0058	IHC hue adjustment of F

具体到 debug tool 中的各个寄存器位置如下图:

■ 在DEBUG tool的位置如下:

The debug tool interface shows the following registers and their values (hexadecimal):

Address	Value
00	18
01	00
02	00
03	00
04	00
05	00
06	00
07	00
08	00
09	00
0A	00
0B	00
0C	00
0D	00
0E	00
0F	00
10	00
11	00
12	00
13	00
14	00
15	00
16	00
17	00
18	00
19	00
1A	00
1B	00
1C	00
1D	00
1E	00
1F	00
20	15
21	00
22	00
23	00
24	00
25	00
26	00
27	00
28	00
29	00
2A	00
2B	00
2C	00
2D	00
2E	00
2F	00
30	73
31	9B
32	6E
33	AC
34	A5
35	69
36	5A
37	9A
38	5A
39	75
3A	B0
3B	48
3C	74
3D	C8
3E	80
3F	80
40	85
41	88
42	88
43	88
44	05
45	00
46	00
47	00
48	A6
49	BA
4A	FB
4B	73
4C	FB
4D	FB
4E	FF
4F	AA
50	24
51	00
52	00
53	00
54	00
55	00
56	00
57	00
58	00
59	00
5A	00
5B	00
5C	00
5D	00
5E	00
5F	00
60	C0
61	00
62	44
63	66
64	99
65	CC
66	CC
67	CC
68	88
69	00
6A	00
6B	00
6C	00
6D	00
6E	00
6F	00
70	42
71	00
72	20
73	0E
74	C0
75	01
76	3C
77	02
78	FF
79	00
7A	00
7B	00
7C	00
7D	00
7E	00
7F	00
80	A0
81	00
82	18
83	10
84	18
85	18
86	18
87	24
88	00
89	20
8A	20
8B	20
8C	20
8D	20
8E	20
8F	20
90	00
91	88
92	00
93	00
94	00
95	00
96	00
97	00
98	00
99	00
9A	00
9B	00
9C	00
9D	00
9E	00
9F	00
A0	00
A1	33
A2	CF
A3	69
A4	24
A5	01
A6	CF
A7	69
A8	24
A9	01
AA	81
AB	C1
AC	11
AD	11
AE	00
AF	00
B0	00
B1	00
B2	00
B3	00
B4	00
B5	00
B6	00
B7	00
B8	00
B9	00
BA	00
BB	00
BC	00
BD	00
BE	00
BF	00
C0	80
C1	00
C2	70
C3	00
C4	60
C5	00
C6	20
C7	00
C8	00
C9	00
CA	00
CB	00
CC	00
CD	00
CE	00
CF	00
D0	00
D1	00
D2	00
D3	00
D4	00
D5	00
D6	00
D7	00
D8	00
D9	00
DA	00
DB	00
DC	00
DD	00
DE	00
DF	00
E0	00
E1	00
E2	00
E3	00
E4	00
E5	00
E6	00
E7	00
E8	00
E9	00
EA	00
EB	00
EC	00
ED	00
EE	00
EF	00
F0	00
F1	00
F2	00
F3	00
F4	00
F5	00
F6	00
F7	00
F8	00
F9	00
FA	00
FB	00
FC	00
FD	00
FE	00
FF	00

具体调试可以参考以下图谱来判断所调颜色是否合适:

1, 先将 3*3 color matrix 的 saturation default 为 128, ACE 中的 ICC 和 IBC 以及 IHCdefault

为中间值，用一个 11 阶 R, G, B 的色阶将此三基色的色饱和度调好，可以在下面图谱中，分别用红，绿，蓝三色滤光片，以看到旁边的亮度跟红色部分-5%---5%亮度接近为原则；



2，如果看主观效果，建议特别注意看下如下图谱：

红色调的是否过分看看下面图片中的毛衣是否能分清细节：



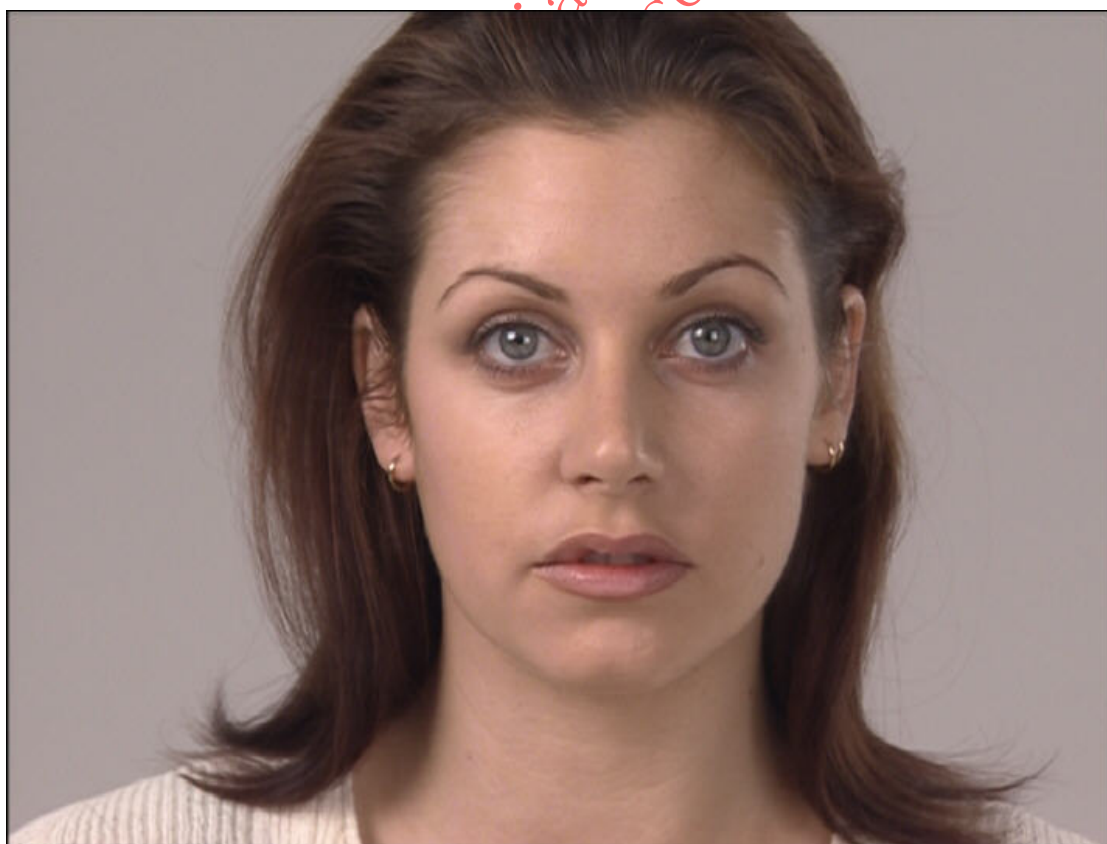
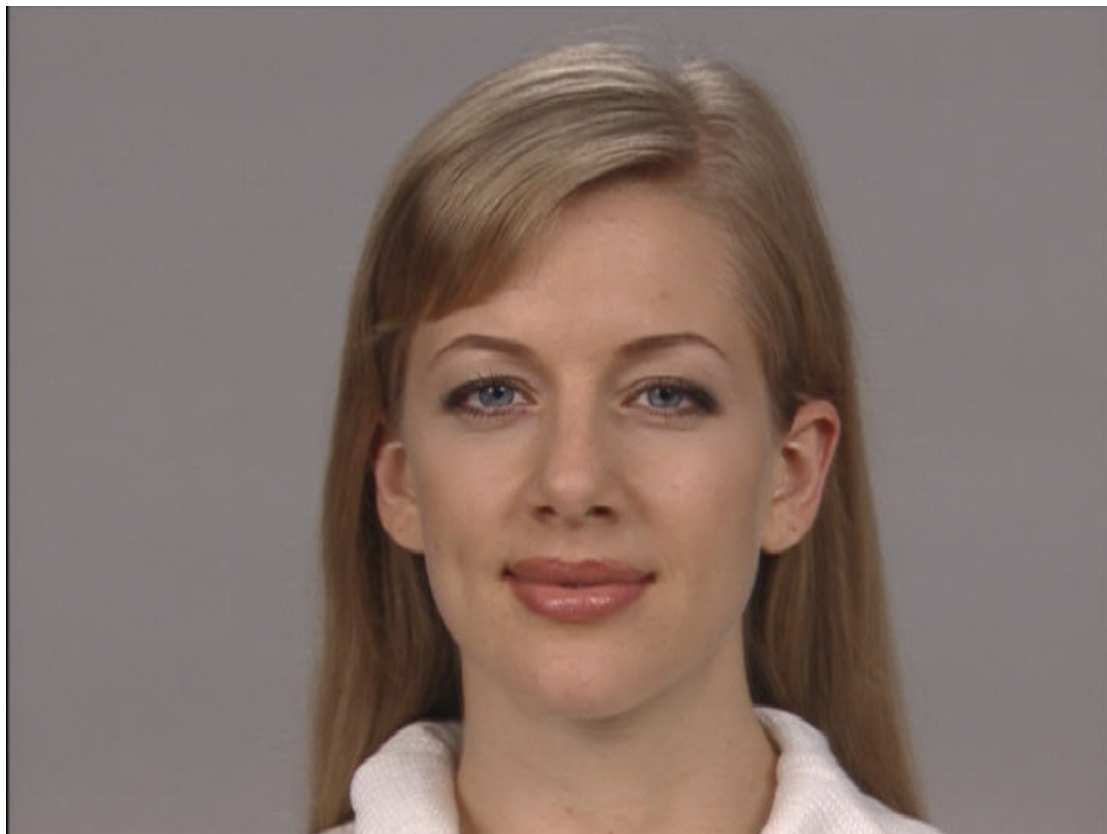
绿色部分调的是否过分请看下面绿色毛衣是否能分清楚细节：



蓝色部分调的是否过分请看下面绿色毛衣是否能分清楚细节：



肤色部分主要 check 以下几幅图谱：





• 21 e







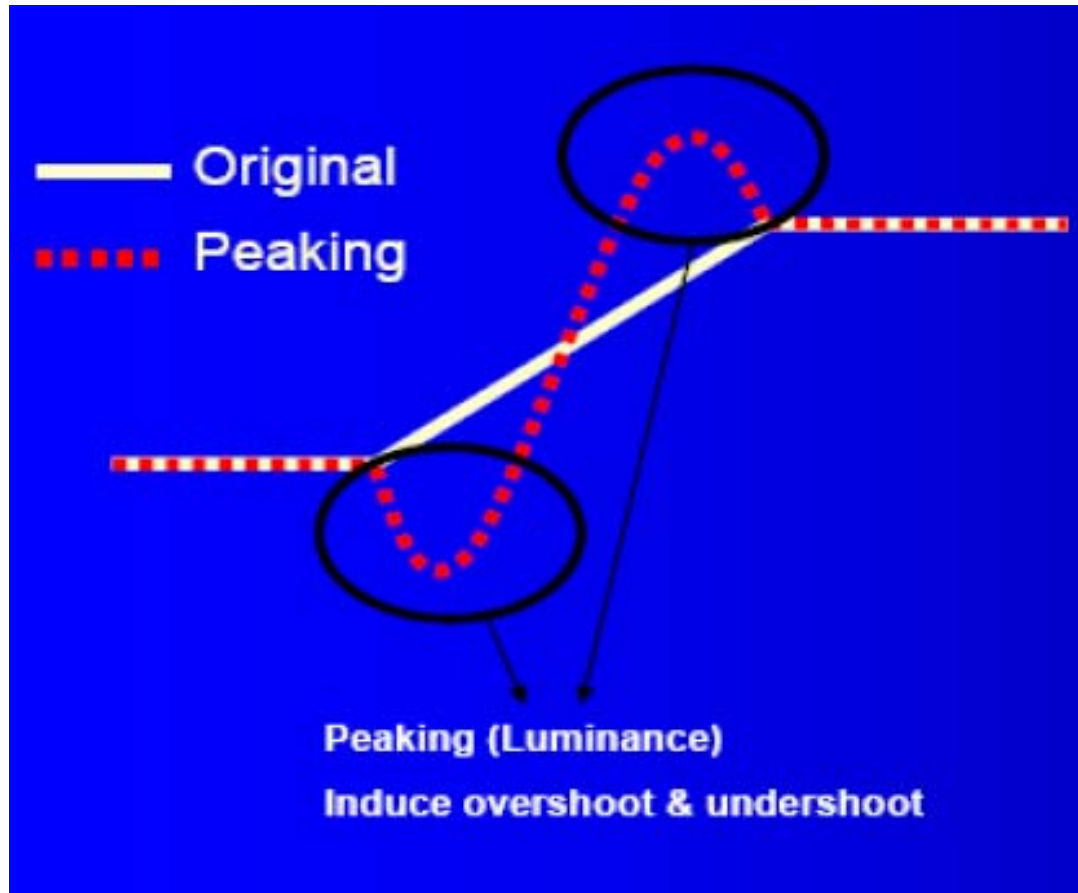
最后申明一点，色彩调整有很大的主观性，具体要调成何种程度，并没有一个很明确的标准，原则只有一个，以客户看着舒服为首要准则!!

MSTAR Confidential
internal use

Step4

清晰度的调整

首先说明一下 Peaking 调整的原理：

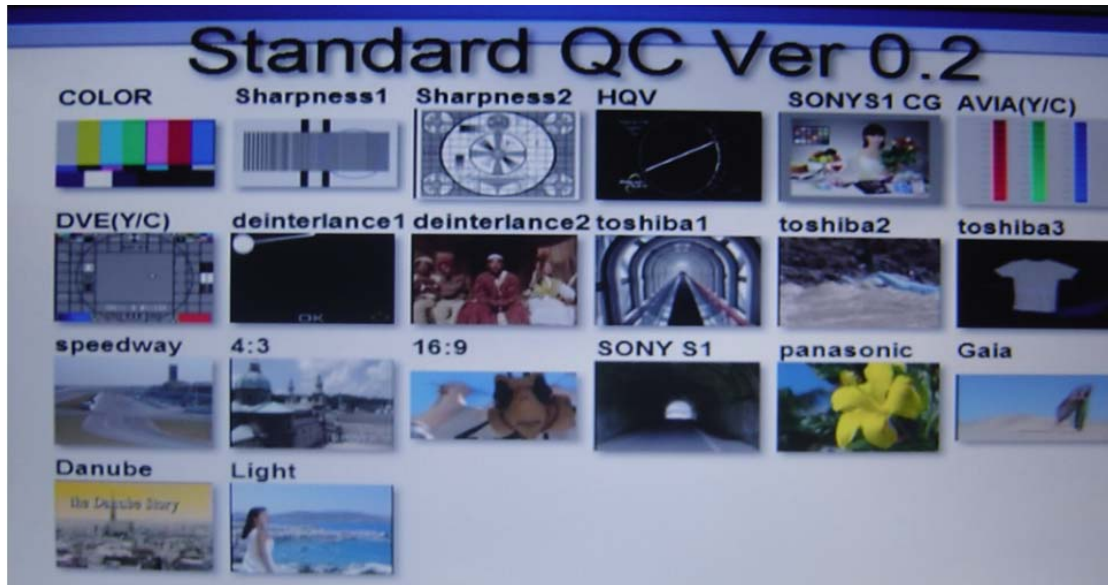


Peaking 是将原始信号变的更加陡峭，从而使整个图像更 Sharp；

Peaking ON (BK19_20[0]=1)：

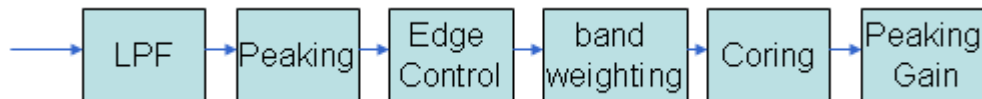


Peaking OFF (BK19_20[0]=0):



BK19_20[0]是 Peaking Block 的开关, 可以做 debug mode 来 CHECK 问题;

Peaking 部分的处理流程如下:



1、Peaking Block 首先会经过一个 Low Pass Filter, 这个 LPF 分 horizontal 和 Vertical 两个方向:

(1) 水平方向的寄存器位置在 BK19_20[6-4], 设置为 0x00 可将 LPF 关掉, 而值越大代表 LPF 高频越强, 图像高频成分会越多;

(2) Vertical 方向寄存器位置在 BK19_25, 低 4 位和高 4 位的设置需要相同, 而值越大代表 LPF 高频越强, 图像高频成分会越多;

我们较常用的值一般在 3-6 之间;

2、Peaking 的调整是以 pixel 为单位分不同的 band 来处理, 目前大部分的 IC 共有 12 个 band, 而一些 IC 只有 4 个 band (只包含水平方向的 4 个 band):

(1)、水平方向最多有 8 个 band 的 peaking 调整, 我们一般使用 band1-4 就可以满足要求, 对应寄存器 BK19_30、BK19_31、BK19_32、BK19_33, band9-12 对应寄存器 BK19_D0、BK19_D1、BK19_D2、BK19_D3;

(2)、垂直方向有 2 个 band 的 peaking 调整, 对应寄存器 BK19_34、BK19_37;

(3)、diagonal 方向有 2 个 band 的 peaking 调整, 对应寄存器 BK19_35、BK19_36;

以上寄存器都有 6 个 bit 位;

同时我们还提供每个 band Peaking 的 step 调整, 最终的

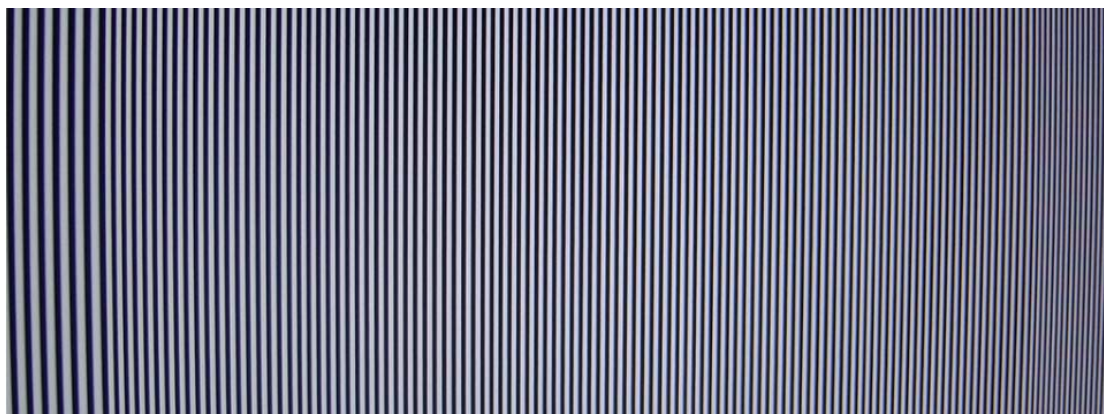
peaking gain=band gain x step,

对应的寄存器:

Band1-8: BK19_22-25;

Band9-12: BK19_1C;

以上寄存器每个 band 对应 2 个 bit;



我们可以使用 Multi-Burst 来初步确认各 band 的 Peaking 值，保证各频率段的频响足够；垂直方向的 Peaking 调整也可用 Multi-Burst 来确认或者采用一些有字符的画面来确定垂直方向较低频的信号，同时应该注意 Edge Control 的设置也会影响到 Multi-Burst 的表现，所以应该注意需要再 Edge Control 不要设置太低的情况下确认；在调整的时候先用各 band 的 peaking gain 去调整，没办法达到预期再去调整 Peaking step，因为是相乘的关系所以会提高很多，变化较大；

3、Edge Control:

在做 Peaking 的过程中，会将信号本身的 ring 加强，如果 Peaking 设置过强同时可能会带来 ring，我们可以用 Edge 控制来降低，针对这些 overshoot 和 undershoot 我们也是根据不同的 band 来做调整，对应 band1-12 的寄存器是：

Band1-8 的 overshoot 对应 BK19_40 到 BK19_47；

Band9-12 的 overshoot 对应 BK19_E0 到 BK19_E3；

Band1-8 的 undershoot 对应 BK19_48 到 BK19_4F；

Band9-12 的 undershoot 对应 BK19_E4 到 BK19_E7；

这些寄存器都是 8bit 的，设置为 0xFF 为最大，就是不去做 edge control 的处理，设置为 0x00，这个 band 会因为 overshoot 或 undershoot 全被限制而无法实现 Peaking 的增强，相当于将这个 band peaking 关掉。

具体如图所示：

在未调整 overshoot/undershoot 之前：



调整 overshoot/undershoot 之后:



Edge Control 对处理中造成的 ring 效果还是比较明显，但是如果信号本身有比较明显的 ring，一般需要用 Peaking 和 Edge Control 配合起来降低，但是不建议为此而降低过度的清晰度；

4、band weighting: 不建议调整；

5、Coring:

做 peaking 的时候整个画面都会被提升, 那么 noise 也会被放大, 我们可以通过调整 Coring 来实现;

Coring 是将 Peaking 做后的 pixel 的值和原本 pixel 的值相比, 小于 Coring value 的时候会代表这段区域为平坦区, 不做 sharpness, 以免放大 Noise;

寄存器位置在 BK19_26, 低 4 位是 threshold 1, 而高 4 位是 threshold 2, 小于 threshold 1 不做 Peaking, 而大于 threshold 2 才做 Peaking, 所以这个寄存器设置是高 4 位 > 低 4 位, 而值越大代表做 coring 的越多而做 Peaking 越少, noise 会越少;

因为小的 noise 和画面中一些小的细节在信号表现上是差不多的, 所以如果 Coring 太大在降低一些 noise 的同时有可能会损伤到细节, 一般我们不会大于 0x43;

同时还可以分不同的 band 来调整不同的 coring, 对应寄存器:

Band1-8: BK19_66 到 BK19_69;

Band9-12: BK19_78 和 BK19_79;

设置方法和上述相同而值越大代表做 coring 的越多而做 Peaking 越少, noise 越小;

这个值的设置一般最大不要大于 4;

另外还有 Coring step 可供调整, 上述两个寄存器最后都会乘上 step 的系数, 对应寄存器为 BK19_61 的 bit5 和 bit4, 显然这个系数越大, 那么 coring 的值就会越大, noise 就会越小, 一般我们设置不会大于 2;

Coring 效果如下图所示:

没加 Coring 调整之前:



Coring 调整之后:



从上图可以明显看到 Coring 对 noise 的影响还是很明显, 所以可以根据实际情况设置适当的值以解决 noise 的问题, 但是也可以明显看到, 如果设置不当对细节的影响也是很大, 所以设置完 Coring 后需要对细节进行 CHECK;

6、Peaking Gain: 对应寄存器 BK19_27;

这个寄存器来控制这个画面的清晰度, 也是 MENU 里控制的寄存器, 用这个寄存器来控制前面送进来信号 sharpness 的 gain;

可建议软件按照不同的 source 来控制这个寄存器;

下面以实际画面为例说明相关问题的 CHECK 方法和清晰度的设置:

草地细节不好:



面包细节少模糊:



从上述 Peaking 调节的流程和方法上看，图像丢失的地方都是一些细节的地方，细节的地方都是相对频率较高，所以我们可以首先 CHECK Peaking LPF 是否设置的不合适，将 BK19_20[6-4] 设置为 0、5、6 等，BK19_25 设置为 0x77 或 0x88 看一下，另外 Coring 在减小 Noise 的同时也会对图像细节有影响，更改 coring 相关的寄存器，将 BK19_26 设置为 0x21 或者 0x31，Coring step BK19_61 的 bit5 和 bit4 设置为 0，将 band Coring BK19_66 到 BK19_69 也设置为 0x00，这时候我们应该可以看到细节都出来了：





另外还要注意一些因为 scaling 造成清晰度方面的损失,当 PANEL 的分辨率小于信号分辨率或者为解决 BW 问题的时候我们一般会采用 Pre-Scaling down 的做法,这时前端 scaler 会有一个 filter table 可选,目前我们 Qmap 会有一些 table 可选择,我们可以先将 BK2_0B[7] Pre-Scaling down 关掉,如果关掉后细节增多,可在 table 表中选择

LPF40Ap0As40	LPF50Ap0As40	LPF60Ap0As35	LPF70Ap0As35	LPF80Ap0As35	LPF90Ap0As35
--------------	--------------	--------------	--------------	--------------	--------------

从左向右画面高频会越强;

FIR40	FIR45	FIR50	FIR55	FIR60	FIR66	FIR75	FIR80	FIR85
-------	-------	-------	-------	-------	-------	-------	-------	-------

从左向右画面高频会越强;

下面介绍一下 CTI (Color Transient Improvement) 的原理和调试方法:

在传输过程中,因为系统的瞬态响应的问题,信号不可避免的会有损伤,在彩色信号方面,我们最常见的就是彩色交接的地方会变模糊,而 CTI 就是为校正这种情况,将已变换较缓的信号变得更加陡峭,使彩色交接变得更加锐利;

目前有的芯片有两个 CTI,一个在 scaler 之前,一个在 scaler 之后,而有的芯片只有 scaler 之前的那个;

PRE-CTI 对应的寄存器如下:

BK2_5B[7]: CTI ON/OFF;

BK2_58[2-0]: CTI Low Pass Filter;

BK2_58[6-4]: CTI step;

BK2_59[5-0]: CTI band pass filter;

配合 CTI step 和 band pass filter 可以实现 CTI 强度的调整,两个设定越强 CTI 效果会越明显,但是也没必要设置过强,设置过强的话彩条交接的地方会有较明显的白边;

POST-CTI 所对应的寄存器:

BK27_40[0]: CTI ON/OFF;

BK27_42[2-0]: CTI Low Pass Filter;

BK27_42[6-4]: CTI step;

BK27_44[5-0]: CTI band pass Filter;

调试方法和 PRE-CTI 的相同

如果输入是 CVBS 信号, 在 Comb 里也有一个 CTI:

BK36_83[5-4]: CTI MODE 00:off, 01:weak, 02:normal, 03:strong;

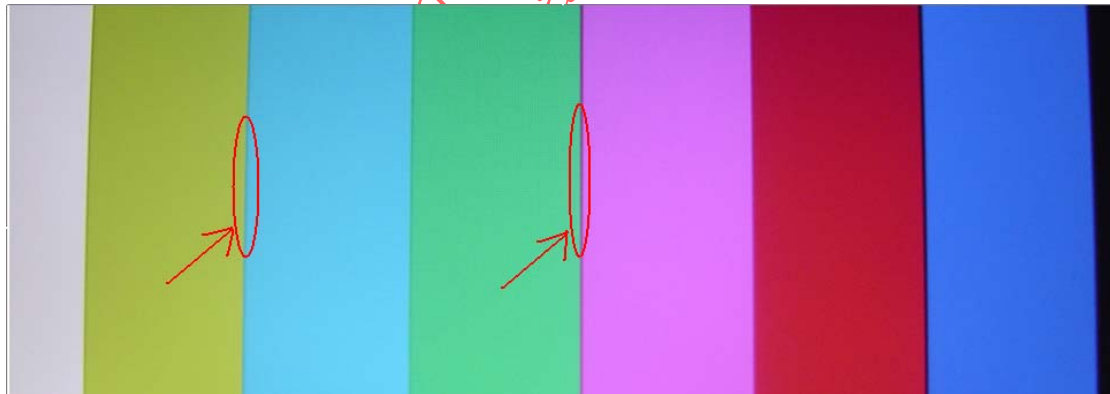
这个寄存器会影响到 C 的相位, 调试的时候注意;

效果如下图所示:

CTI OFF:



CTI ON:



正如下图所示, 彩色交接变化明显, CTI 能使交接变得更加锐利;

Y/C delay 的原理和调试方法:

在信号传输过程中, 不同频率的信号会产生不同的延时特性, 一般来讲信号频率越高延时越小, 而我们有些画面亮度信号和彩色信号的频率相差较大的情况下就会出现 YC 无法完全重叠在一起的现象, YC delay 可以校正这种情况;

目前有的芯片有两个 YC delay, 一个在 scaler 之前, 一个在 scaler 之后, 而有的芯片只有 scaler 之前的那个;

PRE-YC delay 对应的寄存器如下:

BK2_14[3-0]: Y/C delay pipe step;

BK2_14[4]: Y delay enable;

用 Y/C delay pipe step 来调整 YC 中间的相位关系,

用 bit4 来控制将 YC 相位的方向, 是 Y 向前还是 C 向前, bit4=0 的话是 Y 向前, 而 bit4=1 是 C 向前, 所以如果 Y 在 C 的后面可将 bit4=0 而如果 Y 在 C 的前面可将 bit4=1;

POST-YC delay 对应的寄存器如下:

BK19_76[0]: Y/C delay Enable;

BK19_77[1-0]: Y Ahead Number, BK19_77[3-2]: C Ahead Number;

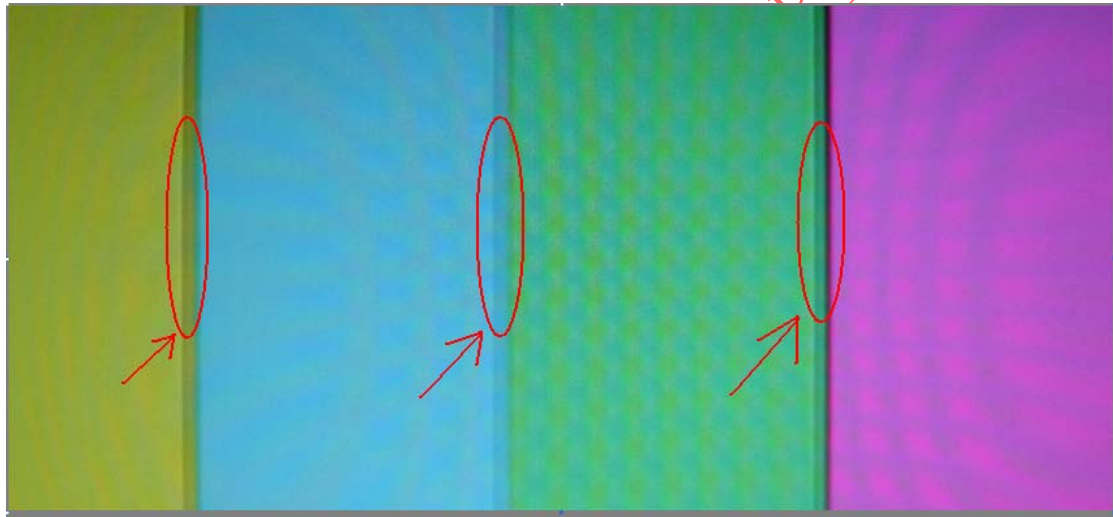
可以用 Ahead Number 来分别控制 Y 和 C 的相位;

如果输入是 CVBS 信号, 在 Comb 里也有一个 YC delay:

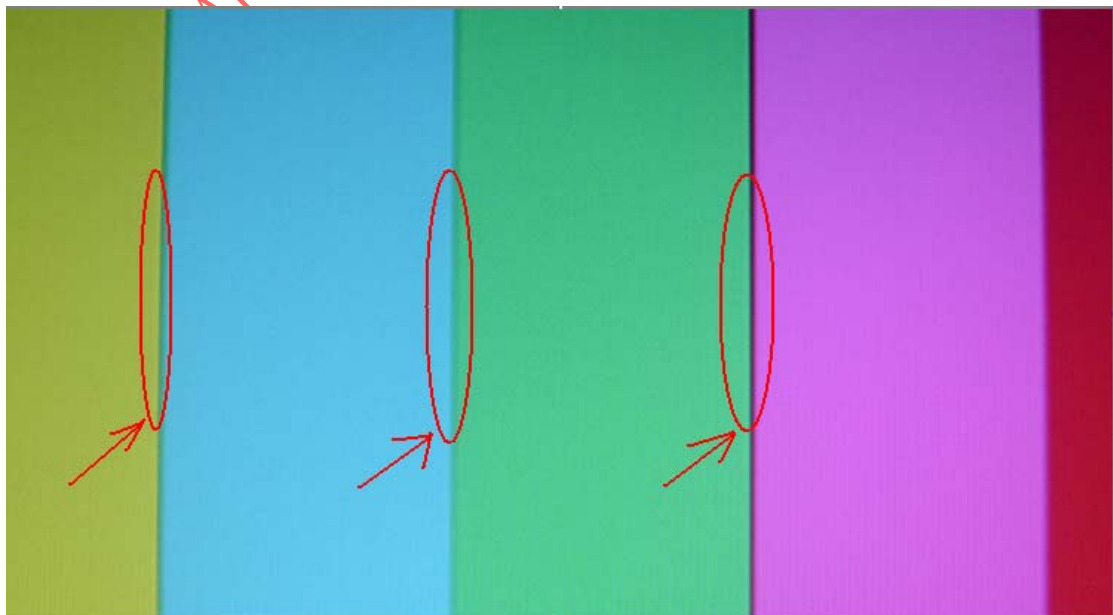
BK36_83[3-2]: 这个寄存器只调整 Y 的相位, 且方向只能向后;

如图所示:

YC 相位有差异:



调整后:

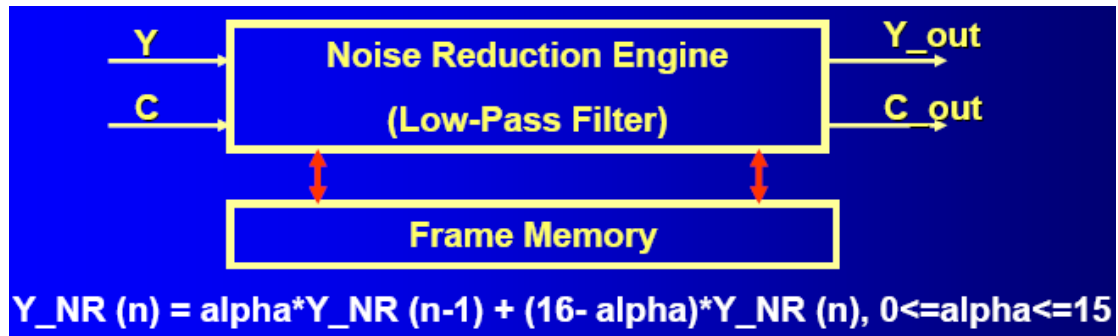


Step5

NR 的调整

常调的有 DNR、SNR

1、DNR



DNR 是唯一与时间有关的 NR，即常说的 3D NR。

$$Y'' = (1 - \alpha) * Y(n) + \alpha * Y(n-1)$$

$Y(n)$, 是指当前画面; $Y(n-1)$, 是指前场画面。

α 越大, $Y(n-1)$ 的权重越大,

α 是根据两帧画面的差异 diff, 查 NR table 得来的。

DNR 强弱可调整 Y table/ C table, 常用 value 如下表, 具体可在 low 和 high 的值间按需调整。设置越大降噪越强, 按照前面由大到小减小。

low	0x56	0x34	0x12	0x00	0x00	0x00	0x00	0x00
mid	0x78	0x56	0x34	0x12	0x00	0x00	0x00	0x00
high	0xCD	0xAB	0x89	0x67	0x45	0x23	0x01	0x00

DNR, 可以把随机运动的 Noise 消掉、变慢、定住, 可用 BOOK1 的寺庙画面检验:



Side effect: 如果 α 设置太大, 会产生拖影。可以用 BOOK1 的鹦鹉画面 check, 看鹦鹉头摆动时是否产生拖影。

OK



NG



当然，拖影成因很多，比如 Content 和 Panel,需要先理清原因: 关掉 DNR, 比较前后.(若疑惑哪个 function 导致 side effect, 开关它, 比较前后, 就可理清)

2、SNR:Spatial Noise Reduction

除了DNR, 其他NR 都是 spatial domain NR, 即在同一场画面中做NR, 但是有的会参考Motion 值来做设置;

(1) 根据画面平坦度来做的 SNR;

Before



After



开关: BKC_60[0];

BKC_61 [7:0], 设成 0xFF 代表完全不参考 Edge, 所有的画面作为 flat area, 都做 SNR, 值越小越会参考 edge, 发现 Edge 就不做;

同时我们可以根据不同的 Motion ratio 设置不同的强度, 开关在 BKC_60[1];

Bank	Addr	Bit	Default	Note
C	h0060	0		SNR enable F2
C	h0060	1		SNR motion ratio enable F2
C	h0060	[15:8]		SNR active threshold(active area)
C	h0062	[3:0]		SNR strength F2
C	h0062	[7:5]		SNR alpha step
C	h0068	[7:0]		SNR LUT_01
C	h0068	[15:8]		SNR LUT_23
C	h006A	[7:0]		SNR LUT_45
C	h006A	[15:8]		SNR LUT_67
C	h006C	[7:0]		SNR LUT_89
C	h006C	[15:8]		SNR LUT_AB
C	h006E	[7:0]		SNR LUT_CD
C	h006E	[15:8]		SNR LUT_EF

(2) 另外的 SNR:

开关: BK18_AA[7]

BK18_AB[7]

强度设置: BK18_AA[5:0]

BK18_AB[5:0]

一般可设 BK18_AA=C2 or C3, 最大可设置为 C4;