Chapter 27 Visual Output Processor (VOP)

27.1 Overview

VOP is the display interface from memory frame buffer to display device (LCD panel, LVDS, MIPI, eDP, HDMI and TV set). VOP is connected to an AHB bus through an AHB slave and AXI bus through an AXI master. The register setting is configured through the AHB slave interface and the display frame data is read through the AXI master interface. Furthermore, there is a data path between IEP and VOP, which can provide frame data from IEP to VOP.

27.1.1 Features

- Display interface
  - Parallel RGB LCD Interface
    - RGB101010,RGB888,RGB666,RGB565
  - Serial RGB LCD Interface
    - 2x12-bit,3x8-bit(RGB delta supported),3x8-bit+dummy
  - Parallel MCU LCD Interface
    - 24-bit(RGB888),18-bit(RGB666),16-bit(RGB565)
    - hold/auto/bypass mode
  - Serial MCU LCD Interface
    - 2x12-bit, 3x8-bit with hold mode
  - TV Interface
    - ITU-R BT.656(8-bit, 480i/576i/1080i)
    - 3 output mode: valid data in lower 8bit, middle 8bit and higher 8bit
    - TV encoder
  - Support SDR(single data rate) interface
  - Support DDR(dual data rate) interface for LVDS/PARALLEL RGB
    - parallel RGB and 2x12-bit serial RGB
    - Single or dual clock out
  - Max output resolution
    - VOP_BIG: 3840x2160
    - VOP_LITTLE: 2560x1600
  - Scaning timing 8192x4096
  - Support configurable polarity of DCLK/HSYNC/VSYNC/DEN
  - 4 groups of scanning output for PARALLEL RGB,LVDS,LVDS,HDMI,eDP
  - MIPI control
    - MIPI dual channel,overlay scan(overlapped pixels=2~16 pix)
    - MIPI flow control(edpihalt)
    - MIPI DCS command mode

- Display process
  - CABC
  - BCSH,8bit
    - Brightness,Contrast,Saturation,Hue adjustment
    - YUV-10bit，RGB-10bit
  - Dither down
    - pre dither down for RGB-10bit to RGB-8bit
    - allegro for RGB565 and RGB666
    - FRC with configurable pattern for RGB666
  - Gamma
    - LUT(lookup table) for R/G/B respectively
    - 8bit/10bit RGB look up table
    - gamma after dither

- Support display data swap
  - BG swap, RB swap, RG swap, dummy swap, delta swap
- Support three YUV2RGB transition modes:
  - 8bit-YUV: rec601-mpeg/rec601-jpeg/rec709
  - 10bit-YUV: BT2020
- blank display
- black display
- standby mode
- auto dynamic power control
- X-MIRROR,Y-MIRROR for win0/win1/win2/win3/hwc?
- scale down for TV overscan
  - after overlay
  - arbitrary non-integer scaling ratio
  - horizontal scale down using bilinear, 0.5~1.0
  - vertical scale down using bilinear, 0.5~1.0

- Layer process
  - Background layer
    - programmable 24-bit color
  - Win0/Win1 layer
    - Support data format
      - RGB888, ARGB888, RGB565,
      - YCbCr420SP, YCbCr422SP, YCbCr444SP
      - YUV-8bit,YUV-10bit
    - YUV clip
      - Y-10bit:64~940;UV-10bit：64~960
      - Y-8bit: 16~235;UV-8bit: 16~240
    - Support max input resolution 4096x8192
    - Support max output resolution 3840x2160
    - Support virtual display
    - Support 1/8 to 8 scaling-down and scaling-up engine
      - scale up using bicubic and bilinear
        - Arbitrary non-integer scaling ratio
        - 4 bicubic table for scale up using precise,spline,catrom,Mitchell
      - scale down using bilinear and average
        - Arbitrary non-integer scaling ratio
      - per-pix alpha + scale
    - Support data swap
      - RGB/BPP: alpha_swap,rb_swap
      - YUV: mid_swap,uv_swap
    - transparency color key,prior to alpha blending and fading
    - Support fading,prior to alpha blending
    - Support alpha blending
    - Support interlace and de-flicker for interlace output
    - Support IEP direct path input
  - Win2/Win3 layer
    - Support data format
      - RGB888, ARGB888, RGB565
      - 1BPP,2BPP,4BPP,8BPP
      - little endian and big endian for BPP
      - BYPASS and LUT mode(25bit LUT，1bit AA+8bit-RGB) for BPP
    - 4 display regions
      - only one region at one scanning line
    - Support data swap
      - RGB/BPP:rb_swap,alpha_swap
    - Support transparency color key,prior to alpha blending and fading
    - Support fading,prior to alpha blending
    - Support alpha blending

- Support interlace read and interlace output
- Support IEP direct path input

- Hardware Cursor layer(HWC for short)
  - Support data format
    - RGB888, ARGB888, RGB565
    - 1BPP,2BPP,4BPP,8BPP
    - little endian and big endian for BPP
    - BYPASS and LUT mode(25bit LUT，1bit AA+8bit-RGB)for BPP
  - Support four hwc size: 32x32,64x64,96x96,128x128
  - Support 2 color modes: normal and reversed color
  - Support fading,prior to alpha blending
  - Support alpha blending
  - Support displaying out of panel,right or bottom
  - Support NORMAL color and REVERSE color mode
  - Support interlace read and interlace output
- Overlay
  - Support 6 layers,background/win0/win1/win2/win3/hwc
  - Win0/Win1/Win2/Win3 overlay position exchangeable
  - Alpha blending
    - Support 12 alpha blending modes
    - Support pre-multiplied alpha
    - Support global alpha and per_pix alpha
    - Support 256 level alpha
    - layer1/layer2/layer3/hwc support alpha

- Bus interface
  - Support AMBA 2.0 AHB slave interface for accessing internal registers and LUT memories，32bit data bus width
  - Support AMBA 3.0 AXI master read interface for loading frame data
    - 128bit data bus width
  - Support MMU
  - Support two transfer modes
    - auto outstanding transfer
    - configuruable outstanding transfer(gather transfer)
  - DMA line mode for YUV
  - Support QoS request for higher bus priority for win2/win3
  - Support NOC hurry for higher bus prioirity for win0/win1
  - Support DMA stop mode
  - max read outstanding number
    - 32 when MMU disable
    - 31 when MMU enable

- Interrupt
  - One combined interrupt
    - high active
    - raw status
    - combinational with 12 interrupt sources
      - frame start interrupt
      - line flag interrupt
      - bus error interrupt
      - win0 empty interrupt
      - win1 empty interrupt
      - win2 empty interrupt
      - win3 empty interrupt
      - hwc empty interrupt
      - post empty interrupt
      - pwm gen interrupt

✧ irq_mmu

27.2 Block Diagram

The architecture is shown in the following figure.

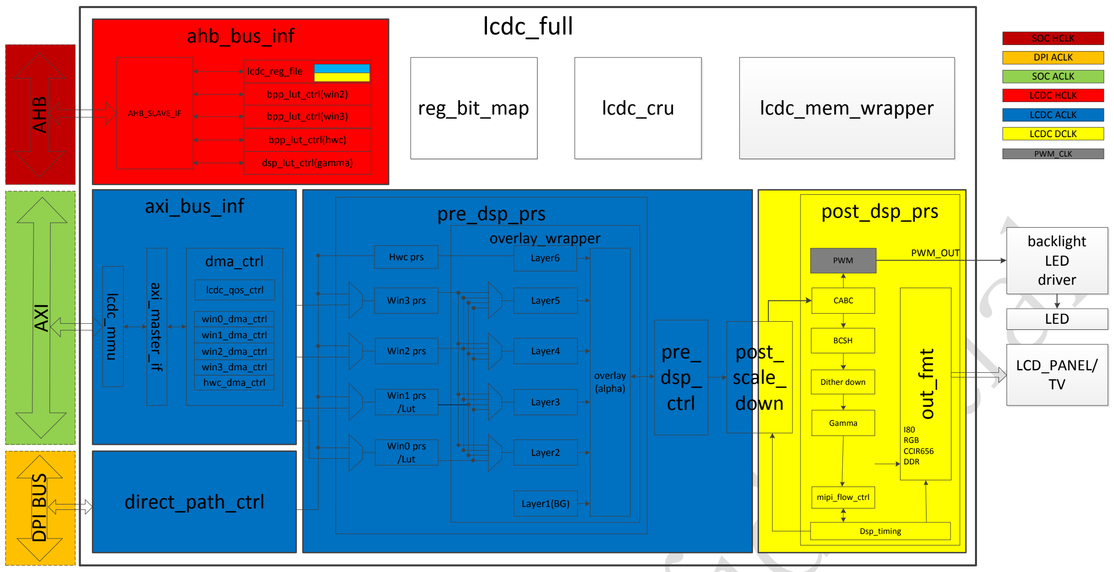


Fig. 27-1 VOP Block Diagram

27.3 Function Description

27.3.1 Pixel format

1.RGB

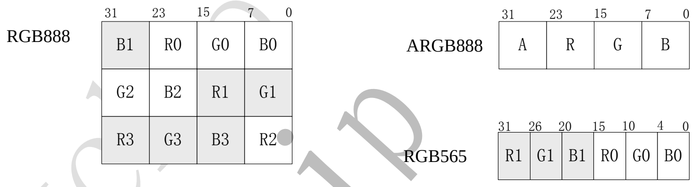


Fig. 27-2 RGB data format

2.YCbCr/YUV(8bit/10bit)

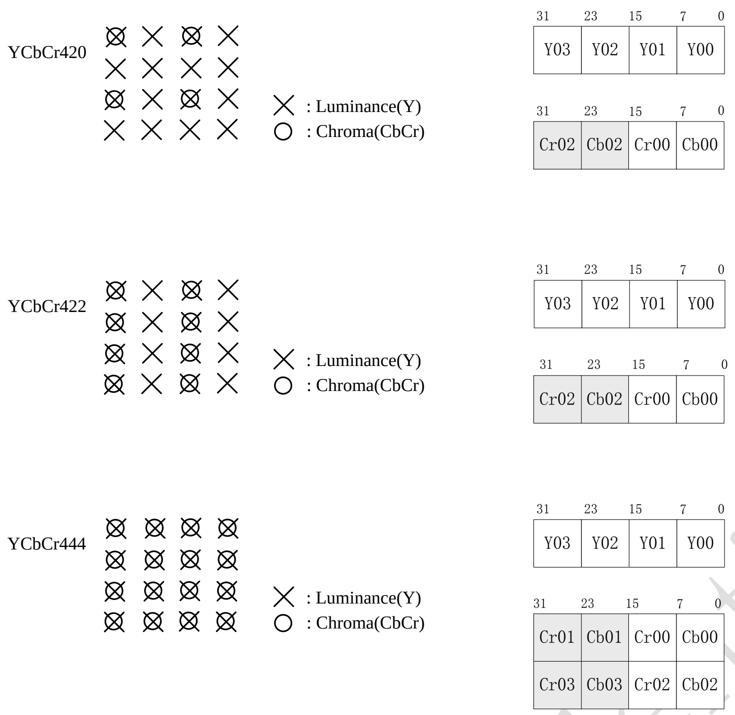


Fig. 27-3 YUV data format

YUV just support SP
YUV-8bit 32bit align
YUV-10bit 128bit align

3.BPP

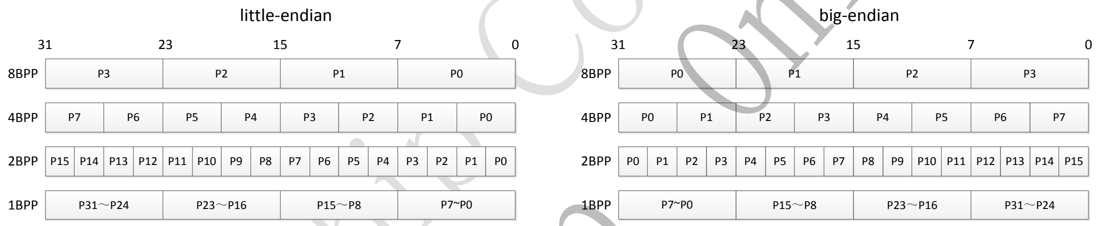

little-endian
31 23 15 7 0
8BPP P3 P2 P1 P0
4BPP P7 P6 P5 P4 P3 P2 P1 P0
2BPP P15 P14 P13 P12 P11 P10 P9 P8 P7 P6 P5 P4 P3 P2 P1 P0
1BPP P31～P24 P23～P16 P15～P8 P7~P0
big-endian
31 23 15 7 0
8BPP P0 P1 P2 P3
4BPP P0 P1 P2 P3 P4 P5 P6 P7
2BPP P0 P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 P11 P12 P13 P14 P15
1BPP P7~P0 P15～P8 P23～P16 P31～P24

Fig. 27-4 BPP little/big endian data format

27.3.2 Pixel Data Path

There are two data input path for VOP to get display layers' pixel data. One is internal DMA; the other is direction path interface.

1.Internal DMA

Internal DMA can fetch the pixel data through AXI bus from system memory (DDR) for all the display layers. Data fetching is driven by display output requirement.

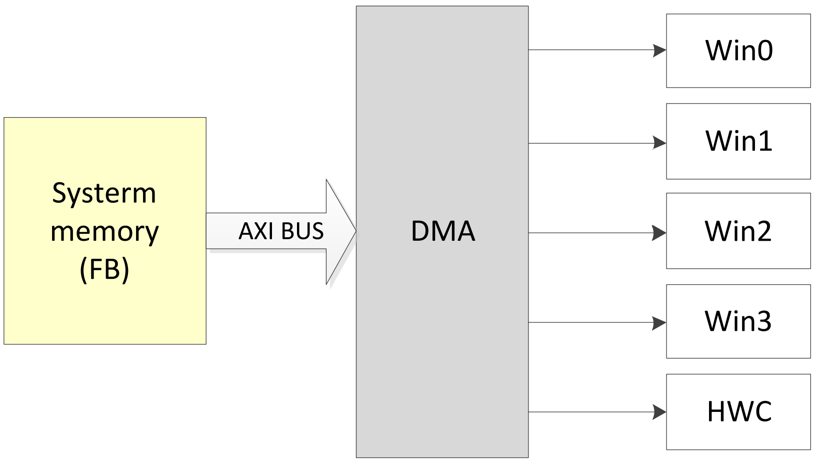


Fig. 27-5 LCDC Internal DMA

2.Direct Path Interface

Direct path interface (DPI) is used for direct image display of external image processing IP. There is a local bus between VOP and external image processing IP for the data transfer.

DPI is connected to WIN0/WIN1/WIN2/WIN3 but can only be configured for One layer use (Win0 or Win1 or Win2 or Win3) in each frame.

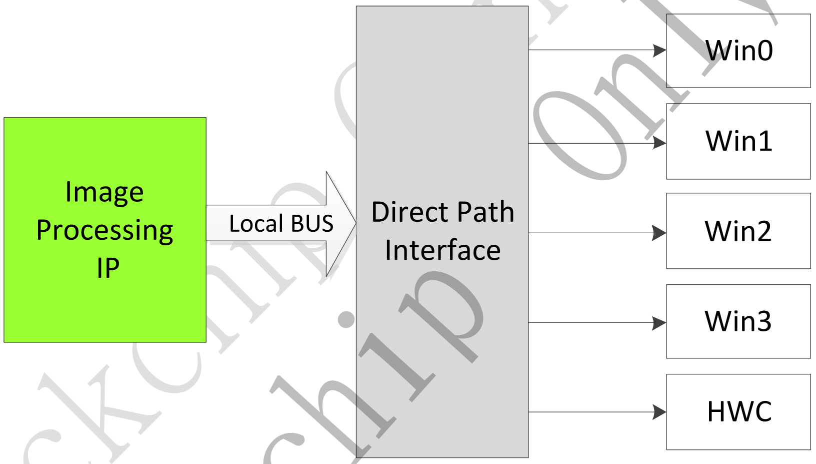


Fig. 27-6 VOP Direct Path Interface

27.3.3 Win Scaling

The scaling operation is the image resizing process by scaling-up or scaling-down the source image from active window size to display window size for displaying on LCD panel or TV set.

Horizontal scaling and vertical scaling are realized independently.

1.Scaling factor

Pseudo Code:

```
void calc_win_scl_factor(LCDC_WIN_PARAMETERS *p_win_para)
{
    u16 srcW;
    u16 srcH;
    u16 dstW;
    u16 dstH;
    u8   is_3d_mix;
    u16 yrgb_srcW;
    u16 yrgb_srcH;
    u16 yrgb_dstW;
    u16 yrgb_dstH;
    u32 yrgb_vScaleDnMult;
    u32 yrgb_xscl_factor;
    u32 yrgb_yscl_factor;
    u8   yrgb_vsd_bil_gt2;
    u8   yrgb_vsd_bil_gt4;
    u8   yrgb_vsd_bil_extra;

    u16 cbcr_srcW;
    u16 cbcr_srcH;
    u16 cbcr_dstW;
    u16 cbcr_dstH;
    u32 cbcr_vScaleDnMult;
    u32 cbcr_xscl_factor;
    u32 cbcr_yscl_factor;
    u8   cbcr_vsd_bil_gt2;
    u8   cbcr_vsd_bil_gt4;
    u8   cbcr_vsd_bil_extra;

    //-------------------------------------------------------------------------
    //width and height, 3D enable
    is_3d_mix = (p_win_para->win_3d_en == ENABLE) && ((p_win_para->win_3d_mode == MIX_R_GB)
|| (p_win_para->win_3d_mode == MIX_G_RB) || (p_win_para->win_3d_mode == MIX_B_RG));

    srcW   = p_win_para->win_act_width;
    if((p_win_para->win_3d_en == ENABLE)&&(p_win_para->win_3d_mode ==
INTERLEAVE_HORIZONTAL)) {
        dstW   = p_win_para->dsp_win_width >> 1;
    } else {
        dstW   = p_win_para->dsp_win_width;
    }
    srcH = p_win_para->win_act_height;

    if((p_win_para->win_3d_en == ENABLE)&&(p_win_para->win_3d_mode == INTERLEAVE_VERTICAL))
{
        dstH = p_win_para->dsp_win_height >> 1;
    } else {
        dstH = p_win_para->dsp_win_height;
    }

    dstH = p_win_para->dsp_win_height;

    if(p_win_para->win_3d_en == ENABLE) {
        if(p_win_para->win_3d_mode == INTERLEAVE_HORIZONTAL) {
            dstW = dstW/2;
        }
        else if(p_win_para->win_3d_mode == INTERLEAVE_VERTICAL) {
            dstH = dstH/2;
        }
    }

    //-------------------------------------------------------------------------
    //SCALE MODE(YGRB)
    yrgb_srcW = srcW;
    yrgb_dstW = dstW;
    yrgb_srcH = srcH;
    yrgb_dstH = dstH;

    printf("[hxx_dbg] yrgb_srcW=%d; yrgb_dstW=%d; yrgb_srcH=%d;
yrgb_dstH=%d;\n",yrgb_srcW,yrgb_dstW,yrgb_srcH,yrgb_dstH);
```

```
if (yrgb_srcW < yrgb_dstW) {
    p_win_para->yrgb_hor_scl_mode = SCALE_UP;
} else if (yrgb_srcW > yrgb_dstW) {
    p_win_para->yrgb_hor_scl_mode = SCALE_DOWN;
} else {
    p_win_para->yrgb_hor_scl_mode = SCALE_NONE;
}

if (yrgb_srcH < yrgb_dstH) {
    p_win_para->yrgb_ver_scl_mode = SCALE_UP;
} else if (yrgb_srcH   > yrgb_dstH) {
    p_win_para->yrgb_ver_scl_mode = SCALE_DOWN;
} else {
    p_win_para->yrgb_ver_scl_mode = SCALE_NONE;
}

//SCALE MODE(CBCR)
if(p_win_para->win_lcdc_format == LCDC_FMT_YUV422) {
    cbcr_srcW = srcW/2;
    cbcr_dstW = dstW;
    cbcr_srcH = srcH;
    cbcr_dstH = dstH;

    if (cbcr_srcW < cbcr_dstW) {
        p_win_para->cbr_hor_scl_mode = SCALE_UP;
    } else if (cbcr_srcW > cbcr_dstW) {
        p_win_para->cbr_hor_scl_mode = SCALE_DOWN;
    } else {
        p_win_para->cbr_hor_scl_mode = SCALE_NONE;
    }

    if (cbcr_srcH < cbcr_dstH) {
        p_win_para->cbr_ver_scl_mode = SCALE_UP;
    } else if (cbcr_srcH > cbcr_dstH) {
        p_win_para->cbr_ver_scl_mode = SCALE_DOWN;
    } else {
        p_win_para->cbr_ver_scl_mode = SCALE_NONE;
    }
}
else if(p_win_para->win_lcdc_format == LCDC_FMT_YUV420) {
    cbcr_srcW = srcW/2;
    cbcr_dstW = dstW;
    cbcr_srcH = srcH/2;
    cbcr_dstH = dstH;

    if (cbcr_srcW < cbcr_dstW) {
        p_win_para->cbr_hor_scl_mode = SCALE_UP;
    } else if (cbcr_srcW > cbcr_dstW) {
        p_win_para->cbr_hor_scl_mode = SCALE_DOWN;
    } else {
        p_win_para->cbr_hor_scl_mode = SCALE_NONE;
    }

    if (cbcr_srcH < cbcr_dstH) {
        p_win_para->cbr_ver_scl_mode = SCALE_UP;
    } else if (cbcr_srcH > cbcr_dstH) {
        p_win_para->cbr_ver_scl_mode = SCALE_DOWN;
    } else {
        p_win_para->cbr_ver_scl_mode = SCALE_NONE;
    }
}
else if(p_win_para->win_lcdc_format == LCDC_FMT_YUV444) {
    cbcr_srcW = srcW;
    cbcr_dstW = dstW;
    cbcr_srcH = srcH;
    cbcr_dstH = dstH;
```

```
        if (cbcr_srcW < cbcr_dstW) {
            p_win_para->cbr_hor_scl_mode = SCALE_UP;
        } else if (cbcr_srcW > cbcr_dstW) {
            p_win_para->cbr_hor_scl_mode = SCALE_DOWN;
        } else {
            p_win_para->cbr_hor_scl_mode = SCALE_NONE;
        }

        if (cbcr_srcH < cbcr_dstH) {
            p_win_para->cbr_ver_scl_mode = SCALE_UP;
        } else if (cbcr_srcH > cbcr_dstH) {
            p_win_para->cbr_ver_scl_mode = SCALE_DOWN;
        } else {
            p_win_para->cbr_ver_scl_mode = SCALE_NONE;
        }
    } else {
        cbcr_srcW = 0;
        cbcr_dstW = 0;
        cbcr_srcH = 0;
        cbcr_dstH = 0;

        p_win_para->cbr_hor_scl_mode = SCALE_NONE;
        p_win_para->cbr_ver_scl_mode = SCALE_NONE;
    }


    printf("[hxx_dbg] cbcr_srcW=%d; cbcr_dstW=%d; cbcr_srcH=%d;
cbcr_dstH=%d;\n",cbcr_srcW,cbcr_dstW,cbcr_srcH,cbcr_dstH);

    //--------------------------------------------------------------------------
    //SCALE ALGORITHM
    if( (p_win_para->win_lcdc_format == LCDC_FMT_YUV422) || (p_win_para->win_lcdc_format ==
LCDC_FMT_YUV420) ) {
        if(p_win_para->cbr_hor_scl_mode == SCALE_DOWN) {
            if(cbcr_dstW > 3840) {
                printf("ERROR cbcr_dst_width exceeds 3840\n");
                exit (-1);
            } else if(cbcr_dstW > 2560) {
                p_win_para->win_lb_mode = LB_RGB_3840X2;
            } else if(cbcr_dstW > 1920) {
                if(p_win_para->yrgb_hor_scl_mode == SCALE_DOWN) {
                    if(yrgb_dstW > 3840) {
                        printf("ERROR yrgb_dst_width exceeds 3840\n");
                        exit (-1);
                    } else if(yrgb_dstW > 2560) {
                        p_win_para->win_lb_mode = LB_RGB_3840X2;
                    } else if(yrgb_dstW > 1920) {
                        p_win_para->win_lb_mode = LB_RGB_2560X4;
                    } else {
                        printf("ERROR never run here!yrgb_dstW<1920 ==> cbcr_dstW<1920");
                        exit (-1);
                    }
                }
            } else if(cbcr_dstW > 1280) {
                p_win_para->win_lb_mode = LB_YUV_3840X5;
            } else {
                p_win_para->win_lb_mode = LB_YUV_2560X8;
            }
        } else { //SCALE_UP or SCALE_NONE
            if(cbcr_srcW > 3840) {
                printf("ERROR cbcr_act_width exceeds 3840\n");
                exit (-1);
            } else if(cbcr_srcW > 2560) {
                p_win_para->win_lb_mode = LB_RGB_3840X2;
            } else if(cbcr_srcW > 1920) {
                if(p_win_para->yrgb_hor_scl_mode == SCALE_DOWN) {
                    if(yrgb_dstW > 3840) {
                        printf("ERROR yrgb_dst_width exceeds 3840\n");
                        exit (-1);
                    } else if(yrgb_dstW > 2560) {
                        p_win_para->win_lb_mode = LB_RGB_3840X2;
                    } else if(yrgb_dstW > 1920) {
                        p_win_para->win_lb_mode = LB_RGB_2560X4;
                    } else {
                        printf("ERROR never run here!yrgb_dstW<1920 ==> cbcr_dstW<1920 ==>
cbcr_srcW<=1920\n");
                        exit (-1);
                    }
                }
            } else if(cbcr_srcW > 1280) {
                p_win_para->win_lb_mode = LB_YUV_3840X5;
            } else {
                p_win_para->win_lb_mode = LB_YUV_2560X8;
            }
        }
    }
    else {
        if(p_win_para->yrgb_hor_scl_mode == SCALE_DOWN) {
            if(yrgb_dstW > 3840) {
                printf("ERROR yrgb_dsp_width exceeds 3840\n");
                exit (-1);
            } else if(yrgb_dstW > 2560) {
                p_win_para->win_lb_mode = LB_RGB_3840X2;
            } else if(yrgb_dstW > 1920) {
                p_win_para->win_lb_mode = LB_RGB_2560X4;
            } else if(yrgb_dstW > 1280){
                p_win_para->win_lb_mode = LB_RGB_1920X5;
            } else {
                p_win_para->win_lb_mode = LB_RGB_1280X8;
            }
        } else { //SCALE_UP or SCALE_NONE
            if(yrgb_srcW > 3840) {
                printf("ERROR yrgb_act_width exceeds 3840\n");
                exit (-1);
            } else if(yrgb_srcW > 2560) {
                p_win_para->win_lb_mode = LB_RGB_3840X2;
            } else if(yrgb_srcW > 1920) {
                p_win_para->win_lb_mode = LB_RGB_2560X4;
            } else if(yrgb_srcW > 1280){
                p_win_para->win_lb_mode = LB_RGB_1920X5;
            } else {
                p_win_para->win_lb_mode = LB_RGB_1280X8;
            }
        }
    }

    printf("[hxx_dbg] p_win_para->win_lb_mode = %d;\n",p_win_para->win_lb_mode);

    //vsd/vsu scale ALGORITHM
    switch(p_win_para->win_lb_mode) {
        case LB_YUV_3840X5:
            p_win_para->yrgb_vsu_mode = SCALE_UP_BIC   ;
          //p_win_para->yrgb_vsd_mode = SCALE_DOWN_BIL; //not to specify
            p_win_para->cbr_vsu_mode   = SCALE_UP_BIC   ;
          //p_win_para->cbr_vsd_mode   = SCALE_DOWN_BIL; //not to specify
            break;
        case LB_YUV_2560X8:
            p_win_para->yrgb_vsu_mode = SCALE_UP_BIC   ;
          //p_win_para->yrgb_vsd_mode = SCALE_DOWN_BIL; //not to specify
            p_win_para->cbr_vsu_mode   = SCALE_UP_BIC   ;
          //p_win_para->cbr_vsd_mode   = SCALE_DOWN_BIL; //not to specify
            break;
        case LB_RGB_3840X2:
            if(p_win_para->yrgb_ver_scl_mode != SCALE_NONE) {
                printf("ERROR : not allow yrgb ver scale\n");
                exit(-1);
            }

            if(p_win_para->cbr_ver_scl_mode != SCALE_NONE) {
                printf("ERROR : not allow cbcr ver scale\n");
                exit(-1);
            }

            //p_win_para->yrgb_vsu_mode = SCALE_UP_BIC   ;
            //p_win_para->yrgb_vsd_mode = SCALE_DOWN_BIL;
            //p_win_para->cbr_vsu_mode   = SCALE_UP_BIC   ;
            //p_win_para->cbr_vsd_mode   = SCALE_DOWN_BIL;
            break;
        case LB_RGB_2560X4:
            p_win_para->yrgb_vsu_mode = SCALE_UP_BIL   ; //<2
          //p_win_para->yrgb_vsd_mode = SCALE_DOWN_BIL; //<2 //not to specify
            p_win_para->cbr_vsu_mode   = SCALE_UP_BIL   ; //<2
          //p_win_para->cbr_vsd_mode   = SCALE_DOWN_BIL; //<2 //not to specify
            break;
        case LB_RGB_1920X5:
            p_win_para->yrgb_vsu_mode = SCALE_UP_BIC   ;
          //p_win_para->yrgb_vsd_mode = SCALE_DOWN_BIL; //not to specify
            p_win_para->cbr_vsu_mode   = SCALE_UP_BIC   ;
          //p_win_para->cbr_vsd_mode   = SCALE_DOWN_BIL; //not to specify
            break;
        case LB_RGB_1280X8:
            p_win_para->yrgb_vsu_mode = SCALE_UP_BIC   ;
          //p_win_para->yrgb_vsd_mode = SCALE_DOWN_BIL; //not to specify
            p_win_para->cbr_vsu_mode   = SCALE_UP_BIC   ;
          //p_win_para->cbr_vsd_mode   = SCALE_DOWN_BIL; //not to specify
            break;
        default :
            break;
    }
    //----------------------------------------------------------------------
    //SCALE FACTOR
    yrgb_vsd_bil_gt4 = 0;
    yrgb_vsd_bil_gt2 = 0;
    cbcr_vsd_bil_gt4 = 0;
    cbcr_vsd_bil_gt2 = 0;

    //(1.1)YRGB HOR SCALE FACTOR
    switch(p_win_para->yrgb_hor_scl_mode) {
        case SCALE_NONE:
            yrgb_xscl_factor = (1<<SCALE_FACTOR_DEFAULT_FIXPOINT_SHIFT);
            break;

        case SCALE_UP   :
            yrgb_xscl_factor = GET_SCALE_FACTOR_BIC(yrgb_srcW, yrgb_dstW);
            break;

        case SCALE_DOWN:
            switch(p_win_para->yrgb_hsd_mode)
            {
                case SCALE_DOWN_BIL:
                    yrgb_xscl_factor = GET_SCALE_FACTOR_BILI_DN(yrgb_srcW, yrgb_dstW);
                    break;
                case SCALE_DOWN_AVG:
                    yrgb_xscl_factor = GET_SCALE_FACTOR_AVRG(yrgb_srcW, yrgb_dstW);
                    break;
                default :
                    break;
            } //p_win_para->yrgb_vsd_mode
            break;

        default :
            break;
    } //p_win_para->yrgb_hor_scl_mode

    //(1.2)YRGB VER SCALE FACTOR
    switch(p_win_para->yrgb_ver_scl_mode)
    {
        case SCALE_NONE:
            yrgb_yscl_factor = (1<<SCALE_FACTOR_DEFAULT_FIXPOINT_SHIFT);
            break;

        case SCALE_UP   :
            switch(p_win_para->yrgb_vsu_mode)
            {
                case SCALE_UP_BIL:
                    yrgb_yscl_factor = GET_SCALE_FACTOR_BILI_UP(yrgb_srcH, yrgb_dstH);
                    break;
                case SCALE_UP_BIC:
                    if(yrgb_srcH < 3) {
                        printf("[hxx_dbg] yrgb_srcH should be greater than 3 !!!\n");
                        exit (-1);
                    }
                    yrgb_yscl_factor = GET_SCALE_FACTOR_BIC(yrgb_srcH, yrgb_dstH);
                    break;
                default :
                    break;
            } //p_win_para->yrgb_vsu_mode
            break;

        case SCALE_DOWN:
            switch(p_win_para->yrgb_vsd_mode)
            {
                case SCALE_DOWN_BIL:
                    yrgb_vScaleDnMult = getHardWareVSkipLines(yrgb_srcH, yrgb_dstH);
                    yrgb_yscl_factor   = GET_SCALE_FACTOR_BILI_DN_VSKIP(yrgb_srcH, yrgb_dstH,
yrgb_vScaleDnMult);
                    //printf("[hxx_dbg] yrgb_vScaleDnMult=%d;
yrgb_yscl_factor=%4x;\n",yrgb_vScaleDnMult,yrgb_yscl_factor);
                    if(yrgb_vScaleDnMult == 4) {
                        yrgb_vsd_bil_gt4 = 1;
                        yrgb_vsd_bil_gt2 = 0;
                    } else if(yrgb_vScaleDnMult == 2) {
                        yrgb_vsd_bil_gt4 = 0;
                        yrgb_vsd_bil_gt2 = 1;
                    } else {
                        yrgb_vsd_bil_gt4 = 0;
                        yrgb_vsd_bil_gt2 = 0;
                    }
                    break;
                case SCALE_DOWN_AVG:
                    yrgb_yscl_factor = GET_SCALE_FACTOR_AVRG(yrgb_srcH, yrgb_dstH);
                    break;
                default :
                    break;
            } //p_win_para->yrgb_vsd_mode
            break;

        default :
            break;
    } //p_win_para->yrgb_hor_scl_mode

    p_win_para->win_h_YRGB_factor = yrgb_xscl_factor;
    p_win_para->win_v_YRGB_factor = yrgb_yscl_factor;
    p_win_para->vsd_yrgb_gt4        = yrgb_vsd_bil_gt4;
    p_win_para->vsd_yrgb_gt2        = yrgb_vsd_bil_gt2;

    //(2.1)CBCR HOR SCALE FACTOR
    switch(p_win_para->cbr_hor_scl_mode)
    {
        case SCALE_NONE:
            cbcr_xscl_factor = (1<<SCALE_FACTOR_DEFAULT_FIXPOINT_SHIFT);
            break;

        case SCALE_UP   :
            cbcr_xscl_factor = GET_SCALE_FACTOR_BIC(cbcr_srcW, cbcr_dstW);
            break;

        case SCALE_DOWN:
            switch(p_win_para->cbr_hsd_mode)
            {
                case SCALE_DOWN_BIL:
                    cbcr_xscl_factor = GET_SCALE_FACTOR_BILI_DN(cbcr_srcW, cbcr_dstW);
                    break;
                case SCALE_DOWN_AVG:
                    cbcr_xscl_factor = GET_SCALE_FACTOR_AVRG(cbcr_srcW, cbcr_dstW);
                    break;
                default :
                    break;
            } //p_win_para->cbr_vsd_mode
            break;

        default :
            break;
    } //p_win_para->cbr_hor_scl_mode

    //(2.2)CBCR VER SCALE FACTOR
    switch(p_win_para->cbr_ver_scl_mode)
    {
        case SCALE_NONE:
            cbcr_yscl_factor = (1<<SCALE_FACTOR_DEFAULT_FIXPOINT_SHIFT);
            break;

        case SCALE_UP   :
            switch(p_win_para->cbr_vsu_mode)
            {
                case SCALE_UP_BIL:
                    cbcr_yscl_factor = GET_SCALE_FACTOR_BILI_UP(cbcr_srcH, cbcr_dstH);
                    break;
                case SCALE_UP_BIC:
                    if(cbcr_srcH < 3) {
                        printf("[hxx_dbg] cbcr_srcH should be greater than 3 !!!\n");
                        exit (-1);
                    }
                    cbcr_yscl_factor = GET_SCALE_FACTOR_BIC(cbcr_srcH, cbcr_dstH);
                    break;
                default :
                    break;
            } //p_win_para->cbr_vsu_mode
            break;

        case SCALE_DOWN:
            switch(p_win_para->cbr_vsd_mode)
            {
                case SCALE_DOWN_BIL:
                    cbcr_vScaleDnMult = getHardWareVSkipLines(cbcr_srcH, cbcr_dstH);
                    cbcr_yscl_factor   = GET_SCALE_FACTOR_BILI_DN_VSKIP(cbcr_srcH, cbcr_dstH,
cbcr_vScaleDnMult);
                    //printf("[hxx_dbg] cbcr_vScaleDnMult=%d;\n",cbcr_vScaleDnMult);
                    if(cbcr_vScaleDnMult == 4) {
                        cbcr_vsd_bil_gt4 = 1;
                        cbcr_vsd_bil_gt2 = 0;
                    } else if(cbcr_vScaleDnMult == 2) {
                        cbcr_vsd_bil_gt4 = 0;
                        cbcr_vsd_bil_gt2 = 1;
                    } else {
```

```
                    cbcr_vsd_bil_gt4 = 0;
                    cbcr_vsd_bil_gt2 = 0;
                }
                break;
            case SCALE_DOWN_AVG:
                cbcr_yscl_factor = GET_SCALE_FACTOR_AVRG(cbcr_srcH, cbcr_dstH);
                break;
            default :
                break;
        } //p_win_para->cbr_vsd_mode
        break;

    default :
        break;
} //p_win_para->cbr_hor_scl_mode

p_win_para->vsd_cbr_gt4        = cbcr_vsd_bil_gt4;
p_win_para->vsd_cbr_gt2        = cbcr_vsd_bil_gt2;
p_win_para->win_h_Cbr_factor = cbcr_xscl_factor;
p_win_para->win_v_Cbr_factor = cbcr_yscl_factor;

//---------------------------------------------------------------------------------
switch(p_win_para->yrgb_hor_scl_mode) {
    case SCALE_NONE :
        printf("[hxx_dbg] X YRGB SCALE_NONE\n");
        break;
    case SCALE_UP :
        printf("[hxx_dbg] X YRGB SCALE_UP_BICUBIC; yrgb_xscl_factor=%04x;\n",yrgb_xscl_factor);
        //switch(p_win_para->yrgb_hsu_mode) {
        //     case SCALE_UP_BIL :
        //          printf("[hxx_dbg] X YRGB SCALE_UP_BILINEAR;[ERROR] yrgb_xscl_factor=%04x;\n
",yrgb_xscl_factor);
        //          break;
        //     case SCALE_UP_BIC :
        //          printf("[hxx_dbg] X YRGB SCALE_UP_BICUBIC; yrgb_xscl_factor=%04x;\n
",yrgb_xscl_factor);
        //          break;
        //     default :
        //}
        break;
    case SCALE_DOWN :
        switch(p_win_para->yrgb_hsd_mode) {
            case SCALE_DOWN_BIL :
                printf("[hxx_dbg] X YRGB SCALE_DOWN_BILINEAR;
yrgb_xscl_factor=%04x;\n",yrgb_xscl_factor);
                break;
            case SCALE_DOWN_AVG :
                printf("[hxx_dbg] X YRGB SCALE_DOWN_AVERAGE;
yrgb_xscl_factor=%04x;\n",yrgb_xscl_factor);
                break;
            default:
                break;
        }
        break;
    default:
        break;
}

switch(p_win_para->yrgb_ver_scl_mode) {
    case SCALE_NONE :
        printf("[hxx_dbg] Y YRGB SCALE_NONE\n");
        break;
    case SCALE_UP :
        switch(p_win_para->yrgb_vsu_mode) {
            case SCALE_UP_BIL :
                printf("[hxx_dbg] Y YRGB SCALE_UP_BILINEAR;
yrgb_yscl_factor=%04x;\n",yrgb_yscl_factor);
```

```
                    break;
                case SCALE_UP_BIC :
                    printf("[hxx_dbg] Y YRGB SCALE_UP_BICUBIC;
yrgb_yscl_factor=%04x;\n",yrgb_yscl_factor);
                    break;
                default :
                    break;
            }
            break;
        case SCALE_DOWN :
            switch(p_win_para->yrgb_vsd_mode) {
                case SCALE_DOWN_BIL :
                    printf("[hxx_dbg] Y YRGB SCALE_DOWN_BILINEAR;
yrgb_yscl_factor=%04x;\n",yrgb_yscl_factor);
                    break;
                case SCALE_DOWN_AVG :
                    printf("[hxx_dbg] Y YRGB SCALE_DOWN_AVERAGE;
yrgb_yscl_factor=%04x;\n",yrgb_yscl_factor);
                    break;
                default:
                    break;
            }
            break;
        default:
            break;
    }
    //--------------------------------------------------------------------------------
    switch(p_win_para->cbr_hor_scl_mode) {
        case SCALE_NONE :
            printf("[hxx_dbg] X CBCR SCALE_NONE\n");
            break;
        case SCALE_UP :
            printf("[hxx_dbg] X CBCR SCALE_UP_BICUBIC; cbcr_xscl_factor=%04x;\n",cbcr_xscl_factor);
            //switch(p_win_para->cbr_hsu_mode) {
            //     printf("[hxx_dbg] X CBCR SCALE_UP_BICUBIC; cbcr_xscl_factor=%04x;\n
",cbcr_xscl_factor);
            //     case SCALE_UP_BIL :
            //          printf("[hxx_dbg] X CBCR SCALE_UP_BILINEAR;[ERROR] cbcr_xscl_factor=%04x;\n
",cbcr_xscl_factor);
            //          break;
            //     case SCALE_UP_BIC :
            //          printf("[hxx_dbg] X CBCR SCALE_UP_BICUBIC; cbcr_xscl_factor=%04x;\n
",cbcr_xscl_factor);
            //          break;
            //     default :
            //}
            break;
        case SCALE_DOWN :
            switch(p_win_para->cbr_hsd_mode) {
                case SCALE_DOWN_BIL :
                    printf("[hxx_dbg] X CBCR SCALE_DOWN_BILINEAR;
cbcr_xscl_factor=%04x;\n",cbcr_xscl_factor);
                    break;
                case SCALE_DOWN_AVG :
                    printf("[hxx_dbg] X CBCR SCALE_DOWN_AVERAGE;
cbcr_xscl_factor=%04x;\n",cbcr_xscl_factor);
                    break;
                default:
                    break;
            }
            break;
        default:
            break;
    }

    switch(p_win_para->cbr_ver_scl_mode) {
        case SCALE_NONE :
            printf("[hxx_dbg] Y CBCR SCALE_NONE\n");
            break;
        case SCALE_UP :
            switch(p_win_para->cbr_vsu_mode) {
                case SCALE_UP_BIL :
                    printf("[hxx_dbg] Y CBCR SCALE_UP_BILINEAR;
cbcr_yscl_factor=%04x;\n",cbcr_yscl_factor);
                    break;
                case SCALE_UP_BIC :
                    printf("[hxx_dbg] Y CBCR SCALE_UP_BICUBIC;
cbcr_yscl_factor=%04x;\n",cbcr_yscl_factor);
                    break;
                default :
                    break;
            }
            break;
        case SCALE_DOWN :
            switch(p_win_para->cbr_vsd_mode) {
                case SCALE_DOWN_BIL :
                    printf("[hxx_dbg] Y CBCR SCALE_DOWN_BILINEAR;
cbcr_yscl_factor=%04x;\n",cbcr_yscl_factor);
                    break;
                case SCALE_DOWN_AVG :
                    printf("[hxx_dbg] Y CBCR SCALE_DOWN_AVERAGE;
cbcr_yscl_factor=%04x;\n",cbcr_yscl_factor);
                    break;
                default:
                    break;
            }
            break;
        default:
            break;
    }

    //-----------------------------------------------------------------------------------
    //printf("[hxx_dbg] p_win_para->yrgb_hor_scl_mode=%d;\n",p_win_para->yrgb_hor_scl_mode);
    //printf("[hxx_dbg] p_win_para->yrgb_ver_scl_mode=%d;\n",p_win_para->yrgb_ver_scl_mode);
    //printf("[hxx_dbg] p_win_para->cbcr_hor_scl_mode=%d;\n",p_win_para->cbr_hor_scl_mode );
    //printf("[hxx_dbg] p_win_para->cbcr_ver_scl_mode=%d;\n",p_win_para->cbr_ver_scl_mode );

    //printf("[hxx_dbg] p_win_para->yrgb_hsd_mode= %d;\n",p_win_para->yrgb_hsd_mode);
    //printf("[hxx_dbg] p_win_para->cbr_hsd_mode = %d;\n",p_win_para->cbr_hsd_mode );

    //printf("[hxx_dbg] p_win_para->yrgb_vsu_mode= %d;\n",p_win_para->yrgb_vsu_mode);
    //printf("[hxx_dbg] p_win_para->yrgb_vsd_mode= %d;\n",p_win_para->yrgb_vsd_mode);
    //printf("[hxx_dbg] p_win_para->cbr_vsu_mode = %d;\n",p_win_para->cbr_vsu_mode );
    //printf("[hxx_dbg] p_win_para->cbr_vsd_mode = %d;\n",p_win_para->cbr_vsd_mode );

    //printf("[hxx_dbg] yrgb_xscl_factor= %04x\n", yrgb_xscl_factor);
    //printf("[hxx_dbg] yrgb_yscl_factor= %04x\n", yrgb_yscl_factor);
    //
    //printf("[hxx_dbg] cbcr_xscl_factor= %04x\n", cbcr_xscl_factor);
    //printf("[hxx_dbg] cbcr_yscl_factor= %04x\n", cbcr_yscl_factor);

} //end calc_win_scl_factor
```

2.Limitation of YUV scaling down(3840 < width < 4096)

Limitation of 3840~4096 horizontal scale down for YUV422/420:

YUV422

(1) not support vertical scale up/down if width(>3840) scale down to width(>2560);
(2) support vertical down but only support vertical bilinear scale up if width(>3840) scale down to width(>1920 and <2560);

(3) no limitation if width(>3840) scale down to width(<=1920);

YUV420

(1) not support width(>3840) scale down to width(>2560);
(2) support vertical down but only support vertical bilinear scale up if width(>3840) scale down to width(>1920 and <2560);
(3) no limitation if width(>3840) scale down to width(<=1920);

Since the sampling rate is different for lumina data and chroma data with the format of YCbCr422 and YCbCr420, the scaling factors for lumina data and chroma data are calculated and configured in VOP_WIN0_SCL_FACTOR_Y/ VOP_WIN0_SCL_FACTOR_CBR respectively.

27.3.4 De-flicker

It is necessary to display a non-interlaced video signal on an interlaced display panel (such as TV set). Thus "non-interlaced-to-interlaced conversion" is required.

The easiest approach is to throw away every other active scan line in each non-interlaced frame. Although the cost is minimal, there are problems with this approach. If there is a sharp vertical transition of color or intensity, it will flicker at one-half the refresh rate.

A better solution is to use two lines of non-interlaced data to generate one line of interlace data. Fast vertical transition is smoothed out over several interlace lines.

The vertical filtering of two non-interlaced lines can be done by enabling the vertical scaling offset updated dynamically in different fields, i.e, even field and odd field. The dynamic updated value of scaling offset is half of the scaling factor.

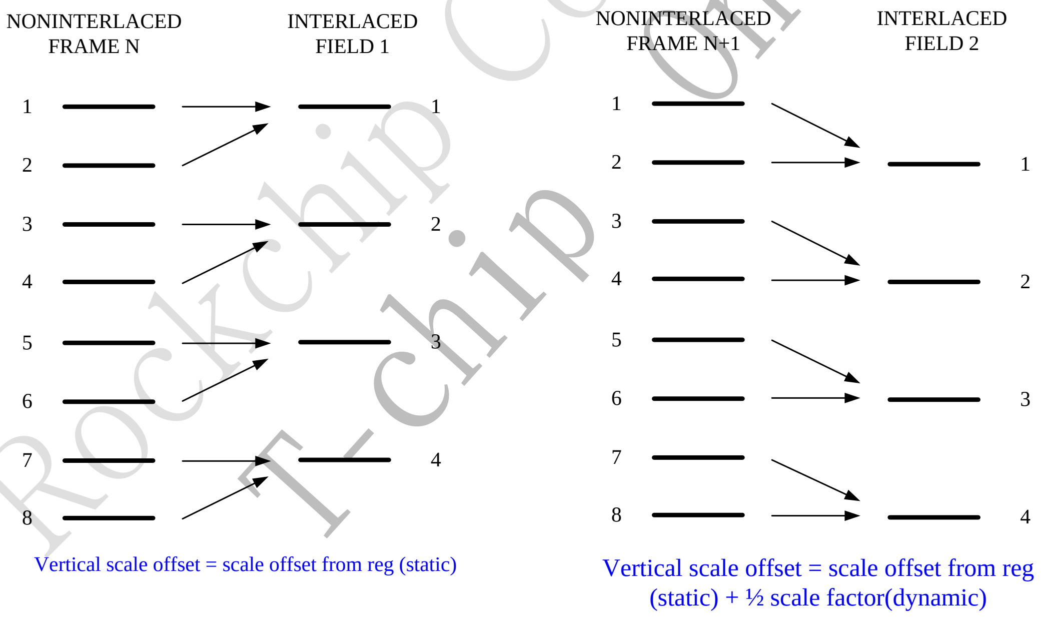

NONINTERLACED FRAME N
INTERLACED FIELD 1
NONINTERLACED FRAME N+1
INTERLACED FIELD 2
1
2
3
4
5
6
7
8
Vertical scale offset = scale offset from reg (static)
Vertical scale offset = scale offset from reg (static) + ½ scale factor(dynamic)

Fig. 27-7 De-flicker

27.3.5 Virtual display

When in virtual display, the active image is part of the virtual (original) image in frame buffer memory.

The virtual width is indicated by setting VIR_STRIDE andVIR_STRIDE for different data format. Note that RGB/BPP has one stride——yrgb_vir_stride； YUV has two virtual stride—— yrgb_vir_stride and cbcr_vir_stride.

For RGB-8bit and YUV-8bit，the stride should be multiples of word (32-bit), with dummy bytes in the end of virtual line if the original width is not 32-bit aligned.

For YUV-10bit，the stride should be multiples of word (-bit), with dummy bytes in the end of virtual line if the original width is not 128-bit aligned.

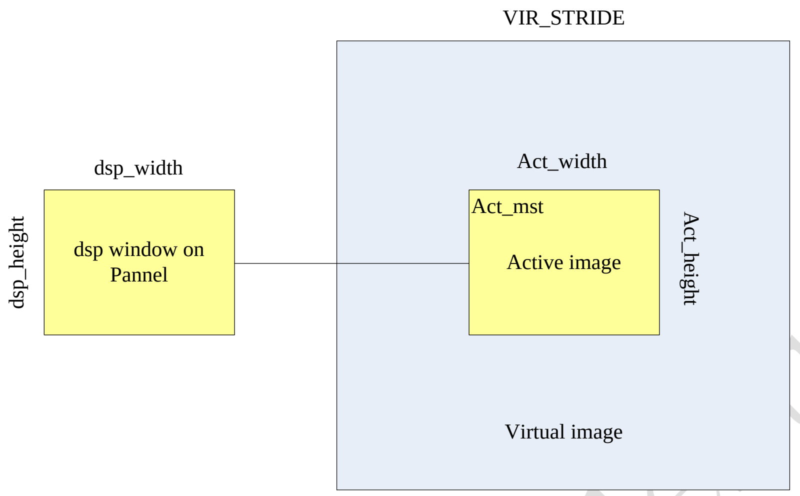


Fig. 27-8 Virtual display

27.3.6 MIRROR display

Mirror display is necessary for the panel with mirror timing interface. There are two types of mirror mode: horizontal mirror(X-Mirror) and vertical mirror(Y-mirror).

The default display order is from left to right(L2R) in horizontal direction and from top to bottom(T2B) in vertical direction. However, when X-Mirror is enable, the horizontal display order is from right to left(R2L); when Y-MIRROR is enable, the vertical display order is from bottom to top(B2T).

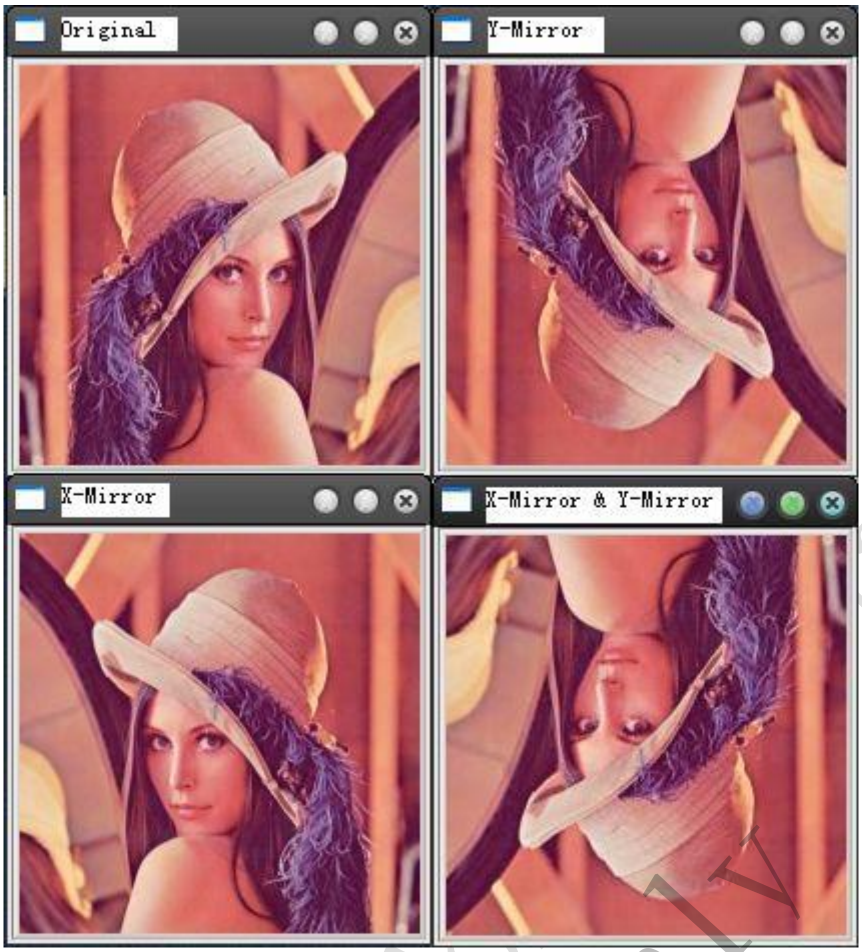
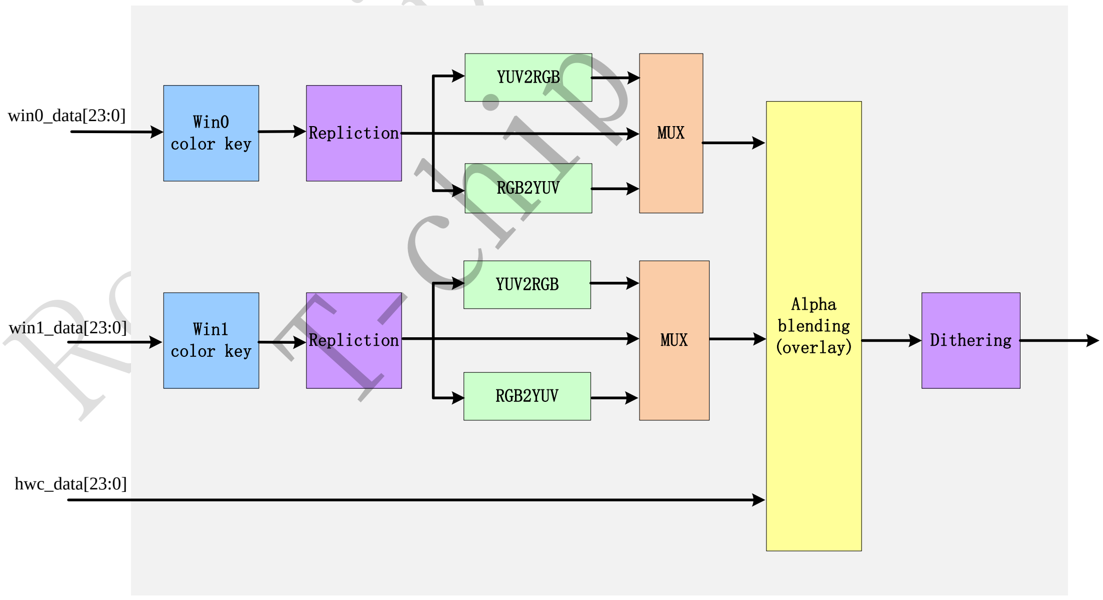


Fig. 27-9 X-Mirror and Y-Mirror

27.3.7 Display process



1.Overlay display

There are totally 4 layers for overlay display: Background, win0 layer, win1 layer and hardware cursor layer(HWC).

Background is a programmable solid color layer, which is always in the bottom of the display screen.

HWC is a 32x32 or 64x64 3-LUT-colors layer, which is always on the top of the display screen.

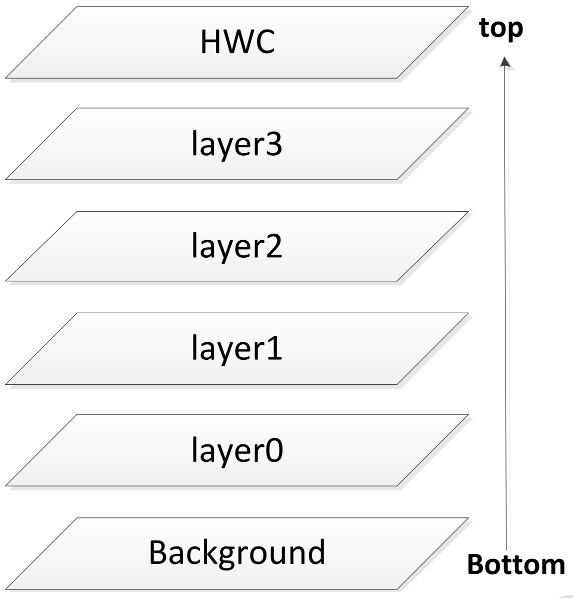
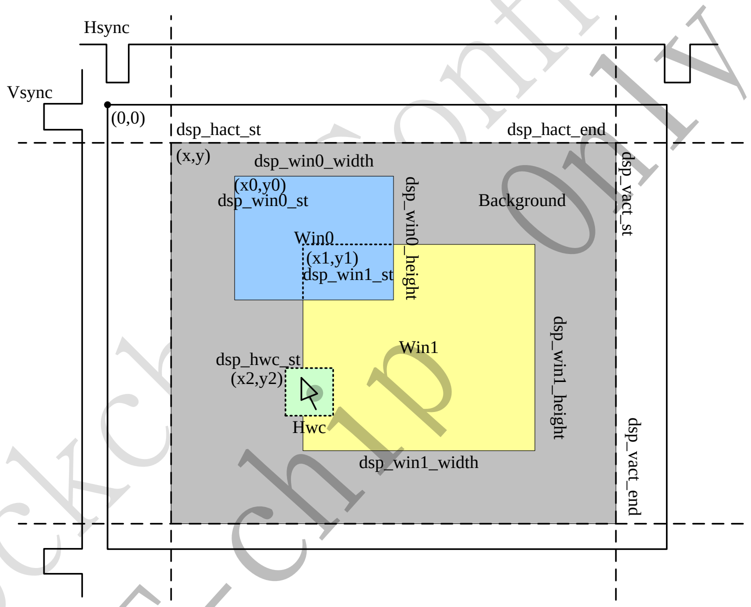


Fig. 27-10 overlay

Following figure is an example of overlay display for win0,win1 and hwc.



2.Post scale down

Post scale down after overlay is supported to fix overscan,that draws the borders of the image beyond the normally visiable area on the screen.

The scale ratio of post scale down is 0.5~1.

Post timing setting

The post scale parameter ,such as,post_dsp_hact_st,post_dsp_hact_end, post_dsp_vact_st,post_dsp_vact_end can be configured.

When post scaling equal "1" ,the post scaler parameter are the same as dsp timing parameter.

eg:

post_dsp_hact_st = dsp_hact_st
post_dsp_hact_end = dap_hact_end
post_dsp_vact_st = dsp_vact_st
post_dsp_vact_end = dsp_vact_end

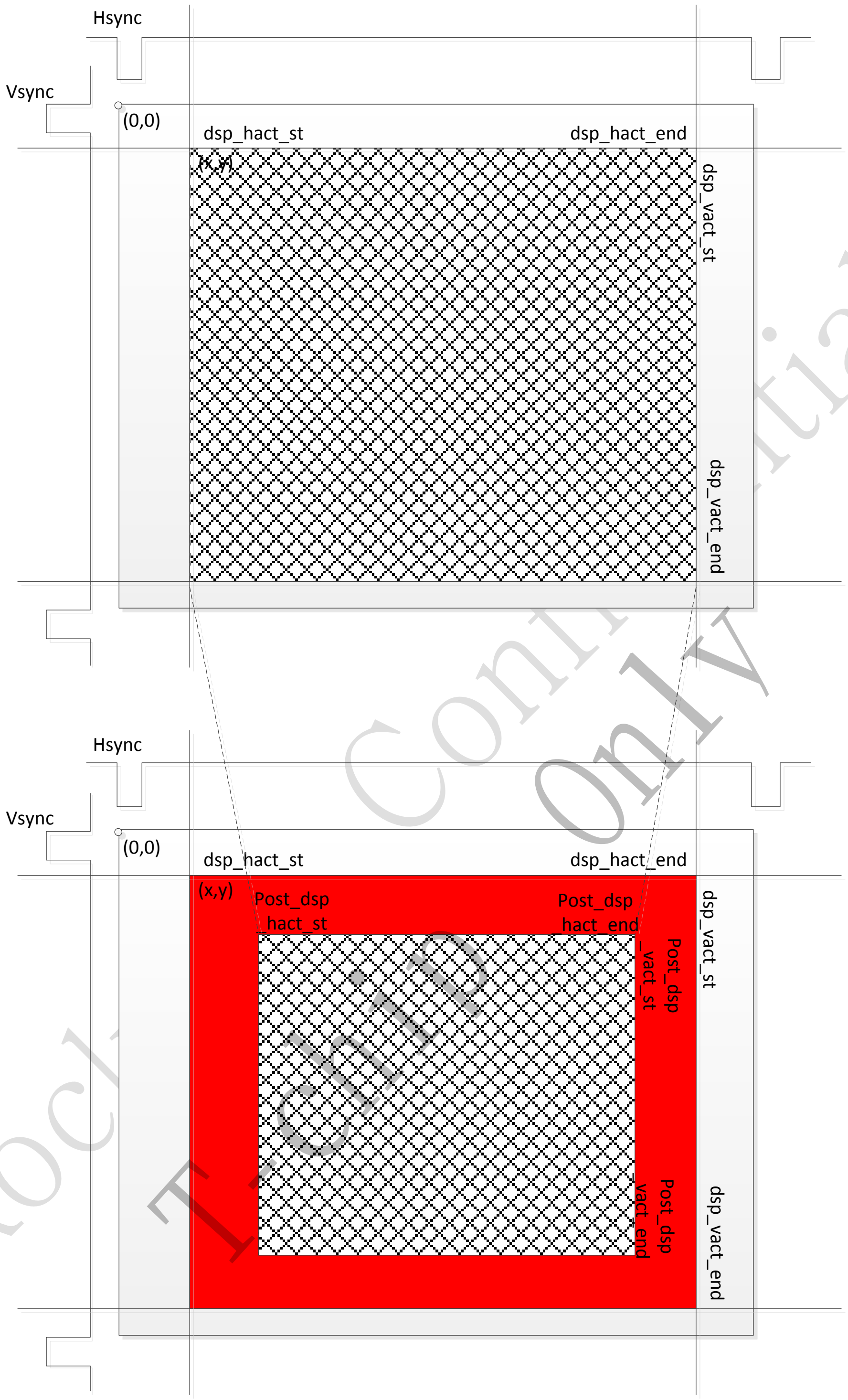


Fig. 27-11 post scaling timing

Post scale down factor

For horizontal scale down,factor = ((src_width*2/3)<<16)/(dst_width-1).

For vertical scale down,factor = ((src_width*2/3)<<16)/(dst_width-1).

3.Transparency color key

The transparency color key value defines the pixel treated as transparent pixel. The pixel whose value is equal to the color key value could not be visible on the screen, instead of the pixel in the under layer or solid background color.

There are two transparency color key for win0 layer and win1 layer respectively. When color key is enable, the transparency process is done after scaling but before YUV2RGB color space converter.

Moreover, transparency color key is just available for non-scaling mode.

Following figure is an example of transparency color key for win0 and win1.

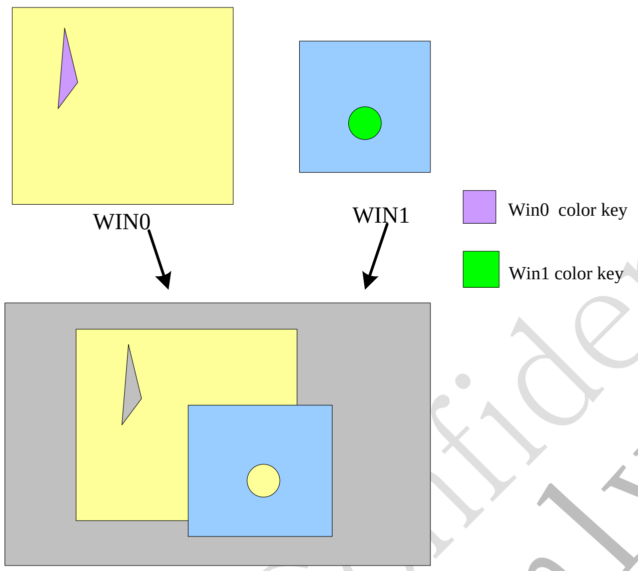
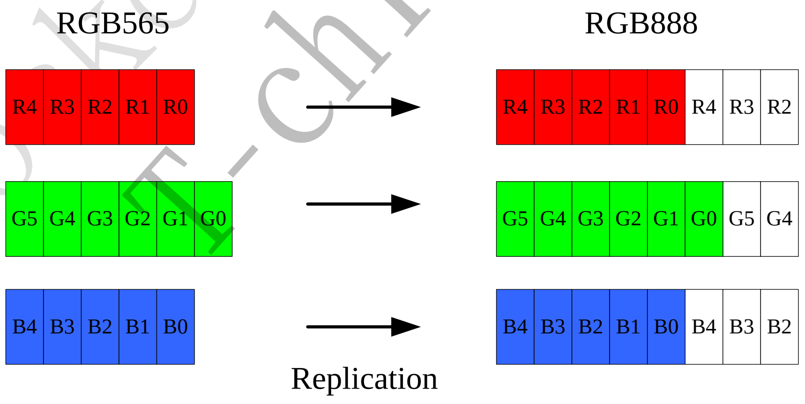


Fig. 27-12 Transparency Color Key

4.Replication(dither up)

If the size of panel data bus is lager than the size of source pixel data,i.e, the source input format is RGB565 and display output format is RGB888, you could do bit replication by replicating MSBs to LSBs if replication is enable (VOP_DSP_CTRL0[9]=1) or filling with "0" to LSBs if replication is disable (VOP_DSP_CTRL0[9]=0).



5.Alpha blending

There are 12 alpha blending mode between two overlay layers for layer1/layer2/layer3/hwc. Layer0 does not support alpha blending with background.

When in per-pixel mode, the alpha value for every pixel is following with the pixel data. i.e, aRGB， and can be scaled like RGB data. Therefore it is just suitable for win0/win1/win2/win3/hwc layer with ARGB data format.

The alpha blending architechture is shown as follows.

Table 27-1 alpha blending mode settings

Blending Mode	**Cs’**	**Fs**	**Cd’**	**Fd**
AA_USER_DEFINED	X	User defined	Cd	User defined
AA_CLEAR	X	0	Cd	0
AA_SRC	X	0	Cd	1
AA_DST	X	1	Cd	1
AA_SRC_OVER	Cs	1	Cd	1-As’’
AA_DST_OVER	Cs	1-As’’	Cd	1
AA_SRC_IN	Cs	As’’	Cd	0
AA_DST_IN	X	0	Cd	As’’
AA_SRC_OUT	Cs	1-As’’	Cd	0
AA_DST_OUT	X	0	Cd	1-As’’
AA_SRC_ATOP	Cs	As’’	Cd	1-As’’
AA_DST_ATOP	Cs	1-As’’	Cd	As’’
AA_XOR	Cs	1-As’’	Cd	1-As’’
AA_SRC_OVER_GLOBAL	Cs*As’’	Ags’’	Cd	1-As’’

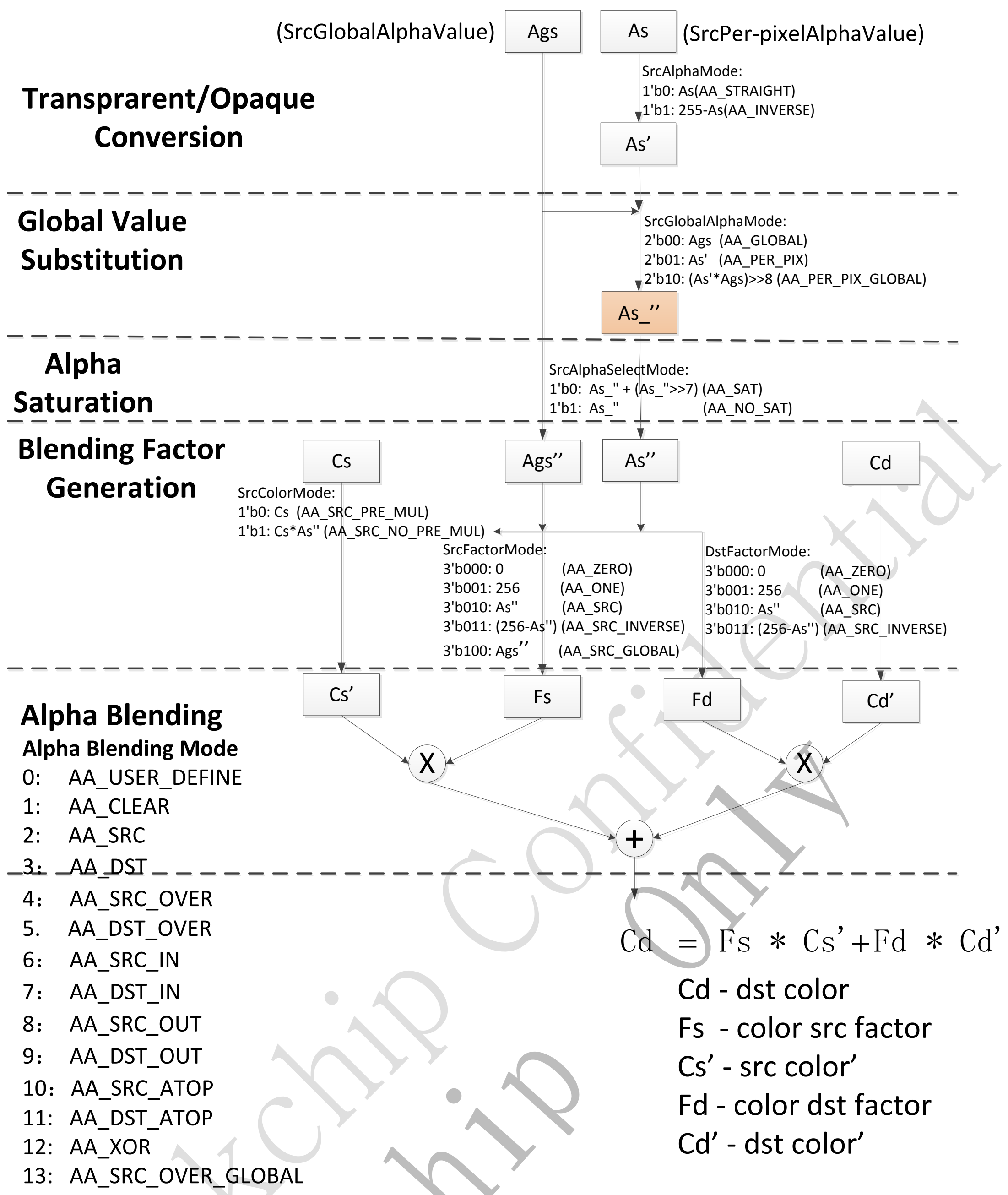


Fig. 27-13 alpha configuration flow

Pseudo Code:

```
switch(alpha_config->alpha_blending_mode)
{
    case AA_USER_DEFINE:

    break;
    case AA_CLEAR:
    alpha_config->src_factor_mode=AA_ZERO;
    alpha_config->dst_factor_mode=AA_ZERO;
    break;
    case AA_SRC:
    alpha_config->src_factor_mode=AA_ONE;
    alpha_config->dst_factor_mode=AA_ZERO;
    break;
    case AA_DST:
    alpha_config->src_factor_mode=AA_ZERO;
    alpha_config->dst_factor_mode=AA_ONE;
    break;
```

```
        case AA_SRC_OVER:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_ONE;
        alpha_config->dst_factor_mode=AA_SRC_INVERSE;
        break;
        case AA_DST_OVER:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC_INVERSE;
        alpha_config->dst_factor_mode=AA_ONE;
        break;
        case AA_SRC_IN:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC;
        alpha_config->dst_factor_mode=AA_ZERO;
        break;
        case AA_DST_IN:
        alpha_config->src_factor_mode=AA_ZERO;
        alpha_config->dst_factor_mode=AA_SRC;
        break;
        case AA_SRC_OUT:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC_INVERSE;
        alpha_config->dst_factor_mode=AA_ZERO;
        break;
        case AA_DST_OUT:
        alpha_config->src_factor_mode=AA_ZERO;
        alpha_config->dst_factor_mode=AA_SRC_INVERSE;
        break;
        case AA_SRC_ATOP:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC;
        alpha_config->dst_factor_mode=AA_SRC_INVERSE;
        break;
        case AA_DST_ATOP:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC_INVERSE;
        alpha_config->dst_factor_mode=AA_SRC;
        break;
        case AA_XOR:
        alpha_config->src_color_mode=AA_SRC_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC_INVERSE;
        alpha_config->dst_factor_mode=AA_SRC_INVERSE;
        break;
        case AA_SRC_OVER_GLOBAL:
        alpha_config->src_global_alpha_mode=AA_PER_PIX_GLOBAL;
        alpha_config->src_color_mode=AA_SRC_NO_PRE_MUL;
        alpha_config->src_factor_mode=AA_SRC_GLOBAL;
        alpha_config->dst_factor_mode=AA_SRC_INVERSE;
        break;
        default:
            printf("alpha mode error\n");
            break;
    }
```

6.CABC

CABC(Content Adaptive Backlight Control) is used to increase the contrast of such LCD-screens the backlight can be (globally) dimmed when the image to be displayed is dark (i.e. not comprising high intensity image data) while the image data is numerically corrected and adapted to the reduced backlight intensity.

Config the panel total pixel num to reg 0x1c4 ,and config the calc pixel num to regfile 0x1c0 (typical calc_pixel_num / total_pixel_num ≈ 80% ~90%).

Config the stage up and stage down to ensure the luminance difference will not to big in each two adjacent frames. Typical value is 0x20~0x40.

There are 3x7 Gaussian filter tables in reg 0x1c8~0x1dc.

default value as follow:
0x1c8 : 0x15110903
0x1cc : 0x00030911
0x1d0 : 0x1a150b04
0x1d4 : 0x00040b15
0x1d8 : 0x15110903
0x1dc : 0x00030911

7.BCSH

BCSH is used to adjust "Brightness,Contrast,Saturation,Hue,like IEP BCSH-8bit. For details,please refer to IEP chapter.

- Extend yuv data from 8bits(IEP) to 10bits.
- The brightness adjust support (-128,127).
- The yuv data of color bar are 10bits.

8.Color space conversion

There are three standards for YUV2RGB-8bit, and BT2020 standard for YUV2RGB-10bit.

- YUV2RGB-8bit

1. yuv to rgb (REC-601) range 0 (Y[16:235], UV[16:240], RGB[0:255])
R = 1.164(Y-16) + 1.596(V-128)
G = 1.164(Y-16) - 0.391(U-128) - 0.813(V-128)
B = 1.164(Y-16) + 2.018(U-128)

2. yuv to rgb (REC-601) range 1 (YUV[0:255], RGB[0:255])
R = (Y-16) + 1.402(V-128)
G = (Y-16) - 0.344(U-128) - 0.714(V-128)
B = (Y-16) + 1.772(U-128)

3. yuv to rgb (REC-709) range 0 (Y[16:235], UV[16:240], RGB[0:255])
R = 1.164(Y-16) + 1.793(V-128)
G = 1.164(Y-16) - 0.213(U-128) - 0.534(V-128)
B = 1.164(Y-16) + 2.115(U-128)

- RGB2YUV-8bit

ccir601
Y = 0.257R + 0.504G + 0.098B + 16
Cb =-0.148R - 0.291G + 0.439B + 128
Cr = 0.439R - 0.368G - 0.071B + 128

- YUV2RGB-10bit

Y = (230R+595G+52B+65536)/1024

U = (-125R-323G+449B+524288)/1024
V = (449R-412G-36B+524288)/1024

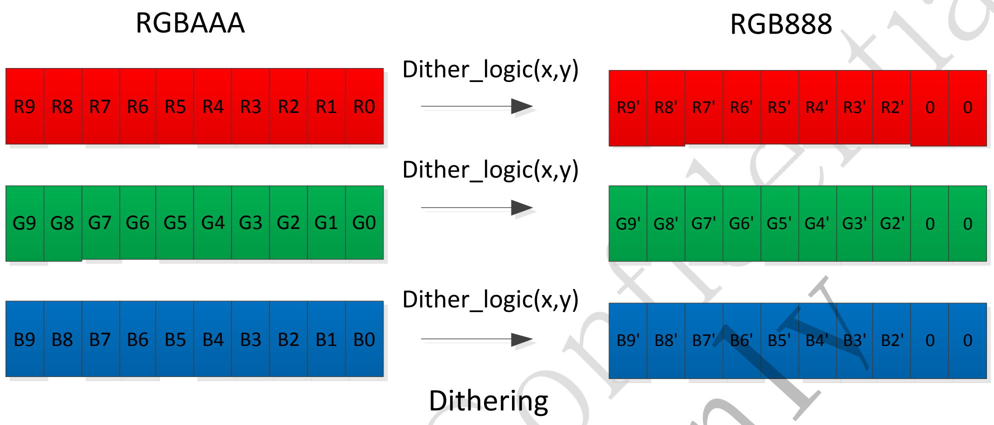
- RGB2YUV-10bit

R = 1.1636Y + 1.6778V - 933.504
G = 1.1636Y - 0.1872U - 0.6501V + 351.9232
B = 1.1636Y + 2.1406U - 1170.4576

9.Dither Down

Dither down directly

The invalid lower bits will be replaced by "0" after dither operation, if disable dither down. eg: 10'b10_1011_00XX →10'b10_1011_0000.

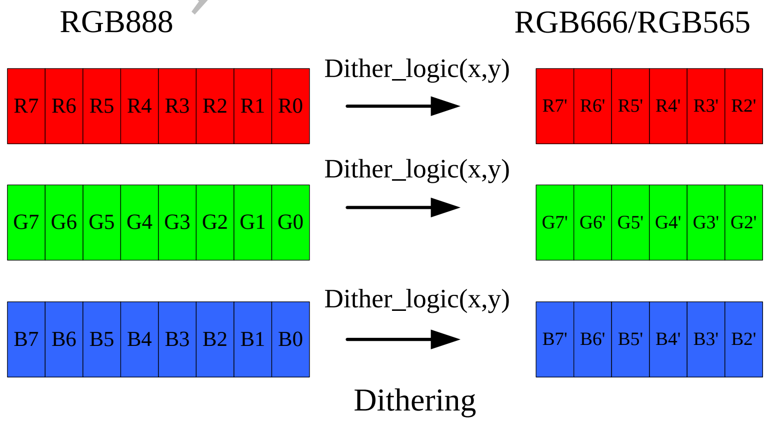

RGBAAA
RGB888
R9 R8 R7 R6 R5 R4 R3 R2 R1 R0
Dither_logic(x,y)
R9' R8' R7' R6' R5' R4' R3' R2' 0 0
G9 G8 G7 G6 G5 G4 G3 G2 G1 G0
Dither_logic(x,y)
G9' G8' G7' G6' G5' G4' G3' G2' 0 0
B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
Dither_logic(x,y)
B9' B8' B7' B6' B5' B4' B3' B2' 0 0
Dithering
Fig. 27-14 Dither down directy

Allegro Dither Down

Dithering is an intentional applied form of noise, using to randomize quantization error, and thereby preventing large-scaling patterns such as "banding".

The pixel value is used by dithering process to display the data in a lower color depth on the LCD panel, i.e, the source input format is RGB888 and display output format is RGB565 or RGB666. When dithering is enable(VOP_DSP_CTRL0[11]=1), the output data is generated by dithering algorithm based on the pixel position and the value of removed bits. Otherwise, the MSBs of the pixel color components are output as display data.

There are two dither modes: "RGB888 to RGB666" and "RGB888 to RGB565", which is defined by VOP_DSP_CTRL0[10]. When VOP_DSP_CTRL0[10] is 1, dithering with "RGB888 to RGB666" is available; otherwise, "RGB888 to RGB565" is used.



FRC Dither Down

The pattern of dither frc was configured by vop regfile vop_base+0x1e0~0x1f4.

The following fig is the default pattern picture in vop, you can config different value of regfile 0x1e0~0x1f4，to change the pattern picture.

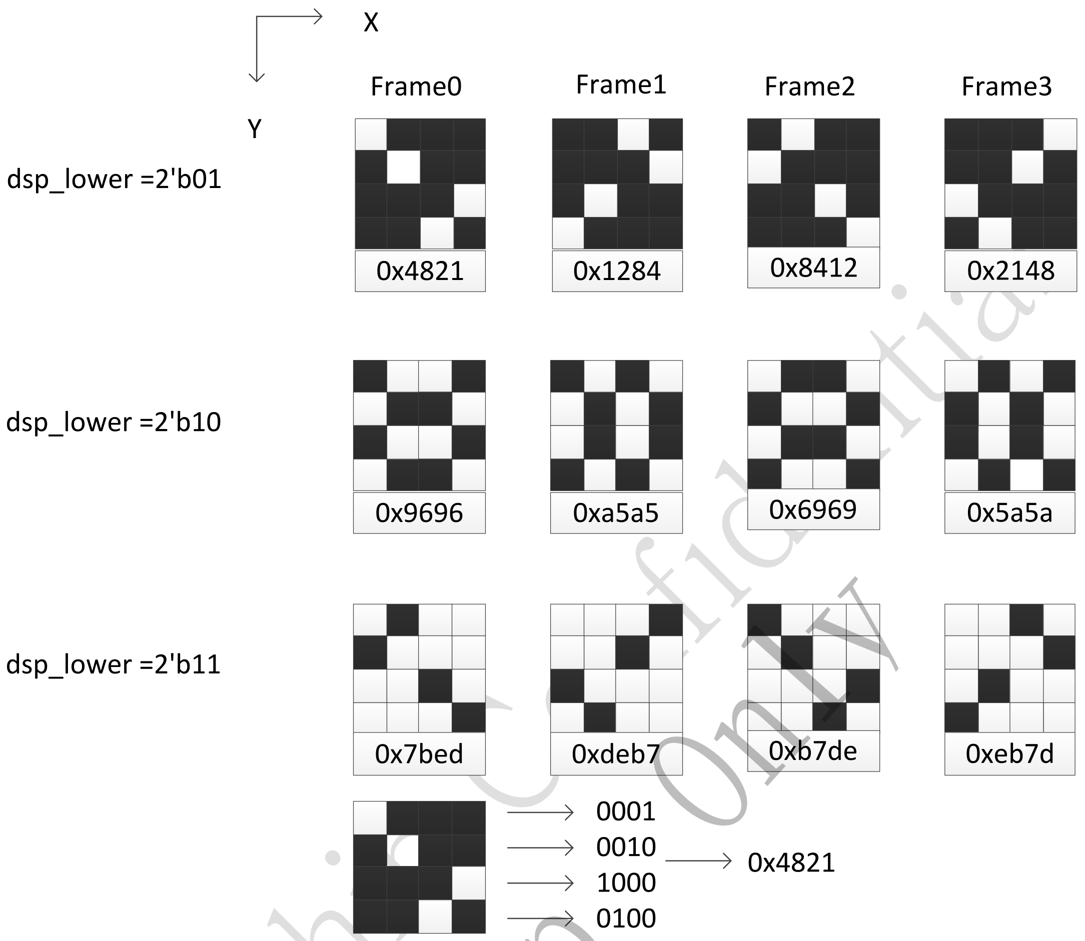


Fig. 27-15 frc pattern diagram

There are four typical pattern as follow ：

(1) default pattern:

0x1e0 : 0x12844821
0x1e4 : 0x21488412
0x1e8 : 0xa55a9696
0x1ec : 0x5aa56969
0x1f0 : 0xdeb77bed
0x1f4 : 0xed7bb7de

(2) for column inversion panel:

0x1e0 : 0x50a00a05
0x1e4 : 0xa050050a
0x1e8 : 0x5a5aa5a5
0x1ec : 0x5a5aa5a5
0x1f0 : 0xaf5ff5fa
0x1f4 : 0x5faffaf5

(3) for 1+2dot panel

0x1e0 : 0x0c308421

```
0x1e4 : 0x124803c0
0x1e8 : 0xcc339669
0x1ec : 0x33cc9669
0x1f0 : 0xf3cf7bde
0x1f4 : 0xedb7fc3f
```

(4) default enhance pattern

```
0x1e0 : 0x12844821
0x1e4 : 0x21488412
0x1e8 : 0x55aaaa55
0x1ec : 0x55aaaa55
0x1f0 : 0xdeb77bed
0x1f4 : 0xed7bb7de
```

10.Gamma Correction

Gamma Correction is necessary because most monitors don't have a linear relationship between the voltage and the brightness, which results in your scene looking like it has too much contrast and the light falling off from the source outward, happens too quickly. The result can also be problematic if you are going into a composition program.

You can correct this by "Gamma Correction", which allows you to display the images and textures on your computer in an accurate manner.

Your screen is not linear, in that it displays the brightness unevenly.As a result, the image looks to be more high contrast than it should, you end up adding more lights or turning up the intensity, or you don't use the lighting in a realistic way that matches well with live action scenes. It also creates problems for you if you use compositing software.

There are three 1024x10bits line buffers sperately for 10bit-R/G/B gamma correction. For 8bit-RGB,it only consumes 256x8bit for each channel.You can write gamma correction LUT through register" GAMMA_LUT_ADDR" one by one.

11.Output format

Config dsp_out_mode register to adapt a variety of panel interface. As follow:

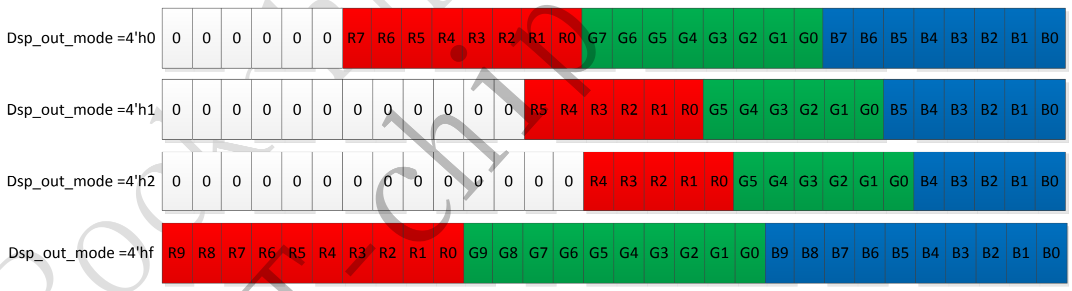

Dsp_out_mode =4'h0 0 0 0 0 0 0 R7 R6 R5 R4 R3 R2 R1 R0 G7 G6 G5 G4 G3 G2 G1 G0 B7 B6 B5 B4 B3 B2 B1 B0
Dsp_out_mode =4'h1 0 0 0 0 0 0 0 0 0 0 0 0 R5 R4 R3 R2 R1 R0 G5 G4 G3 G2 G1 G0 B5 B4 B3 B2 B1 B0
Dsp_out_mode =4'h2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 R4 R3 R2 R1 R0 G5 G4 G3 G2 G1 G0 B4 B3 B2 B1 B0
Dsp_out_mode =4'hf R9 R8 R7 R6 R5 R4 R3 R2 R1 R0 G9 G8 G7 G6 G5 G4 G3 G2 G1 G0 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0

Fig. 27-16 dsp_out_mode description

27.4 Register Description

27.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
VOP_REG_CFG_DONE	0x0000	W	0x00000000	Register config done flag
VOP_VERSION_INFO	0x0004	W	0x00000000	
VOP_SYS_CTRL	0x0008	W	0x00801000	System control register0

Name	Offset	Size	Reset Value	Description
VOP_SYS_CTRL1	0x000c	W	0x00000000	
VOP_DSP_CTRL0	0x0010	W	0x00000000	Display control register0
VOP_DSP_CTRL1	0x0014	W	0x0000e400	Display control register1
VOP_DSP_BG	0x0018	W	0x00000000	background color
VOP_MCU_CTRL	0x001c	W	0x00711c08	MCU mode control register
VOP_INTR_CTRL0	0x0020	W	0x00000000	Interrupt ctrl register0
VOP_INTR_CTRL1	0x0024	W	0x00000000	Interrupt ctrl register1
VOP_INTR_RESERVED0	0x0028	W	0x00000000	
VOP_INTR_RESERVED1	0x002c	W	0x00000000	
VOP_WIN0_CTRL0	0x0030	W	0x00000040	win0 ctrl register0
VOP_WIN0_CTRL1	0x0034	W	0x00000000	win1 ctrl register1
VOP_WIN0_COLOR_KEY	0x0038	W	0x00000000	Win0 color key register
VOP_WIN0_VIR	0x003c	W	0x00000140	Win0 virtual stride
VOP_WIN0_YRGB_MST	0x0040	W	0x00000000	Win0 YRGB memory start address
VOP_WIN0_CBR_MST	0x0044	W	0x00000000	Win0 Cbr memory start address
VOP_WIN0_ACT_INFO	0x0048	W	0x00ef013f	Win0 active window width/height
VOP_WIN0_DSP_INFO	0x004c	W	0x00ef013f	Win0 display width/height on panel
VOP_WIN0_DSP_ST	0x0050	W	0x000a000a	Win0 display start point on panel
VOP_WIN0_SCL_FACTOR_YRGB	0x0054	W	0x10001000	Win0 YRGB scaling factor
VOP_WIN0_SCL_FACTOR_CBR	0x0058	W	0x10001000	Win0 Cbr scaling factor
VOP_WIN0_SCL_OFFSET	0x005c	W	0x00000000	Win0 scaling start point offset
VOP_WIN0_SRC_ALPHA_CTRL	0x0060	W	0x00000000	
VOP_WIN0_DST_ALPHA_CTRL	0x0064	W	0x00000000	
VOP_WIN0_FADING_CTRL	0x0068	W	0x00000000	
VOP_WIN0_RESERVED0	0x006c	W	0x00000000	
VOP_WIN1_CTRL0	0x0070	W	0x00000040	win1 ctrl register0
VOP_WIN1_CTRL1	0x0074	W	0x00000000	win1 ctrl register1

Name	Offset	Size	Reset Value	Description
VOP_WIN1_COLOR_KEY	0x0078	W	0x00000000	Win1 color key register
VOP_WIN1_VIR	0x007c	W	0x00000140	win1 virtual stride
VOP_WIN1_YRGB_MST	0x0080	W	0x00000000	Win1 YRGB memory start address
VOP_WIN1_CBR_MST	0x0084	W	0x00000000	Win1 Cbr memory start address
VOP_WIN1_ACT_INFO	0x0088	W	0x00ef013f	Win1 active window width/height
VOP_WIN1_DSP_INFO	0x008c	W	0x00ef013f	Win1 display width/height on panel
VOP_WIN1_DSP_ST	0x0090	W	0x000a000a	Win1 display start point on panel
VOP_WIN1_SCL_FACTOR_YRGB	0x0094	W	0x10001000	Win1 YRGB scaling factor
VOP_WIN1_SCL_FACTOR_CBR	0x0098	W	0x10001000	Win1 Cbr scaling factor
VOP_WIN1_SCL_OFFSET	0x009c	W	0x00000000	Win1 scaling start point offset
VOP_WIN1_SRC_ALPHA_CTRL	0x00a0	W	0x00000000	
VOP_WIN1_DST_ALPHA_CTRL	0x00a4	W	0x00000000	
VOP_WIN1_FADING_CTRL	0x00a8	W	0x00000000	
VOP_WIN1_RESERVED0	0x00ac	W	0x00000000	
VOP_WIN2_CTRL0	0x00b0	W	0x00000000	win2 ctrl register0
VOP_WIN2_CTRL1	0x00b4	W	0x00000000	win2 ctrl register0
VOP_WIN2_VIR0_1	0x00b8	W	0x01400140	Win2 virtual stride0 and virtaul stride1
VOP_WIN2_VIR2_3	0x00bc	W	0x01400140	Win2 virtual stride2 and virtaul stride3
VOP_WIN2_MST0	0x00c0	W	0x00000000	Win2 memory start address0
VOP_WIN2_DSP_INFO0	0x00c4	W	0x00ef013f	Win2 display width0/height0 on panel
VOP_WIN2_DSP_ST0	0x00c8	W	0x000a000a	Win2 display start point0 on panel
VOP_WIN2_COLOR_KEY	0x00cc	W	0x00000000	Win2 color key register
VOP_WIN2_MST1	0x00d0	W	0x00000000	Win2 memory start address1

Name	Offset	Size	Reset Value	Description
VOP_WIN2_DSP_INFO1	0x00d4	W	0x00ef013f	Win2 display width1/height1 on panel
VOP_WIN2_DSP_ST1	0x00d8	W	0x000a000a	Win2 display start point1 on panel
VOP_WIN2_SRC_ALPHA_CTRL	0x00dc	W	0x00000000	
VOP_WIN2_MST2	0x00e0	W	0x00000000	Win2 memory start address2
VOP_WIN2_DSP_INFO2	0x00e4	W	0x00ef013f	Win2 display width2/height2 on panel
VOP_WIN2_DSP_ST2	0x00e8	W	0x000a000a	Win2 display start point2 on panel
VOP_WIN2_DST_ALPHA_CTRL	0x00ec	W	0x00000000	
VOP_WIN2_MST3	0x00f0	W	0x00000000	Win2 memory start address3
VOP_WIN2_DSP_INFO3	0x00f4	W	0x00ef013f	Win2 display width3/height3 on panel
VOP_WIN2_DSP_ST3	0x00f8	W	0x000a000a	Win2 display start point3 on panel
VOP_WIN2_FADING_CTRL	0x00fc	W	0x00000000	
VOP_WIN3_CTRL0	0x0100	W	0x00000000	win0 ctrl register0
VOP_WIN3_CTRL1	0x0104	W	0x00000000	win0 ctrl register1
VOP_WIN3_VIR0_1	0x0108	W	0x01400140	Win3 virtual stride0 and virtaul stride1
VOP_WIN3_VIR2_3	0x010c	W	0x01400140	Win3 virtual stride2 and virtaul stride3
VOP_WIN3_MST0	0x0110	W	0x00000000	Win3 memory start address0
VOP_WIN3_DSP_INFO0	0x0114	W	0x00ef013f	Win3 display width0/height0 on panel
VOP_WIN3_DSP_ST0	0x0118	W	0x000a000a	Win3 display start point0 on panel
VOP_WIN3_COLOR_KEY	0x011c	W	0x00000000	Win3 color key register
VOP_WIN3_MST1	0x0120	W	0x00000000	Win3 memory start address1

Name	Offset	Size	Reset Value	Description
VOP_WIN3_DSP_INFO1	0x0124	W	0x00ef013f	Win3 display width1/height1 on panel
VOP_WIN3_DSP_ST1	0x0128	W	0x000a000a	Win3 display start point1 on panel
VOP_WIN3_SRC_ALPHA_CTRL	0x012c	W	0x00000000	
VOP_WIN3_MST2	0x0130	W	0x00000000	Win3 memory start address2
VOP_WIN3_DSP_INFO2	0x0134	W	0x00ef013f	Win3 display width2/height2 on panel
VOP_WIN3_DSP_ST2	0x0138	W	0x000a000a	Win3 display start point2 on panel
VOP_WIN3_DST_ALPHA_CTRL	0x013c	W	0x00000000	
VOP_WIN3_MST3	0x0140	W	0x00000000	Win3 memory start address3
VOP_WIN3_DSP_INFO3	0x0144	W	0x00ef013f	Win3 display width3/height3 on panel
VOP_WIN3_DSP_ST3	0x0148	W	0x000a000a	Win3 display start point3 on panel
VOP_WIN3_FADING_CTRL	0x014c	W	0x00000000	
VOP_HWC_CTRL0	0x0150	W	0x00000000	Hwc ctrl register0
VOP_HWC_CTRL1	0x0154	W	0x00000000	Hwc ctrl register1
VOP_HWC_MST	0x0158	W	0x00000000	Hwc memory start address
VOP_HWC_DSP_ST	0x015c	W	0x000a000a	Hwc display start point on panel
VOP_HWC_SRC_ALPHA_CTRL	0x0160	W	0x00000000	
VOP_HWC_DST_ALPHA_CTRL	0x0164	W	0x00000000	
VOP_HWC_FADING_CTRL	0x0168	W	0x00000000	
VOP_HWC_RESERVED1	0x016c	W	0x00000000	
VOP_POST_DSP_HACT_INFO	0x0170	W	0x000a014a	post scaler down horizontal start and end

Name	Offset	Size	Reset Value	Description
VOP_POST_DSP_VACT_INFO	0x0174	W	0x000a00fa	Panel active horizontal scanning start point and end point
VOP_POST_SCL_FACTOR_YRGB	0x0178	W	0x10001000	post yrgb scaling factor
VOP_POST_RESERVED	0x017c	W	0x00001000	
VOP_POST_SCL_CTRL	0x0180	W	0x00000000	post scaling start point offset
VOP_POST_DSP_VACT_INFO_F1	0x0184	W	0x000a00fa	Panel active horizontal scanning start point and end point F1
VOP_DSP_HTOTAL_HS_END	0x0188	W	0x014a000a	Panel scanning horizontal width and hsync pulse end point
VOP_DSP_HACT_ST_END	0x018c	W	0x000a014a	Panel active horizontal scanning start point and end point
VOP_DSP_VTOTAL_VS_END	0x0190	W	0x00fa000a	Panel scanning vertical height and vsync pulse end point
VOP_DSP_VACT_ST_END	0x0194	W	0x000a00fa	Panel active vertical scanning start point and end point
VOP_DSP_VS_ST_END_F1	0x0198	W	0x00000000	Vertical scanning start point and vsync pulse end point of even filed in interlace mode
VOP_DSP_VACT_ST_END_F1	0x019c	W	0x00000000	Vertical scanning active start point and end point of even filed in interlace mode
VOP_PWM_CTRL	0x01a0	W	0x0000200a	PWM Control Register
VOP_PWM_PERIOD_HPR	0x01a4	W	0x00000000	PWM Period Register/High Polarity Capture Register
VOP_PWM_DUTY_LPR	0x01a8	W	0x00000000	PWM Duty Register/Low Polarity Capture Register
VOP_PWM_CNT	0x01ac	W	0x00000000	PWM Counter Register

Name	Offset	Size	Reset Value	Description
VOP_BCSH_COLOR_BAR	0x01b0	W	0x00000000	color bar config register
VOP_BCSH_BCS	0x01b4	W	0xd0010000	brightness contrast saturation*contrast config register
VOP_BCSH_H	0x01b8	W	0x01000000	sin hue and cos hue config register
VOP_BCSH_RESERVED	0x01bc	W	0x00000000	
VOP_CABC_CTRL0	0x01c0	W	0x00000000	
VOP_CABC_CTRL1	0x01c4	W	0x00000000	
VOP_CABC_GAUSS_LINE0_0	0x01c8	W	0x15110903	Register0000 Abstract
VOP_CABC_GAUSS_LINE0_1	0x01cc	W	0x00030911	Register0001 Abstract
VOP_CABC_GAUSS_LINE1_0	0x01d0	W	0x1a150b04	Register0002 Abstract
VOP_CABC_GAUSS_LINE1_1	0x01d4	W	0x00040b15	Register0003 Abstract
VOP_CABC_GAUSS_LINE2_0	0x01d8	W	0x15110903	Register0004 Abstract
VOP_CABC_GAUSS_LINE2_1	0x01dc	W	0x00030911	Register0005 Abstract
VOP_FRC_LOWER01_0	0x01e0	W	0x12844821	
VOP_FRC_LOWER01_1	0x01e4	W	0x21488412	
VOP_FRC_LOWER10_0	0x01e8	W	0xa55a9696	
VOP_FRC_LOWER10_1	0x01ec	W	0x5aa56969	
VOP_FRC_LOWER11_0	0x01f0	W	0xdeb77bed	
VOP_FRC_LOWER11_1	0x01f4	W	0xed7bb7de	
VOP_FRC_RESERVED0	0x01f8	W	0x00000000	
VOP_FRC_RESERVED1	0x01fc	W	0x00000000	
VOP_MMU_DTE_ADDR	0x0300	W	0x00000000	MMU current page Table address
VOP_MMU_STATUS	0x0304	W	0x00000000	MMU status register
VOP_MMU_COMMAND	0x0308	W	0x00000000	MMU command register
VOP_MMU_PAGE_FAULT_ADDR	0x030c	W	0x00000000	MMU logical address of last page fault
VOP_MMU_ZAP_ONE_LINE	0x0310	W	0x00000000	MMU Zap cache line register
VOP_MMU_INT_RAWSTAT	0x0314	W	0x00000000	MMU raw interrupt status register
VOP_MMU_INT_CLEAR	0x0318	W	0x00000000	MMU raw interrupt status register

Name	Offset	Size	Reset Value	Description
VOP_MMU_INT_MASK	0x031c	W	0x00000000	MMU raw interrupt status register
VOP_MMU_INT_STATUS	0x0320	W	0x00000000	MMU raw interrupt status register
VOP_MMU_AUTO_GATING	0x0324	W	0x00000000	mmu auto gating
VOP_WIN2_LUT_ADDR	0x0400	W	0x00000000	
VOP_WIN3_LUT_ADDR	0x0800	W	0x00000000	
VOP_HWC_LUT_ADDR	0x0c00	W	0x00000000	
VOP_GAMMA_LUT_ADDR	0x1000	W	0x00000000	
VOP_MCU_BYPASS_WPORT	0x2200	W	0x00000000	Register0000 Abstract
VOP_MCU_BYPASS_RPORT	0x2300	W	0x00000000	Register0001 Abstract

*Notes: Size : **B** - Byte (8 bits) access, **HW** - Half WORD (16 bits) access, **W** -WORD (32 bits) access*

27.4.2 Detail Register Description

VOP_REG_CFG_DONE
Address: Operational Base + offset (0x0000)
Register config done flag

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	WO	0x0	reg_load_en lcdc register config done flag In the first setting of the register, the new value was saved into the mirror register. When all the register config finish, writing this register to enable the copyright of the mirror register to real register. Then register would be updated at the start of every frame.

VOP_VERSION_INFO
Address: Operational Base + offset (0x0004)

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	fpga_version
15:0	RW	0x0000	rtl_version

VOP_SYS_CTRL
Address: Operational Base + offset (0x0008)
System control register0

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved

Bit	Attr	Reset Value	Description
23	RW	0x1	auto_gating_en LCDC layer axi-clk auto gating enable 1'b0 : disable auto gating 1'b1 : enable auto gating default auto gating enable
22	RW	0x0	vop_standby_en LCDC standby mode Writing "1" to turn LCDC into standby mode, All the layer would disable and the data transfer from frame buffer memory would stop at the end of current frame. The output would be blank. When writing "0" to this bit, standby mode would disable and the LCDC go back to work immediately. 1'b0 : disable 1'b1 : enable * Black display is recommended before setting standby mode enable.
21	RW	0x0	vop_dma_stop LCDC DMA stop mode 1'b0 : disable 1'b1 : enable * If DMA is working, the stop mode would not be active until current bus transfer is finished.
20	RW	0x0	vop_mmu_en vop mmu enable signal 1'b0 : bypass mmu 1'b1 : enable mmu
19:18	RW	0x0	dma_burst_length DMA read Burst length 2'b00 : burst16 (burst 15 in rgb888 pack mode) 2'b01 : burst8 (burst 12 in rgb888 pack mode) 2'b10 : burst4 (burst 6 in rgb888 pack mode)
17:16	RO	0x0	reserved
15	RW	0x0	mipi_out_en 1'b0 : gating output clk ,data and control signal 1'b1 : mipi interface enable
14	RW	0x0	edp_out_en 1'b0 : gating output clk ,data and control signal 1'b1 : edp interface enable
13	RW	0x0	hdmi_out_en 1'b0 : gating output clk ,data and control signal 1'b1 : hdmi interface enable

Bit	Attr	Reset Value	Description
12	RW	0x1	rgb_out_en 1'b0 : gating output clk ,data and control signal 1'b1 : rgb/lvds interface enable
11	RO	0x0	reserved
10	RW	0x0	edpi_wms_fs edpi wms mode ，rame st signal write 禈: edpi_wms_mode frame start (when other register is config done) read : wms mode hold status
9	RW	0x0	edpi_wms_mode 1'b1: mipi command mode
8	RW	0x0	edpi_halt_en mipi flow ctrl enable
7:4	RW	0x0	doub_ch_overlap_num 4'h0: overlap num 0 4'h1: overlap num 2 4'h2: overlap num 4 4'h3: overlap num 6 4'h4: overlap num 8 4'h5: overlap num 10 4'h6: overlap num 12 4'h7: overlap num 14 4'h8: overlap num 16
3	RW	0x0	doub_channel_en mipi double channel enable
2:1	RW	0x0	direct_path_layer_sel direct path layer select 2'b00 : select win0 2'b01 : select win1 2'b10 : select win2 2'b11 : select win3
0	RW	0x0	direct_path_en iep direct path enable signal 1'b0 : disable iep direct path 1'b1 : enable iep direct path

VOP_SYS_CTRL1
Address: Operational Base + offset (0x000c)

Bit	Attr	Reset Value	Description
31:18	RO	0x0	reserved
17:13	RW	0x00	axi_outstanding_max_num

Bit	Attr	Reset Value	Description
12	RW	0x0	axi_max_outstanding_en
11:10	RW	0x0	noc_win_qos
9	RW	0x0	noc_qos_en
8:3	RW	0x00	noc_hurry_threshold
2:1	RW	0x0	noc_hurry_value
0	RW	0x0	noc_hurry_en

VOP_DSP_CTRL0
Address: Operational Base + offset (0x0010)
Display control register0

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23	RW	0x0	dsp_y_mir_en 1'b0 : no y_mirror 1'b1 : y_mirror
22	RW	0x0	dsp_x_mir_en 1'b0 : no x_mirror 1'b1 : x_mirror
21	RW	0x0	dsp_yuv_clip YCrCb clip 1'b0 : disable, YCbCr no clip 1'b1 : enable, YCbCr clip before YCbCr2RGB10bit *Y clip: 64~940, CbCr clip: 64~960
20	RW	0x0	dsp_ccir656_avg Cb-Cr filter in CCIR656 mode 1'b0 : drop mode 1'b1 : average mode
19	RW	0x0	dsp_black_en Black display mode When this bit enable, the pixel data output is all black (0x000000)
18	RW	0x0	dsp_blank_en Blank display mode When this bit enable, the Hsync/Vsync/Den output is blank

Bit	Attr	Reset Value	Description
17	RW	0x0	dsp_out_zero Hsync/Vsync/Den output software ctrl 1'b0 : normal output 1'b1 : all output '0'
16	RW	0x0	dsp_dummy_swap Display dummy swap enable 1'b0 : B+G+R+dummy 1'b1 : dummy+B+G+R
15	RW	0x0	dsp_delta_swap Display delta swap enable 1'b0 : disable 1'b1 : enable *See detail description in Delta display charpter.
14	RW	0x0	dsp_rg_swap Display output red and green swap enable 1'b0 : RGB 1'b1 : GRB
13	RW	0x0	dsp_rb_swap Display output red and blue swap enable 1'b0 : RGB 1'b1 : BGR
12	RW	0x0	dsp_bg_swap Display output blue and green swap enable 1'b0 : RGB 1'b1 : RBG
11	RW	0x0	dsp_field_pol field polarity when interlace dsp 1'b0 : normal 1'b1 : invert
10	RW	0x0	dsp_interlace Interlace display enable 1'b0 : disable 1'b1 : enable *This mode is related to the ITU-R656 output, the display timing of odd field must be set correctly. (lcdc_dsp_vs_st_end_f1/lcdc_dsp_vact_end_f1)
9	RW	0x0	dsp_ddr_phase dclk phase lock 1'b0 : no lock 1'b1 : lock every line
8	RW	0x0	dsp_dclk_ddr dclk output mode 1'b0 : SDR 1'b1 : DDR

Bit	Attr	Reset Value	Description
7	RW	0x0	dsp_dclk_pol DCLK invert enable 1'b0 : normal 1'b1 : invert default dclk invert
6	RW	0x0	dsp_den_pol DEN polarity 1'b0 : positive 1'b1 : negative
5	RW	0x0	dsp_vsync_pol VSYNC polarity 1'b0 : negative 1'b1 : positive
4	RW	0x0	dsp_hsync_pol HSYNC polarity 1'b0 : negative 1'b1 : positive
3:0	RW	0x0	dsp_out_mode Display output format 4'b0000: Parallel 24-bit RGB888 output R[7:0],G[7:0],B[7:0] 4'b0001: Parallel 18-bit RGB666 output 6'b0,R[5:0],G[5:0],B[5:0] 4'b0010: Parallel 16-bit RGB565 output 8'b0,R[4:0],G[5:0],B[4:0] 4'b0011: Parallel 24-bit RGB888 double pixel mix out phase0:G1[3:0],B1[7:0],G0[3:0],B0[7:0] phase1:R1[7:0],G1[7:4],R0[7:0],G0[7:4] 4'b0100: Serial 2x12-bit 12'b0,G[3:0],B[7:0] + 12'b0,R[7:0],G[7:4] 4'b0101: ITU-656 output mode0 16'b0,pixel_data[7:0] 4'b0110: ITU-656 output mode1 8'b0,pixel_data[7:0],8'b0 4'b0111: ITU-656 output mode2 9'b0,pixel_data[7:0],7'b0 4'b1000: Serial 3x8-bit RGB888 16'b0, B[7:0]+16'b0,G[7:0]+16'b0,R[7:0] 4'b1100: Serial 3x8-bit RGB888 + dummy 16'b0, B[7:0]+16'b0,G[7:0]+16'b0,R[7:0] + dummy 4'b1111: Parallel 30-bit RGBaaa output R[9:0],G[9:0],B[9:0] Others: Reserved.

VOP_DSP_CTRL1

Address: Operational Base + offset (0x0014)
Display control register1

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:14	RW	0x3	dsp_layer3_sel
13:12	RW	0x2	dsp_layer2_sel
11:10	RW	0x1	dsp_layer1_sel
9:8	RW	0x0	dsp_layer0_sel
7	RO	0x0	reserved
6	RW	0x0	dither_up_en
5	RO	0x0	reserved
4	RW	0x0	dither_down_sel dither down mode select 2'b0 : allegro 2'b1 : FRC
3	RW	0x0	dither_down_mode Dither-down mode 1'b0 : RGB888 to RGB565 1'b1 : RGB888 to RGB666
2	RW	0x0	dither_down_en Dither-down enable 1'b0 : disable 1'b1 : enable
1	RW	0x0	pre_dither_down_en 10bit -> 8bit (allegro)
0	RW	0x0	dsp_lut_en Display LUT ram enable 1'b0 : disable 1'b1 : enable *This bit should be "0" when CPU updates the LUT, and should be "1" when Display LUT mode enable.

VOP_DSP_BG
Address: Operational Base + offset (0x0018)
background color

Bit	Attr	Reset Value	Description
31:30	RO	0x0	reserved

Bit	Attr	Reset Value	Description
29:20	RW	0x000	dsp_bg_red Background Red color
19:10	RW	0x000	dsp_bg_green Background Green color
9:0	RW	0x000	dsp_bg_blue Background Blue color

VOP_MCU_CTRL
Address: Operational Base + offset (0x001c)
MCU mode control register

Bit	Attr	Reset Value	Description
31	RW	0x0	mcu_type MCU LCD output SELECT
30	RW	0x0	mcu_bypass MCU LCD BYPASS MODE Select
29	RW	0x0	mcu_rs MCU LCD RS Select
28	RW	0x0	muc_frame_st Write"1" : MCU HOLD Mode Frame Start Read : MCU/LCDC standby HOLD status
27	RW	0x0	mcu_hold_mode MCU HOLD Mode Select
26	RW	0x0	mcu_clk_sel MCU_CLK_SEL for MCU bypass 1'b1 : MCU BYPASS sync with DCLK 1'b0 : MCU BYPASS sync with HCLK
25:20	RW	0x07	mcu_rw_pend MCU_RW signal end point (0-63)
19:16	RW	0x1	mcu_rw_pst MCU_RW signal start point (0-15)
15:10	RW	0x07	mcu_cs_pend MCU_CS signal end point (0-63)
9:6	RW	0x0	mcu_cs_pst MCU_CS signal start point (0-15)

Bit	Attr	Reset Value	Description
5:0	RW	0x08	mcu_pix_total MCU LCD Interface writing period (1-63)

VOP_INTR_CTRL0
Address: Operational Base + offset (0x0020)
Interrupt ctrl register0

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24:12	RW	0x0000	dsp_line_frag_num Line number of the Line flag interrupt The display line number when the flag interrupt occur, the range is (0~ DSP_VTOTAL-1).
11	WO	0x0	bus_error_intr_clr Bus error Interrupt clear (Auto clear)
10	WO	0x0	line_frag_intr_clr Line flag Interrupt clear (Auto clear)
9	WO	0x0	fs_intr_clr Frame start interrupt clear (Auto clear)
8	WO	0x0	dsp_hold_valid_intr_clr display hold valid interrupt clear (Auto clear)
7	RW	0x0	bus_error_intr_en Bus error interrupt enable 1'b0 : disable 1'b1 : enable
6	RW	0x0	line_frag_intr_en Line flag Interrupt enable 1'b0 : disable 1'b1 : enable
5	RW	0x0	fs_intr_en Frame start interrupt enable 1'b0 : disable 1'b1 : enable
4	RW	0x0	dsp_hold_valid_intr_en display hold valid interrupt enable 1'b0 : disable 1'b1 : enable
3	RO	0x0	bus_error_intr_sts Bus error Interrupt status

Bit	Attr	Reset Value	Description
2	RO	0x0	line_frag_intr_sts Line flag Interrupt status
1	RO	0x0	fs_intr_sts Frame start interrupt status
0	RO	0x0	dsp_hold_valid_intr_sts display hold valid interrupt status

VOP_INTR_CTRL1
Address: Operational Base + offset (0x0024)
Interrupt ctrl register1

Bit	Attr	Reset Value	Description
31:23	RO	0x0	reserved
22	RW	0x0	pwm_gen_intr_clr
21	RW	0x0	post_buf_empty_intr_clr post line buffer empty interrupt clear(auto clear)
20	RW	0x0	hwc_empty_intr_clr hwc data empty interrupt clear(auto clear)
19	RW	0x0	win3_empty_intr_clr win3 data empty interrupt clear(auto clear)
18	RW	0x0	win2_empty_intr_clr win2 data empty interrupt clear(auto clear)
17	RW	0x0	win1_empty_intr_clr win1 data empty interrupt clear(auto clear)
16	W1C	0x0	win0_empty_intr_clr win0 data empty interrupt clear(auto clear)
15	RO	0x0	reserved
14	RW	0x0	pwm_gen_intr_en
13	RW	0x0	post_buf_empty_intr_en post line buffer empty interrupt enable signal 1'b0 : disable 1'b1 : enable

Bit	Attr	Reset Value	Description
12	RW	0x0	hwc_empty_intr_en hwc data empty interrupt enable signal 1'b0 : disable 1'b1 : enable
11	RW	0x0	win3_empty_intr_en win3 data empty interrupt enable signal 1'b0 : disable 1'b1 : enable
10	RW	0x0	win2_empty_intr_en win2 data empty interrupt enable signal 1'b0 : disable 1'b1 : enable
9	RW	0x0	win1_empty_intr_en win1 data empty interrupt enable signal 1'b0 : disable 1'b1 : enable
8	RW	0x0	win0_empty_intr_en win0 data empty interrupt enable signal 1'b0 : disable 1'b1 : enable
7	RO	0x0	reserved
6	RW	0x0	pwm_gen_intr_sts pwm generated interrupt 0: Channel 0 Interrupt not generated 1: Channel 0 Interrupt generated
5	RW	0x0	post_buf_empty_intr_sts post buffer empty interrupt status
4	RW	0x0	hwc_empty_intr_sts hwc data empty interrupt status
3	RW	0x0	win3_empty_intr_sts win3 data empty interrupt status
2	RW	0x0	win2_empty_intr_sts win2 data empty interrupt status
1	RW	0x0	win1_empty_intr_sts win1 data empty interrupt status
0	RO	0x0	win0_empty_intr_sts win0 data empty interrupt status

VOP_INTR_RESERVED0

Address: Operational Base + offset (0x0028)

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	reserved

VOP_INTR_RESERVED1
Address: Operational Base + offset (0x002c)

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	reserved

VOP_WIN0_CTRL0
Address: Operational Base + offset (0x0030)
win0 ctrl register0

Bit	Attr	Reset Value	Description
31:21	RO	0x0	reserved
20	RW	0x0	win0_yuv_clip YCrCb clip 1'b0 : disable, YCbCr no clip 1'b1 : enable, YCbCr clip before YCbCr2RGB *Y clip: 16~235, CbCr clip: 16~239
19	RW	0x0	win0_cbr_deflick Win0 Cbr deflick mode 1'b0 : disable 1'b1 : enable
18	RW	0x0	win0_yrgb_deflick win0 YRGB deflick mode 1'b0 : disable 1'b1 : enable
17	RO	0x0	reserved

16	RW	0x0	win0_ppas_zero_en
			0:per_pix_alpha+scale,pix not change;
			1:per_pix_alpha_scale,pix=0 when alpha=0;

15	RW	0x0	win0_uv_swap Win0 CbCr swap 1'b0 : CrCb 1'b1 : CbCr
14	RW	0x0	win0_mid_swap Win0 Y middle swap 1'b0 : Y3Y2Y1Y0 1'b1 : Y3Y1Y2Y0

Bit	Attr	Reset Value	Description
13	RW	0x0	win0_alpha_swap win0 alpha swap 1'b0 : ARGB 1'b1 : RGBA
12	RW	0x0	win0_rb_swap win0 RGB RED and BLUE swap 1'b0 : RGB 1'b1 : BGR
11:10	RW	0x0	win0_csc_mode Win0 YUV2RGB Color space conversion: 2'b00/01 : mpeg 2'b10 : jpeg 2'b11 : hd
9	RW	0x0	win0_no_outstanding win0 AXI master read outstanding 1'b0 : enable 1'b1 : disable
8	RW	0x0	win0_interlace_read Win0 interlace read mode 1'b0 : disable 1'b1 : enable
7:5	RW	0x2	win0_lb_mode
4	RW	0x0	win0_fmt_10 0: yuv 8bit fmt mode 1: yuv 10bit fmt mode
3:1	RW	0x0	win0_data_fmt 3'b000 : ARGB888 3'b001 : RGB888 3'b010 : RGB565 3'b100 : YcbCr420 3'b101 : YcbCr422 3'b110 : YcbCr444
0	RW	0x0	win0_en

VOP_WIN0_CTRL1
Address: Operational Base + offset (0x0034)
win1 ctrl register1

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31	RW	0x0	win0_cbr_vsd_mode win0 vertical scaler down mode select 1'b0 : bilinear 1'b1 : average
30	RW	0x0	win0_cbr_vsu_mode win0 vertical scaler down mode select 1'b0 : bilinear 1'b1 : bicubic
29:28	RW	0x0	win0_cbr_hsd_mode win0 horizontal scaler down mode select 2'b00 : bilinear 2'b01 : bicubic 2'b10 : average
27:26	RW	0x0	win0_cbr_ver_scl_mode
25:24	RW	0x0	win0_cbr_hor_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale
23	RW	0x0	win0_yrgb_vsd_mode win0 vertical scaler down mode select 1'b0 : bilinear 1'b1 : average
22	RW	0x0	win0_yrgb_vsu_mode win0 vertical scaler down mode select 1'b0 : bilinear 1'b1 : bicubic
21:20	RW	0x0	win0_yrgb_hsd_mode win0 horizontal scaler down mode select 2'b00 : bilinear 2'b01 : average
19:18	RW	0x0	win0_yrgb_ver_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale
17:16	RW	0x0	win0_yrgb_hor_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale

Bit	Attr	Reset Value	Description
15	RW	0x0	win0_line_load_mode when yuv fmt,if 1'b0: load data by axi trans 1'b1: load data by lines
14:12	RW	0x0	win0_cbr_axi_gather_num
11:8	RW	0x0	win0_yrgb_axi_gather_num
7	RW	0x0	win0_vsd_cbr_gt2
6	RW	0x0	win0_vsd_cbr_gt4
5	RW	0x0	win0_vsd_yrgb_gt2
4	RW	0x0	win0_vsd_yrgb_gt4
3:2	RW	0x0	win0_bic_coe_sel 2'b00 : PRECISE 2'b01 : SPLINE 2'b10 : CATROM 2'b11 : MITCHELL
1	RW	0x0	win0_cbr_axi_gather_en
0	RW	0x0	win0_yrgb_axi_gather_en

VOP_WIN0_COLOR_KEY
Address: Operational Base + offset (0x0038)
Win0 color key register

Bit	Attr	Reset Value	Description

31	RW	0x0	win0_key_en
			Win0 transparency color key enable
			1'b0 : disable;
			1'b1 : enable;

30	RO	0x0	reserved
29:0	RW	0x00000000	win0_key_color Win0 key color

VOP_WIN0_VIR
Address: Operational Base + offset (0x003c)
Win0 virtual stride

Bit	Attr	Reset Value	Description
31:30	RO	0x0	reserved

Bit	Attr	Reset Value	Description
29:16	RW	0x0000	win0_vir_stride_uv
15:14	RO	0x0	reserved
13:0	RW	0x0140	win0_vir_stride Win0 Virtual stride Number of words of Win0 Virtual width ARGB888 : win0_vir_width RGB888 : (win0_vir_width*3/4) + (win0_vir_width%3) RGB565 : ceil(win0_vir_width/2) YUV : ceil(win0_vir_width/4)

VOP_WIN0_YRGB_MST
Address: Operational Base + offset (0x0040)
Win0 YRGB memory start address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win0_yrgb_mst win0 YRGB frame buffer memory start address

VOP_WIN0_CBR_MST
Address: Operational Base + offset (0x0044)
Win0 Cbr memory start address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win0_cbr_mst win0 CBR frame buffer memory start address

VOP_WIN0_ACT_INFO
Address: Operational Base + offset (0x0048)
Win0 active window width/height

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x00ef	win0_act_height Win0 active(original) window height win_act_height = (win0 vertical size -1)
15:13	RO	0x0	reserved
12:0	RW	0x013f	win0_act_width Win0 active(original) window width win_act_width = (win0 horizontial size -1)

VOP_WIN0_DSP_INFO
Address: Operational Base + offset (0x004c)
Win0 display width/height on panel

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win0_dsp_height Win0 display window height win0_dsp_height = (win0 vertical size -1)
15:12	RO	0x0	reserved
11:0	RW	0x13f	win0_dsp_width Win0 display window width win0_dsp_width = (win0 horizontial size -1)

VOP_WIN0_DSP_ST
Address: Operational Base + offset (0x0050)
Win0 display start point on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win0_dsp_yst Win0 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win0_dsp_xst Win0 horizontal start point(x) of the Panel scanning

VOP_WIN0_SCL_FACTOR_YRGB
Address: Operational Base + offset (0x0054)
Win0 YRGB scaling factor

Bit	Attr	Reset Value	Description
31:16	RW	0x1000	win0_vs_factor_yrgb Win0 YRGB vertical scaling factor: factor=((LCDC_WIN0_ACT_INFO[31:16]) /(LCDC_WIN0_DSP_INFO[31:16]))*2^12
15:0	RW	0x1000	win0_hs_factor_yrgb Win0 YRGB horizontal scaling factor: factor=((LCDC_WIN0_ACT_INFO[15:0]) /(LCDC_WIN0_DSP_INFO[15:0]))*2^12

VOP_WIN0_SCL_FACTOR_CBR
Address: Operational Base + offset (0x0058)
Win0 Cbr scaling factor

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:16	RW	0x1000	win0_vs_factor_cbr Win0 CBR vertical scaling factor: YCbCr420: factor=((LCDC_WIN0_ACT_INFO[31:16]/ 2) /(LCDC_WIN0_DSP_INFO[31:16]))*2^12 YCbCr422,YCbCr444: factor=((LCDC_WIN0_ACT_INFO[31:16]) /(LCDC_WIN0_DSP_INFO[31:16]))*2^12
15:0	RW	0x1000	win0_hs_factor_cbr Win0 CBR horizontal scaling factor: YCbCr422,YCbCr420: factor=((LCDC_WIN0_ACT_INFO[15:0]/2) /(LCDC_WIN0_DSP_INFO[15:0]))*2^12 YCbCr444: factor=((LCDC_WIN0_ACT_INFO[15:0]) /(LCDC_WIN0_DSP_INFO[15:0]))*2^12

VOP_WIN0_SCL_OFFSET
Address: Operational Base + offset (0x005c)
Win0 scaling start point offset

Bit	Attr	Reset Value	Description
31:24	RW	0x00	win0_vs_offset_cbr Cbr Vertical scaling start point offset (0x00~0xff)/0x100 = 0~0.99
23:16	RW	0x00	win0_vs_offset_yrgb Y Vertical scaling start point offset (0x00~0xff)/0x100 = 0~0.99
15:8	RW	0x00	win0_hs_offset_cbr Cbr Horizontal scaling start point offset (0x00~0xff)/0x100 = 0~0.99
7:0	RW	0x00	win0_hs_offset_yrgb Y Horizontal scaling start point offset (0x00~0xff)/0x100 = 0~0.99

VOP_WIN0_SRC_ALPHA_CTRL
Address: Operational Base + offset (0x0060)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	win0_fading_value
23:16	RW	0x00	win0_src_global_alpha layer0 src global alpha （eused by fading value）
15:9	RO	0x0	reserved

Bit	Attr	Reset Value	Description
8:6	RW	0x0	win0_src_factor_m0
5	RW	0x0	win0_src_alpha_cal_m0
4:3	RW	0x0	win0_src_blend_m0
2	RW	0x0	win0_src_alpha_m0
1	RW	0x0	win0_src_color_m0
0	RW	0x0	win0_src_alpha_en

VOP_WIN0_DST_ALPHA_CTRL
Address: Operational Base + offset (0x0064)

Bit	Attr	Reset Value	Description
31:9	RO	0x0	reserved
8:6	RW	0x0	win0_dst_factor_m0
5:0	RW	0x00	win0_dst_m0_reserved

VOP_WIN0_FADING_CTRL
Address: Operational Base + offset (0x0068)

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	layer0_fading_en
23:16	RW	0x00	layer0_fading_offset_b
15:8	RW	0x00	layer0_fading_offset_g
7:0	RW	0x00	layer0_fading_offset_r

VOP_WIN0_RESERVED0
Address: Operational Base + offset (0x006c)

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	reserved

VOP_WIN1_CTRL0
Address: Operational Base + offset (0x0070)
win1 ctrl register0

Bit	Attr	Reset Value	Description
31:21	RO	0x0	reserved
20	RW	0x0	win1_yuv_clip YCrCb clip 1'b0 : disable, YCbCr no clip 1'b1 : enable, YCbCr clip before YCbCr2RGB *Y clip: 16~235, CbCr clip: 16~239
19	RW	0x0	win1_cbr_deflick Win1 Cbr deflick mode 1'b0 : disable 1'b1 : enable
18	RW	0x0	win1_yrgb_deflick win1 YRGB deflick mode 1'b0 : disable 1'b1 : enable
17	RO	0x0	reserved

16	RW	0x0	win1_ppas_zero_en
			0:per_pix_alpha+scale,pix not change;
			1:per_pix_alpha_scale,pix=0 when alpha=0;

15	RW	0x0	win1_uv_swap Win1 CbCr swap 1'b0 : CrCb 1'b1 : CbCr
14	RW	0x0	win1_mid_swap Win1 Y middle 8-bit swap 1'b0 : Y3Y2Y1Y0 1'b1 : Y3Y1Y2Y0
13	RW	0x0	win1_alpha_swap win1 alpha swap 1'b0 : ARGB 1'b1 : RGBA
12	RW	0x0	win1_rb_swap win1 RGB RED and BLUE swap 1'b0 : RGB 1'b1 : BGR
11:10	RW	0x0	win1_csc_mode Win1 YUV2RGB Color space conversion: 2'b00/01 : mpeg 2'b10 : jpeg 2'b11 : hd

Bit	Attr	Reset Value	Description
9	RW	0x0	win1_no_outstanding win1 AXI master read outstanding 1'b0 : enable 1'b1 : disable
8	RW	0x0	win1_interlace_read Win1 interlace read mode 1'b0 : disable 1'b1 : enable
7:5	RW	0x2	win1_lb_mode
4	RW	0x0	win1_fmt_10 0: yuv 8bit fmt mode 1: yuv 10bit fmt mode
3:1	RW	0x0	win1_data_fmt 3'b000 : ARGB888 3'b001 : RGB888 3'b010 : RGB565 3'b100 : YcbCr420 3'b101 : YcbCr422 3'b110 : YcbCr444
0	RW	0x0	win1_en

VOP_WIN1_CTRL1
Address: Operational Base + offset (0x0074)
win1 ctrl register1

Bit	Attr	Reset Value	Description
31	RW	0x0	win1_cbr_vsd_mode win1 vertical scaler down mode select 1'b0 : bilinear 1'b1 : average
30	RW	0x0	win1_cbr_vsu_mode win1 vertical scaler up mode select 1'b0 : bilinear 1'b1 : bicubic
29:28	RW	0x0	win1_cbr_hsd_mode win1 horizontal scaler down mode select 2'b00 : bilinear 2'b01 : bicubic 2'b10 : average

Bit	Attr	Reset Value	Description
27:26	RW	0x0	win1_cbr_ver_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale
25:24	RW	0x0	win1_cbr_hor_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale
23	RW	0x0	win1_yrgb_vsd_mode win1 vertical scaler down mode select 1'b0 : bilinear 1'b1 : average
22	RW	0x0	win1_yrgb_vsu_mode win1 vertical scaler up mode select 1'b0 : bilinear 1'b1 : bicubic
21:20	RW	0x0	win1_yrgb_hsd_mode win1 horizontal scaler down mode select 2'b00 : bilinear 2'b01 : average
19:18	RW	0x0	win1_yrgb_ver_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale
17:16	RW	0x0	win1_yrgb_hor_scl_mode 2'b00 : no scale 2'b01 : scale up 2'b10 : scale down 2'b11 : no scale
15	RW	0x0	win1_line_load_mode when yuv fmt,if 1'b0: load data by pixels 1'b1: load data by lines
14:12	RW	0x0	win1_cbr_axi_gather_num
11:8	RW	0x0	win1_yrgb_axi_gather_num
7	RW	0x0	win1_vsd_cbr_gt2
6	RW	0x0	win1_vsd_cbr_gt4

Bit	Attr	Reset Value	Description
5	RW	0x0	win1_vsd_yrgb_gt2
4	RW	0x0	win1_vsd_yrgb_gt4
3:2	RW	0x0	win1_bic_coe_sel 2'b00 : PRECISE 2'b01 : SPLINE 2'b10 : CATROM 2'b11 : MITCHELL
1	RW	0x0	win1_cbr_axi_gather_en
0	RW	0x0	win1_yrgb_axi_gather_en

VOP_WIN1_COLOR_KEY
Address: Operational Base + offset (0x0078)
Win1 color key register

Bit	Attr	Reset Value	Description

31	RW	0x0	win1_key_en
			Win1 transparency color key enable
			1'b0 : disable;
			1'b1 : enable;

30	RO	0x0	reserved
29:0	RW	0x00000000	win1_key_color Win1 key color

VOP_WIN1_VIR
Address: Operational Base + offset (0x007c)
win1 virtual stride

Bit	Attr	Reset Value	Description
31:30	RO	0x0	reserved
29:16	RW	0x0000	win1_vir_stride_uv
15:14	RO	0x0	reserved
13:0	RW	0x0140	win1_vir_stride Win1 Virtual stride Number of words of Win1 Virtual width ARGB888 : win1_vir_width RGB888 : (win1_vir_width*3/4) + (win1_vir_width%3) RGB565 : ceil(win1_vir_width/2) YUV : ceil(win1_vir_width/4)

VOP_WIN1_YRGB_MST
Address: Operational Base + offset (0x0080)
Win1 YRGB memory start address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win1_yrgb_mst win1 YRGB frame buffer memory start address

VOP_WIN1_CBR_MST
Address: Operational Base + offset (0x0084)
Win1 Cbr memory start address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win1_cbr_mst win1 CBR frame buffer memory start address

VOP_WIN1_ACT_INFO
Address: Operational Base + offset (0x0088)
Win1 active window width/height

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x00ef	win1_act_height Win1 active(original) window height win_act_height = (win1 vertical size -1)
15:13	RO	0x0	reserved
12:0	RW	0x013f	win1_act_width Win1 active(original) window width win_act_width = (win1 horizontial size -1)

VOP_WIN1_DSP_INFO
Address: Operational Base + offset (0x008c)
Win1 display width/height on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win1_dsp_height Win1 display window height win1_dsp_height = (win1 vertical size -1)
15:12	RO	0x0	reserved
11:0	RW	0x13f	win1_dsp_width Win1 display window width win1_dsp_width = (win1 horizontial size -1)

VOP_WIN1_DSP_ST
Address: Operational Base + offset (0x0090)
Win1 display start point on panel

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win1_dsp_yst Win1 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win1_dsp_xst Win1 horizontal start point(x) of the Panel scanning

VOP_WIN1_SCL_FACTOR_YRGB
Address: Operational Base + offset (0x0094)
Win1 YRGB scaling factor

Bit	Attr	Reset Value	Description
31:16	RW	0x1000	win1_vs_factor_yrgb Win1 YRGB vertical scaling factor: factor=((LCDC_WIN1_ACT_INFO[31:16]) /(LCDC_WIN1_DSP_INFO[31:16]))*2^12
15:0	RW	0x1000	win1_hs_factor_yrgb Win1 YRGB horizontal scaling factor: factor=((LCDC_WIN1_ACT_INFO[15:0]) /(LCDC_WIN1_DSP_INFO[15:0]))*2^12

VOP_WIN1_SCL_FACTOR_CBR
Address: Operational Base + offset (0x0098)
Win1 Cbr scaling factor

Bit	Attr	Reset Value	Description
31:16	RW	0x1000	win1_vs_factor_cbr Win1 CBR vertical scaling factor: YCbCr420: factor=((LCDC_WIN1_ACT_INFO[31:16]/ 2) /(LCDC_WIN1_DSP_INFO[31:16]))*2^12 YCbCr422,YCbCr444: factor=((LCDC_WIN1_ACT_INFO[31:16]) /(LCDC_WIN1_DSP_INFO[31:16]))*2^12
15:0	RW	0x1000	win1_hs_factor_cbr Win1 Cbr horizontal scaling factor: YCbCr422,YCbCr420: factor=((LCDC_WIN1_ACT_INFO[15:0]/2) /(LCDC_WIN1_DSP_INFO[15:0]))*2^12 YCbCr444: factor=((LCDC_WIN1_ACT_INFO[15:0]) /(LCDC_WIN1_DSP_INFO[15:0]))*2^12

VOP_WIN1_SCL_OFFSET

Address: Operational Base + offset (0x009c)
Win1 scaling start point offset

Bit	Attr	Reset Value	Description
31:24	RW	0x00	win1_vs_offset_cbr Cbr Vertical scaling start point offset (0x00~0xff)/0x100 = 0~0.99
23:16	RW	0x00	win1_vs_offset_yrgb Y Vertical scaling start point offset (0x00~0xff)/0x100 = 0~0.99
15:8	RW	0x00	win1_hs_offset_cbr Cbr Horizontal scaling start point offset (0x00~0xff)/0x100 = 0~0.99
7:0	RW	0x00	win1_hs_offset_yrgb Y Horizontal scaling start point offset (0x00~0xff)/0x100 = 0~0.99

VOP_WIN1_SRC_ALPHA_CTRL
Address: Operational Base + offset (0x00a0)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	win1_fading_value
23:16	RW	0x00	win1_src_global_alpha layer0 src global alpha （eused by fading value）
15:9	RO	0x0	reserved
8:6	RW	0x0	win1_src_factor_m0
5	RW	0x0	win1_src_alpha_cal_m0
4:3	RW	0x0	win1_src_blend_m0
2	RW	0x0	win1_src_alpha_m0
1	RW	0x0	win1_src_color_m0
0	RW	0x0	win1_src_alpha_en

VOP_WIN1_DST_ALPHA_CTRL
Address: Operational Base + offset (0x00a4)

Bit	Attr	Reset Value	Description
31:9	RO	0x0	reserved
8:6	RW	0x0	win1_dst_factor_m0

Bit	Attr	Reset Value	Description
5:0	RW	0x00	win1_dsp_m0_reserved

VOP_WIN1_FADING_CTRL
Address: Operational Base + offset (0x00a8)

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	win1_fading_en
23:16	RW	0x00	win1_fading_offset_b
15:8	RW	0x00	win1_fading_offset_g
7:0	RW	0x00	win1_fading_offset_r

VOP_WIN1_RESERVED0
Address: Operational Base + offset (0x00ac)

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	reserved

VOP_WIN2_CTRL0
Address: Operational Base + offset (0x00b0)
win2 ctrl register0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18	RW	0x0	win2_lut_en Win2 LUT ram enable 1'b0 : disable 1'b1 : enable *This bit should be "0" when CPU updates the LUT, and should be "1" when Win1 LUT mode enable.
17:15	RO	0x0	reserved
14	RW	0x0	win2_endian_swap Win2 8pp palette data Big-endian/ Little-endian select 1'b0 : Big-endian 1'b1 : Little-endian
13	RW	0x0	win2_alpha_swap Win2 RGB alpha swap 1'b0 : ARGB 1'b1 : RGBA

Bit	Attr	Reset Value	Description
12	RW	0x0	win2_rb_swap Win2 RGB Red and Blue swap 1'b0 : RGB 1'b1 : BGR
11	RO	0x0	reserved
10	RW	0x0	win2_csc_mode Win2 RGB2YUV Color space conversion: 1'b0 : no CSC 1'b1 : RGB2YUV
9	RW	0x0	win2_no_outstanding Win2 AXI master read outstanding 1'b0 : enable 1'b1 : disable
8	RW	0x0	win2_interlace_read Win2 interlace read mode 1'b0 : disable 1'b1 : enable
7	RW	0x0	win2_mst3_en win2 master3 enable
6	RW	0x0	win2_mst2_en win2 master2 enable
5	RW	0x0	win2_mst1_en win2 master1 enable
4	RW	0x0	win2_mst0_en win2 master0 enable
3:1	RW	0x0	win2_data_fmt 3'b000 : ARGB888 3'b001 : RGB888 3'b010 : RGB565 3'b100: 8bpp 3'b101: 4bpp 3'b110: 2bpp 3'b111: 1bpp
0	RW	0x0	win2_en

VOP_WIN2_CTRL1
Address: Operational Base + offset (0x00b4)
win2 ctrl register0

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:4	RW	0x0	win2_axi_gather_num
3:1	RO	0x0	reserved
0	RW	0x0	win2_axi_gather_en

VOP_WIN2_VIR0_1
Address: Operational Base + offset (0x00b8)
Win2 virtual stride0 and virtaul stride1

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x0140	win2_vir_stride1 Win2 Virtual stride1 Number of words of Win2 Virtual1 width ARGB888 : win2_vir_width1 RGB888 : (win2_vir_width1 * 3/4) + (win2_vir_width1 % 3) RGB565 : ceil(win2_vir_width1 / 2) 8BPP : ceil(win2_vir_width1 / 4) 4BPP : ceil(win2_vir_width1 / 8) 2BPP : ceil(win2_vir_width1 / 16) 1BPP : ceil(win2_vir_width1 / 32)
15:13	RO	0x0	reserved
12:0	RW	0x0140	win2_vir_stride0 Win2 Virtual stride0 Number of words of Win2 Virtual0 width ARGB888 : win2_vir_width0 RGB888 : (win2_vir_width0 * 3/4) + (win2_vir_width0 % 3) RGB565 : ceil(win2_vir_width0 / 2) 8BPP : ceil(win2_vir_width0 / 4) 4BPP : ceil(win2_vir_width0 / 8) 2BPP : ceil(win2_vir_width0 / 16) 1BPP : ceil(win2_vir_width0 / 32)

VOP_WIN2_VIR2_3
Address: Operational Base + offset (0x00bc)
Win2 virtual stride2 and virtaul stride3

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved

Bit	Attr	Reset Value	Description
28:16	RW	0x0140	win2_vir_stride3 Win2 Virtual stride3 Number of words of Win2 Virtual3 width ARGB888 : win2_vir_width3 RGB888 : (win2_vir_width3 * 3/4) + (win2_vir_width3 % 3) RGB565 : ceil(win2_vir_width3 / 2) 8BPP : ceil(win2_vir_width3 / 4) 4BPP : ceil(win2_vir_width3 / 8) 2BPP : ceil(win2_vir_width3 / 16) 1BPP : ceil(win1_vir_width3 / 32)
15:13	RO	0x0	reserved
12:0	RW	0x0140	win2_vir_stride2 Win2 Virtual stride2 Number of words of Win2 Virtual2 width ARGB888 : win2_vir_width2 RGB888 : (win2_vir_width2 * 3/4) + (win2_vir_width2 % 3) RGB565 : ceil(win2_vir_width2 / 2) 8BPP : ceil(win2_vir_width2 / 4) 4BPP : ceil(win2_vir_width2 / 8) 2BPP : ceil(win2_vir_width2 / 16) 1BPP : ceil(win1_vir_width2 / 32)

VOP_WIN2_MST0
Address: Operational Base + offset (0x00c0)
Win2 memory start address0

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win2_mst0 Win2 frame buffer memory start address0 *must be aliened to 8byte address

VOP_WIN2_DSP_INFO0
Address: Operational Base + offset (0x00c4)
Win2 display width0/height0 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win2_dsp_height0 Win2 display window height0 win2_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win2_dsp_width0 Win2 display window width0 win2_dsp_width = size -1

VOP_WIN2_DSP_ST0
Address: Operational Base + offset (0x00c8)
Win2 display start point0 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win2_dsp_yst0 Win2 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win2_dsp_xst0 Win2 horizontal start point(x) of the Panel scanning

VOP_WIN2_COLOR_KEY
Address: Operational Base + offset (0x00cc)
Win2 color key register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved

24	RW	0x0	win2_key_en
			Win2 transparency color key enable
			1'b0 : disable;
			1'b1 : enable;

23:0	RW	0x000000	win2_key_color Win2 key color

VOP_WIN2_MST1
Address: Operational Base + offset (0x00d0)
Win2 memory start address1

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win2_mst1 Win2 frame buffer memory start address1 *must be aligned to 8byte address

VOP_WIN2_DSP_INFO1
Address: Operational Base + offset (0x00d4)
Win2 display width1/height1 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win2_dsp_height1 Win2 display window height1 win2_dsp_height0 = size -1
15:12	RO	0x0	reserved

Bit	Attr	Reset Value	Description
11:0	RW	0x13f	win2_dsp_width1 Win2 display window width1 win2_dsp_width = size -1

VOP_WIN2_DSP_ST1
Address: Operational Base + offset (0x00d8)
Win2 display start point1 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win2_dsp_yst1 Win2 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win2_dsp_xst1 Win2 horizontal start point(x) of the Panel scanning

VOP_WIN2_SRC_ALPHA_CTRL
Address: Operational Base + offset (0x00dc)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	win2_fading_value
23:16	RW	0x00	win2_src_global_alpha layer0 src global alpha （eused by fading value）
15:9	RO	0x0	reserved
8:6	RW	0x0	win2_src_factor_m0
5	RW	0x0	win2_src_alpha_cal_m0
4:3	RW	0x0	win2_src_blend_m0
2	RW	0x0	win2_src_alpha_m0
1	RW	0x0	win2_src_color_m0
0	RW	0x0	win2_src_alpha_en

VOP_WIN2_MST2
Address: Operational Base + offset (0x00e0)
Win2 memory start address2

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win2_mst2 Win2 frame buffer memory start address2 *must be alianed to 8byte address

VOP_WIN2_DSP_INFO2
Address: Operational Base + offset (0x00e4)
Win2 display width2/height2 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win2_dsp_height2 Win2 display window height2 win2_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win2_dsp_width2 Win2 display window width2 win2_dsp_width = size -1

VOP_WIN2_DSP_ST2
Address: Operational Base + offset (0x00e8)
Win2 display start point2 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win2_dsp_yst2 Win2 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win2_dsp_xst2 Win2 horizontal start point(x) of the Panel scanning

VOP_WIN2_DST_ALPHA_CTRL
Address: Operational Base + offset (0x00ec)

Bit	Attr	Reset Value	Description
31:9	RO	0x0	reserved
8:6	RW	0x0	win2_dst_factor_m0
5:0	RW	0x00	win2_dst_m0_reserved

VOP_WIN2_MST3
Address: Operational Base + offset (0x00f0)
Win2 memory start address3

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win2_mst3 Win2 frame buffer memory start address3 *must be alianed to 8byte address

VOP_WIN2_DSP_INFO3
Address: Operational Base + offset (0x00f4)
Win2 display width3/height3 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win2_dsp_height3 Win2 display window height3 win2_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win2_dsp_width3 Win2 display window width3 win2_dsp_width = size -1

VOP_WIN2_DSP_ST3
Address: Operational Base + offset (0x00f8)
Win2 display start point3 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win2_dsp_yst3 Win2 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win2_dsp_xst3 Win2 horizontal start point(x) of the Panel scanning

VOP_WIN2_FADING_CTRL
Address: Operational Base + offset (0x00fc)

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	win2_fading_en
23:16	RW	0x00	win2_fading_offset_b
15:8	RW	0x00	win2_fading_offset_g
7:0	RW	0x00	win2_fading_offset_r

VOP_WIN3_CTRL0
Address: Operational Base + offset (0x0100)
win0 ctrl register0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18	RW	0x0	win3_lut_en Win3 LUT ram enable 1'b0 : disable 1'b1 : enable *This bit should be "0" when CPU updates the LUT, and should be "1" when Win1 LUT mode enable.
17:15	RO	0x0	reserved
14	RW	0x0	win3_endian_swap Win3 8pp palette data Big-endian/ Little-endian select 1'b0 : Big-endian 1'b1 : Little-endian
13	RW	0x0	win3_alpha_swap Win3 RGB alpha swap 1'b0 : ARGB 1'b1 : RGBA
12	RW	0x0	win3_rb_swap Win3 RGB Red and Blue swap 1'b0 : RGB 1'b1 : BGR
11	RO	0x0	reserved
10	RW	0x0	win3_csc_mode Win3 RGB2YUV Color space conversion: 1'b0 : no CSC 1'b1 : RGB2YUV
9	RW	0x0	win3_no_outstanding Win3 AXI master read outstanding 1'b0 : enable 1'b1 : disable
8	RW	0x0	win3_interlace_read Win3 interlace read mode 1'b0 : disable 1'b1 : enable
7	RW	0x0	win3_mst3_en win3 master3 enable
6	RW	0x0	win3_mst2_en win3 master2 enable

Bit	Attr	Reset Value	Description
5	RW	0x0	win3_mst1_en win3 master1 enable
4	RW	0x0	win3_mst0_en win3 master0 enable
3:1	RW	0x0	win3_data_fmt 3'b000 : ARGB888 3'b001 : RGB888 3'b010 : RGB565 3'b100: 8bpp 3'b101: 4bpp 3'b110: 2bpp 3'b111: 1bpp
0	RW	0x0	win3_en

VOP_WIN3_CTRL1
Address: Operational Base + offset (0x0104)
win0 ctrl register1

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:4	RW	0x0	win3_axi_gather_num
3:1	RO	0x0	reserved
0	RW	0x0	win3_axi_gather_en

VOP_WIN3_VIR0_1
Address: Operational Base + offset (0x0108)
Win3 virtual stride0 and virtaul stride1

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x0140	win3_vir_stride1 Win3 Virtual stride1 Number of words of Win3 Virtual1 width ARGB888 : win3_vir_width1 RGB888 : (win3_vir_width1 * 3/4) + (win3_vir_width1 % 3) RGB565 : ceil(win3_vir_width1 / 2) 8BPP : ceil(win3_vir_width1 / 4) 4BPP : ceil(win3_vir_width1 / 8) 2BPP : ceil(win3_vir_width1 / 16) 1BPP : ceil(win3_vir_width1 / 32)

Bit	Attr	Reset Value	Description
15:13	RO	0x0	reserved
12:0	RW	0x0140	win3_vir_stride0 Win3 Virtual stride0 Number of words of Win3 Virtual1 width ARGB888 : win3_vir_width1 RGB888 : (win3_vir_width1 * 3/4) + (win3_vir_width1 % 3) RGB565 : ceil(win3_vir_width1 / 2) 8BPP : ceil(win3_vir_width1 / 4) 4BPP : ceil(win3_vir_width1 / 8) 2BPP : ceil(win3_vir_width1 / 16) 1BPP : ceil(win3_vir_width1 / 32)

VOP_WIN3_VIR2_3
Address: Operational Base + offset (0x010c)
Win3 virtual stride2 and virtaul stride3

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x0140	win3_vir_stride3 Win3 Virtual stride3 Number of words of Win3 Virtual1 width ARGB888 : win3_vir_width1 RGB888 : (win3_vir_width1 * 3/4) + (win3_vir_width1 % 3) RGB565 : ceil(win3_vir_width1 / 2) 8BPP : ceil(win3_vir_width1 / 4) 4BPP : ceil(win3_vir_width1 / 8) 2BPP : ceil(win3_vir_width1 / 16) 1BPP : ceil(win3_vir_width1 / 32)
15:13	RO	0x0	reserved
12:0	RW	0x0140	win3_vir_stride2 Win3 Virtual stride2 Number of words of Win3 Virtual1 width ARGB888 : win3_vir_width1 RGB888 : (win3_vir_width1 * 3/4) + (win3_vir_width1 % 3) RGB565 : ceil(win3_vir_width1 / 2) 8BPP : ceil(win3_vir_width1 / 4) 4BPP : ceil(win3_vir_width1 / 8) 2BPP : ceil(win3_vir_width1 / 16) 1BPP : ceil(win3_vir_width1 / 32)

VOP_WIN3_MST0
Address: Operational Base + offset (0x0110)
Win3 memory start address0

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win3_mst0 Win3 frame buffer memory start address0 *must be ali231ned to 8byte address

VOP_WIN3_DSP_INFO0
Address: Operational Base + offset (0x0114)
Win3 display width0/height0 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win3_dsp_height0 Win3 display window height0 win3_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win3_dsp_width0 Win3 display window width0 win3_dsp_width = size -1

VOP_WIN3_DSP_ST0
Address: Operational Base + offset (0x0118)
Win3 display start point0 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win3_dsp_yst0 Win3 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win3_dsp_xst0 Win3 horizontal start point(x) of the Panel scanning

VOP_WIN3_COLOR_KEY
Address: Operational Base + offset (0x011c)
Win3 color key register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved

24	RW	0x0	win3_key_en
			Win3 transparency color key enable
			1'b0 : disable;
			1'b1 : enable;

23:0	RW	0x000000	win3_key_color Win3 key color

VOP_WIN3_MST1

Address: Operational Base + offset (0x0120)
Win3 memory start address1

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win3_mst1 Win3 frame buffer memory start address1 *must be aliened to 8byte address

VOP_WIN3_DSP_INFO1
Address: Operational Base + offset (0x0124)
Win3 display width1/height1 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win3_dsp_height1 Win3 display window height1 win3_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win3_dsp_width1 Win3 display window width1 win3_dsp_width = size -1

VOP_WIN3_DSP_ST1
Address: Operational Base + offset (0x0128)
Win3 display start point1 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win3_dsp_yst1 Win3 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win3_dsp_xst1 Win3 horizontal start point(x) of the Panel scanning

VOP_WIN3_SRC_ALPHA_CTRL
Address: Operational Base + offset (0x012c)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	win3_fading_value
23:16	RW	0x00	win3_src_global_alpha layer0 src global alpha （eused by fading value）
15:9	RO	0x0	reserved
8:6	RW	0x0	win3_src_factor_m0

Bit	Attr	Reset Value	Description
5	RW	0x0	win3_src_alpha_cal_m0
4:3	RW	0x0	win3_src_blend_m0
2	RW	0x0	win3_src_alpha_m0
1	RW	0x0	win3_src_color_m0
0	RW	0x0	win3_src_alpha_en

VOP_WIN3_MST2
Address: Operational Base + offset (0x0130)
Win3 memory start address2

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win3_mst2 Win3 frame buffer memory start address2 *must be aliened to 8byte address

VOP_WIN3_DSP_INFO2
Address: Operational Base + offset (0x0134)
Win3 display width2/height2 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win3_dsp_height2 Win3 display window height2 win3_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win3_dsp_width2 Win3 display window width2 win3_dsp_width = size -1

VOP_WIN3_DSP_ST2
Address: Operational Base + offset (0x0138)
Win3 display start point2 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win3_dsp_yst2 Win3 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win3_dsp_xst2 Win3 horizontal start point(x) of the Panel scanning

VOP_WIN3_DST_ALPHA_CTRL
Address: Operational Base + offset (0x013c)

Bit	Attr	Reset Value	Description
31:9	RO	0x0	reserved
8:6	RW	0x0	win3_dst_factor_m0
5:0	RW	0x00	win3_dst_factor_reserved

VOP_WIN3_MST3
Address: Operational Base + offset (0x0140)
Win3 memory start address3

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	win3_mst3 Win3 frame buffer memory start address3 *must be alianed to 8byte address

VOP_WIN3_DSP_INFO3
Address: Operational Base + offset (0x0144)
Win3 display width3/height3 on panel

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x0ef	win3_dsp_height3 Win3 display window height3 win3_dsp_height0 = size -1
15:12	RO	0x0	reserved
11:0	RW	0x13f	win3_dsp_width3 Win3 display window width3 win3_dsp_width = size -1

VOP_WIN3_DSP_ST3
Address: Operational Base + offset (0x0148)
Win3 display start point3 on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	win3_dsp_yst3 Win3 vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	win3_dsp_xst3 Win3 horizontal start point(x) of the Panel scanning

VOP_WIN3_FADING_CTRL

Address: Operational Base + offset (0x014c)

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	win3_fading_en
23:16	RW	0x00	win3_fading_offset_b
15:8	RW	0x00	win3_fading_offset_g
7:0	RW	0x00	win3_fading_offset_r

VOP_HWC_CTRL0
Address: Operational Base + offset (0x0150)
Hwc ctrl register0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18	RW	0x0	hwc_lut_en
17:15	RO	0x0	reserved
14	RW	0x0	hwc_endian_swap hwc 8pp palette data Big-endian/ Little-endian select 1'b0 : Big-endian 1'b1 : Little-endian
13	RW	0x0	hwc_alpha_swap hwc RGB alpha swap 1'b0 : ARGB 1'b1 : RGBA
12	RW	0x0	hwc_rb_swap hwc RGB Red and Blue swap 1'b0 : RGB 1'b1 : BGR
11	RO	0x0	reserved
10	RW	0x0	hwc_csc_mode hwc RGB2YUV Color space conversion: 1'b0 : no CSC 1'b1 : RGB2YUV
9	RW	0x0	hwc_no_outstanding hwc AXI master read outstanding 1'b0 : enable 1'b1 : disable

Bit	Attr	Reset Value	Description
8	RW	0x0	hwc_interlace_read hwc interlace read mode 1'b0 : disable 1'b1 : enable
7	RO	0x0	reserved
6:5	RW	0x0	hwc_size 2'b00 : 32x32 2'b01 : 64x64 2'b10 : 96x96 2'b11 : 128x128
4	RW	0x0	hwc_mode hwc color mode 1'b0 : normal color mode 1'b1 : reversed color mode
3:1	RW	0x0	hwc_data_fmt
0	RW	0x0	hwc_en

VOP_HWC_CTRL1
Address: Operational Base + offset (0x0154)
Hwc ctrl register1

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6:4	RW	0x0	win3_axi_gather_num
3:1	RO	0x0	reserved
0	RW	0x0	win3_axi_gather_en

VOP_HWC_MST
Address: Operational Base + offset (0x0158)
Hwc memory start address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	hwc_mst HWC data memory start address

VOP_HWC_DSP_ST
Address: Operational Base + offset (0x015c)
Hwc display start point on panel

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved

Bit	Attr	Reset Value	Description
28:16	RW	0x000a	hwc_dsp_yst HWC vertical start point(y) of the Panel scanning
15:13	RO	0x0	reserved
12:0	RW	0x000a	hwc_dsp_xst HWC horizontal start point(x) of the Panel scanning

VOP_HWC_SRC_ALPHA_CTRL
Address: Operational Base + offset (0x0160)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	hwc_fading_value
23:16	RW	0x00	hwc_src_global_alpha layer0 src global alpha （eused by fading value）
15:9	RO	0x0	reserved
8:6	RW	0x0	hwc_src_factor_m0
5	RW	0x0	hwc_src_alpha_cal_m0
4:3	RW	0x0	hwc_src_blend_m0
2	RW	0x0	hwc_src_alpha_m0
1	RW	0x0	hwc_src_color_m0
0	RW	0x0	hwc_src_alpha_en

VOP_HWC_DST_ALPHA_CTRL
Address: Operational Base + offset (0x0164)

Bit	Attr	Reset Value	Description
31:9	RO	0x0	reserved
8:6	RW	0x0	hwc_dst_factor_m0
5:0	RW	0x00	hwc_dst_m0_reserved

VOP_HWC_FADING_CTRL
Address: Operational Base + offset (0x0168)

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	hwc_fading_en
23:16	RW	0x00	hwc_fading_offset_b
15:8	RW	0x00	hwc_fading_offset_g
7:0	RW	0x00	hwc_fading_offset_r

VOP_HWC_RESERVED1
Address: Operational Base + offset (0x016c)

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	reserved

VOP_POST_DSP_HACT_INFO
Address: Operational Base + offset (0x0170)
post scaler down horizontal start and end

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	dsp_hact_st_post Panel display scanning horizontal active start point
15:13	RO	0x0	reserved
12:0	RW	0x014a	dsp_hact_end_post Panel display scanning horizontal active end point

VOP_POST_DSP_VACT_INFO
Address: Operational Base + offset (0x0174)
Panel active horizontal scanning start point and end point

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	dsp_vact_st_post Panel display scanning horizontal active start point
15:13	RO	0x0	reserved
12:0	RW	0x00fa	dsp_vact_end_post Panel display scanning horizontal active end point

VOP_POST_SCL_FACTOR_YRGB

Address: Operational Base + offset (0x0178)
post yrgb scaling factor

Bit	Attr	Reset Value	Description
31:16	RW	0x1000	post_vs_factor_yrgb post YRGB vertical scaling factor: factor=((src_height[31:16]) /(dst_height[31:16]))*2^12
15:0	RW	0x1000	post_hs_factor_yrgb Post YRGB horizontal scaling factor: factor=((src_width[15:0]) /(dst_width[15:0]))*2^12

VOP_POST_RESERVED
Address: Operational Base + offset (0x017c)

Bit	Attr	Reset Value	Description
31:0	RW	0x00001000	post_reserved

VOP_POST_SCL_CTRL
Address: Operational Base + offset (0x0180)
post scaling start point offset

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	post_ver_sd_en 1'b0 : post ver scl down disable 1'b1 : post ver scl down enable
0	RW	0x0	post_hor_sd_en 1'b0 : post hor scl down disable 1'b1 : post hor scl down enable

VOP_POST_DSP_VACT_INFO_F1
Address: Operational Base + offset (0x0184)
Panel active horizontal scanning start point and end point F1

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	dsp_vact_st_post Panel display scanning horizontal active start point
15:13	RO	0x0	reserved
12:0	RW	0x00fa	dsp_vact_end_post Panel display scanning horizontal active end point

VOP_DSP_HTOTAL_HS_END
Address: Operational Base + offset (0x0188)

Panel scanning horizontal width and hsync pulse end point

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x014a	dsp_htotal Panel display scanning horizontal period
15:13	RO	0x0	reserved
12:0	RW	0x000a	dsp_hs_end Panel display scanning hsync pulse width

VOP_DSP_HACT_ST_END
Address: Operational Base + offset (0x018c)
Panel active horizontal scanning start point and end point

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	dsp_hact_st Panel display scanning horizontal active start point
15:13	RO	0x0	reserved
12:0	RW	0x014a	dsp_hact_end Panel display scanning horizontal active end point

VOP_DSP_VTOTAL_VS_END
Address: Operational Base + offset (0x0190)
Panel scanning vertical height and vsync pulse end point

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x00fa	dsp_vtotal Panel display scanning vertical period.
15:13	RO	0x0	reserved
12:0	RW	0x000a	dsp_vs_end Panel display scanning vsync pulse width

VOP_DSP_VACT_ST_END
Address: Operational Base + offset (0x0194)
Panel active vertical scanning start point and end point

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x000a	dsp_vact_st Panel display scanning vertical active start point
15:13	RO	0x0	reserved

Bit	Attr	Reset Value	Description
12:0	RW	0x00fa	dsp_vact_end Panel display scanning vertical active end point

VOP_DSP_VS_ST_END_F1
Address: Operational Base + offset (0x0198)
Vertical scanning start point and vsync pulse end point of even filed in interlace mode

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x0000	dsp_vs_st_f1 Panel display scanning vertical vsync start point of 2nd field (interlace display mode)
15:13	RO	0x0	reserved
12:0	RW	0x0000	dsp_vs_end_f1 Panel display scanning vertical vsync end point of 2nd field(interlace display mode)

VOP_DSP_VACT_ST_END_F1
Address: Operational Base + offset (0x019c)
Vertical scanning active start point and end point of even filed in interlace mode

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:16	RW	0x0000	dsp_vact_st_f1 Panel display scanning vertical active start point of 2nd field (interlace display mode)
15:13	RO	0x0	reserved
12:0	RW	0x0000	dsp_vact_end_f1 Panel display scanning vertical active end point of 2nd field (interlace display mode)

VOP_PWM_CTRL
Address: Operational Base + offset (0x01a0)
PWM Control Register

Bit	Attr	Reset Value	Description
31:24	RW	0x00	rpt Repeat Counter This field defines the repeated effective periods of output waveform in one-shot mode. The value N means N+1 repeated effective periods.
23:16	RW	0x00	scale Scale Factor This fields defines the scale factor applied to prescaled clock. The value N means the clock is divided by 2*N. If N is 0, it means that the clock is divided by 512(2*256).

Bit	Attr	Reset Value	Description
15	RO	0x0	reserved
14:12	RW	0x2	prescale Prescale Factor This field defines the prescale factor applied to input clock. The value N means that the input clock is divided by 2^N.
11:10	RO	0x0	reserved
9	RW	0x0	clk_sel Clock Source Select 0: non-scaled clock is selected as PWM clock source. It means that the prescale clock is directly used as the PWM clock source 1: scaled clock is selected as PWM clock source
8	RW	0x0	lp_en Low Power Mode Enable 0: disabled 1: enabled When PWM channel is inactive state and Low Power Mode is enabled, the path to PWM Clock prescale module is blocked to reduce power consumption.
7:6	RO	0x0	reserved
5	RW	0x0	output_mode PWM Output mode 0: left aligned mode 1: center aligned mode
4	RW	0x0	inactive_pol Inactive State Output Polarity This defines the output waveform polarity when PWM channel is in inactive state. The inactive state means that PWM finishes the complete waveform in one-shot mode or PWM channel is disabled. 0: negative 1: positive
3	RW	0x1	duty_pol Duty Cycle Output Polarity This defines the polarity for duty cycle. PWM starts the output waveform with duty cycle. 0: negative 1: positive

Bit	Attr	Reset Value	Description
2:1	RW	0x1	pwm_mode PWM Operation Mode 00: One shot mode. PWM produces the waveform within the repeated times defined by PWMx_CTRL_rpt . 01: Continuous mode. PWM produces the waveform continuously 10: Capture mode. PWM measures the cycles of high/low polarity of input waveform. 11: reserved
0	RW	0x0	pwm_en PWM channel enable 0: disabled 1: enabled. If the PWM is worked the one-shot mode, this bit will be cleared at the end of operation

VOP_PWM_PERIOD_HPR
Address: Operational Base + offset (0x01a4)
PWM Period Register/High Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	pwm_period Output Waveform Period/Input Waveform High Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the period of the output waveform. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the bit [31:1] is taken into account and bit [0] always considered as 0. If PWM is operated at the capture mode, this value indicates the effective high polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to (2^32-1).

VOP_PWM_DUTY_LPR
Address: Operational Base + offset (0x01a8)
PWM Duty Register/Low Polarity Capture Register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	pwm_duty Output Waveform Duty Cycle/Input Waveform Low Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the duty cycle of the output waveform. The PWM starts the output waveform with duty cycle. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the [31:1] is taken into account. If PWM is operated at the capture mode, this value indicates the effective low polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to (2^32-1).

VOP_PWM_CNT
Address: Operational Base + offset (0x01ac)
PWM Counter Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	pwm_cnt Timer Counter The 32-bit indicates current value of PWM Channel 0 counter. The counter runs at the rate of PWM clock. The value ranges from 0 to (2^32-1).

VOP_BCSH_COLOR_BAR
Address: Operational Base + offset (0x01b0)
color bar config register

Bit	Attr	Reset Value	Description
31:22	RW	0x000	color_bar_v v color value
21:12	RW	0x000	color_bar_u u color value
11:2	RW	0x000	color_bar_y y color value
1	RO	0x0	reserved
0	RW	0x0	bcsh_en 1'b0 : bcsh bypass 1'b1 : bcsh enable

VOP_BCSH_BCS
Address: Operational Base + offset (0x01b4)
brightness contrast saturation*contrast config register

Bit	Attr	Reset Value	Description
31:30	RW	0x3	out_mode video out mode config register 2'b00 : black 2'b01 : blue 2'b10 : color bar 2'b11 : normal video
29:20	RW	0x100	sat_con Saturation*Contrast*256 : 0,1.992*1.992
19:17	RO	0x0	reserved
16:8	RW	0x100	contrast Contrast*256 : 0,1.992
7:0	RW	0x00	brightness Brightness : -128,127

VOP_BCSH_H
Address: Operational Base + offset (0x01b8)
sin hue and cos hue config register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24:16	RW	0x100	cos_hue cos hue value
15:9	RO	0x0	reserved
8:0	RW	0x000	sin_hue sin hue value

VOP_BCSH_RESERVED
Address: Operational Base + offset (0x01bc)

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	reserved

VOP_CABC_CTRL0
Address: Operational Base + offset (0x01c0)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	cabc_stage_up

Bit	Attr	Reset Value	Description
23:1	RW	0x000000	cabc_calc_pixel_num Field0000 Abstract Field0000 Description
0	RW	0x0	cabc_en 1'b0 : cabc disable 1'b1 : cabc enable

VOP_CABC_CTRL1
Address: Operational Base + offset (0x01c4)

Bit	Attr	Reset Value	Description
31:24	RW	0x00	cabc_stage_down Field0000 Abstract Field0000 Description
23:1	RW	0x000000	cabc_total_num
0	RO	0x0	reserved

VOP_CABC_GAUSS_LINE0_0
Address: Operational Base + offset (0x01c8)
Register0000 Abstract

Bit	Attr	Reset Value	Description
31:24	RW	0x15	t_line0_3
23:16	RW	0x11	t_line0_2
15:8	RW	0x09	t_line0_1
7:0	RW	0x03	t_line0_0

VOP_CABC_GAUSS_LINE0_1
Address: Operational Base + offset (0x01cc)
Register0001 Abstract

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RW	0x03	t_line0_6
15:8	RW	0x09	t_line0_5
7:0	RW	0x11	t_line0_4

VOP_CABC_GAUSS_LINE1_0

Address: Operational Base + offset (0x01d0)
Register0002 Abstract

Bit	Attr	Reset Value	Description
31:24	RW	0x1a	t_line1_3
23:16	RW	0x15	t_line1_2
15:8	RW	0x0b	t_line1_1
7:0	RW	0x04	t_line1_0

VOP_CABC_GAUSS_LINE1_1
Address: Operational Base + offset (0x01d4)
Register0003 Abstract

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RW	0x04	t_line1_6
15:8	RW	0x0b	t_line1_5
7:0	RW	0x15	t_line1_4

VOP_CABC_GAUSS_LINE2_0
Address: Operational Base + offset (0x01d8)
Register0004 Abstract

Bit	Attr	Reset Value	Description
31:24	RW	0x15	t_line2_3
23:16	RW	0x11	t_line2_2
15:8	RW	0x09	t_line2_1
7:0	RW	0x03	t_line2_0

VOP_CABC_GAUSS_LINE2_1
Address: Operational Base + offset (0x01dc)
Register0005 Abstract

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RW	0x03	t_line2_6

Bit	Attr	Reset Value	Description
15:8	RW	0x09	t_line2_5
7:0	RW	0x11	t_line2_4

VOP_FRC_LOWER01_0
Address: Operational Base + offset (0x01e0)

Bit	Attr	Reset Value	Description
31:16	RW	0x1284	lower01_frm1
15:0	RW	0x4821	lower01_frm0

VOP_FRC_LOWER01_1
Address: Operational Base + offset (0x01e4)

Bit	Attr	Reset Value	Description
31:16	RW	0x2148	lower01_frm3
15:0	RW	0x8412	lower01_frm2

VOP_FRC_LOWER10_0
Address: Operational Base + offset (0x01e8)

Bit	Attr	Reset Value	Description
31:16	RW	0xa55a	lower10_frm1
15:0	RW	0x9696	lower10_frm0

VOP_FRC_LOWER10_1
Address: Operational Base + offset (0x01ec)

Bit	Attr	Reset Value	Description
31:16	RW	0x5aa5	lower10_frm3
15:0	RW	0x6969	lower10_frm2

VOP_FRC_LOWER11_0
Address: Operational Base + offset (0x01f0)

Bit	Attr	Reset Value	Description
31:16	RW	0xdeb7	lower11_frm1
15:0	RW	0x7bed	lower11_frm0

VOP_FRC_LOWER11_1
Address: Operational Base + offset (0x01f4)

Bit	Attr	Reset Value	Description
31:16	RW	0xed7b	lower11_frm3
15:0	RW	0xb7de	lower11_frm2

VOP_FRC_RESERVED0
Address: Operational Base + offset (0x01f8)

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

VOP_FRC_RESERVED1
Address: Operational Base + offset (0x01fc)

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

VOP_MMU_DTE_ADDR
Address: Operational Base + offset (0x0300)
MMU current page Table address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	MMU_DTE_ADDR Field0000 Abstract Field0000 Description

VOP_MMU_STATUS
Address: Operational Base + offset (0x0304)
MMU status register

Bit	Attr	Reset Value	Description
31:11	RO	0x0	reserved

Bit	Attr	Reset Value	Description
10:6	RO	0x00	PAGE_FAULT_BUS_ID Field0000 Abstract Index of master reponsible for last page fault
5	RO	0x0	PAGE_FAULT_IS_WRITE Field0000 Abstract The direction of access for last page fault: 0 = Read 1 = Write
4	RO	0x0	REPLAY_BUFFER_EMPTY Field0000 Abstract The MMU replay buffer is empty
3	RO	0x0	MMU_IDLE Field0000 Abstract The MMU is idle when accesses are being translated and there are no unfinished translated accesses.
2	RO	0x0	STAIL_ACTIVE Field0001 Abstract MMU stall mode currently enabled. The mode is enabled by command
1	RO	0x0	PAGE_FAULT_ACTIVE Field0000 Abstract MMU page fault mode currently enabled . The mode is enabled by command.
0	RO	0x0	PAGING_ENABLED Field0000 Abstract Paging is enabled

VOP_MMU_COMMAND
Address: Operational Base + offset (0x0308)
MMU command register

Bit	Attr	Reset Value	Description
31:3	RO	0x0	reserved
2:0	WO	0x0	MMU_CMD Field0000 Abstract MMU_CMD. This can be: 0: MMU_ENABLE_PAGING 1: MMU_DISABLE_PAGING 2: MMU_ENABLE_STALL 3: MMU_DISABLE_STALL 4: MMU_ZAP_CACHE 5: MMU_PAGE_FAULT_DONE 6: MMU_FORCE_RESET

VOP_MMU_PAGE_FAULT_ADDR

Address: Operational Base + offset (0x030c)
MMU logical address of last page fault

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	PAGE_FAULT_ADDR Field0000 Abstract address of last page fault

VOP_MMU_ZAP_ONE_LINE
Address: Operational Base + offset (0x0310)
MMU Zap cache line register

Bit	Attr	Reset Value	Description
31:0	WO	0x00000000	MMU_ZAP_ONE_LINE Field0000 Abstract address to be invalidated from the page table cache

VOP_MMU_INT_RAWSTAT
Address: Operational Base + offset (0x0314)
MMU raw interrupt status register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	READ_BUS_ERROR Field0000 Abstract read bus error
0	RW	0x0	PAGE_FAULT Field0000 Abstract page fault

VOP_MMU_INT_CLEAR
Address: Operational Base + offset (0x0318)
MMU raw interrupt status register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	WO	0x0	READ_BUS_ERROR Field0000 Abstract read bus error
0	WO	0x0	PAGE_FAULT Field0000 Abstract page fault

VOP_MMU_INT_MASK
Address: Operational Base + offset (0x031c)
MMU raw interrupt status register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved

Bit	Attr	Reset Value	Description
1	RW	0x0	READ_BUS_ERROR Field0000 Abstract read bus error
0	RW	0x0	PAGE_FAULT Field0000 Abstract page fault

VOP_MMU_INT_STATUS
Address: Operational Base + offset (0x0320)
MMU raw interrupt status register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RO	0x0	READ_BUS_ERROR Field0000 Abstract read bus error
0	RO	0x0	PAGE_FAULT Field0000 Abstract page fault

VOP_MMU_AUTO_GATING
Address: Operational Base + offset (0x0324)
mmu auto gating

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	mmu_auto_gating mmu auto gating when it is 1'b1, the mmu will auto gating it self

VOP_WIN2_LUT_ADDR
Address: Operational Base + offset (0x0400)

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

VOP_WIN3_LUT_ADDR
Address: Operational Base + offset (0x0800)

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

VOP_HWC_LUT_ADDR
Address: Operational Base + offset (0x0c00)

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

VOP_GAMMA_LUT_ADDR
Address: Operational Base + offset (0x1000)

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

VOP_MCU_BYPASS_WPORT
Address: Operational Base + offset (0x2200)
Register0000 Abstract

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

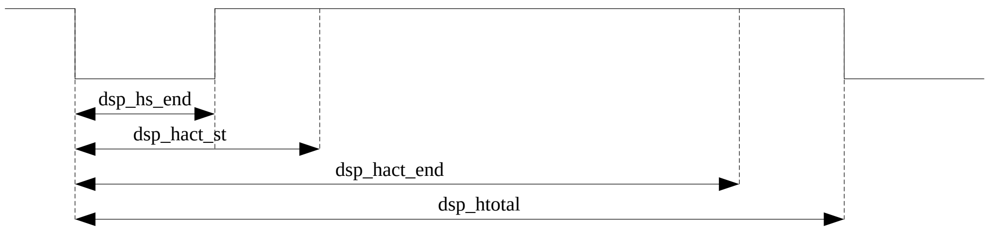
VOP_MCU_BYPASS_RPORT
Address: Operational Base + offset (0x2300)
Register0001 Abstract

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	Field0000 Abstract Field0000 Description

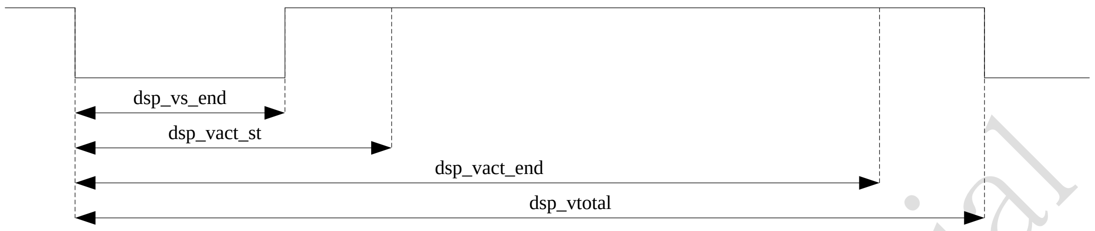
27.5 Timing Diagram

27.5.1 RGB LCD interface timing

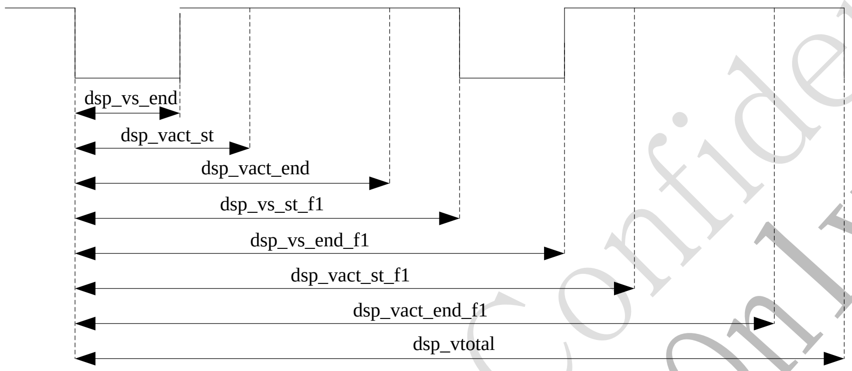
1.Timing parameter



Horizontal timing

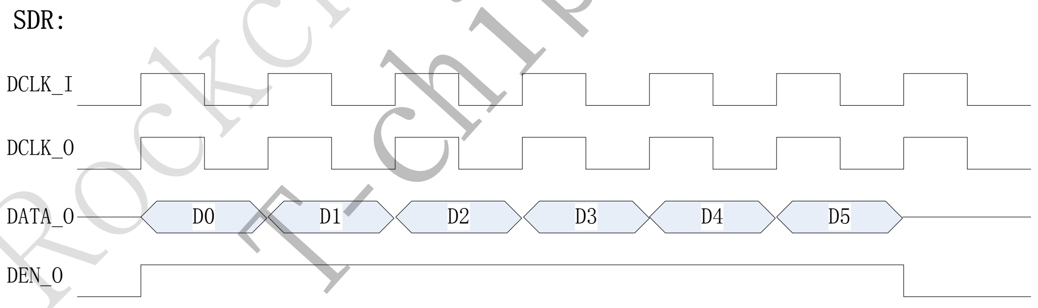


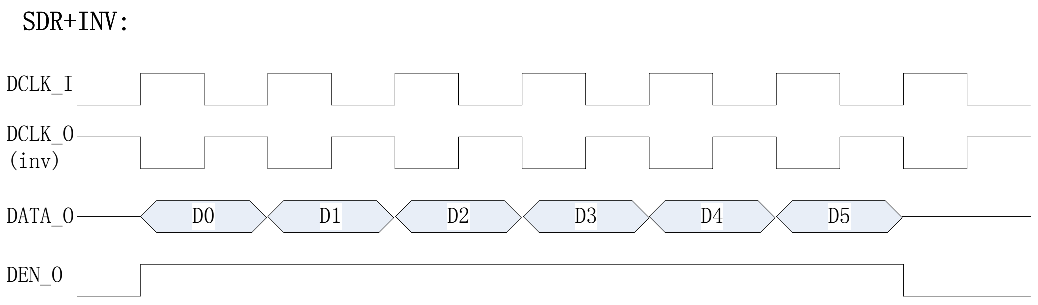
Vertical timing(Progressive mode)



Vertical timing(Interlace mode)

2.Data timing for RGB LCD SDR interface





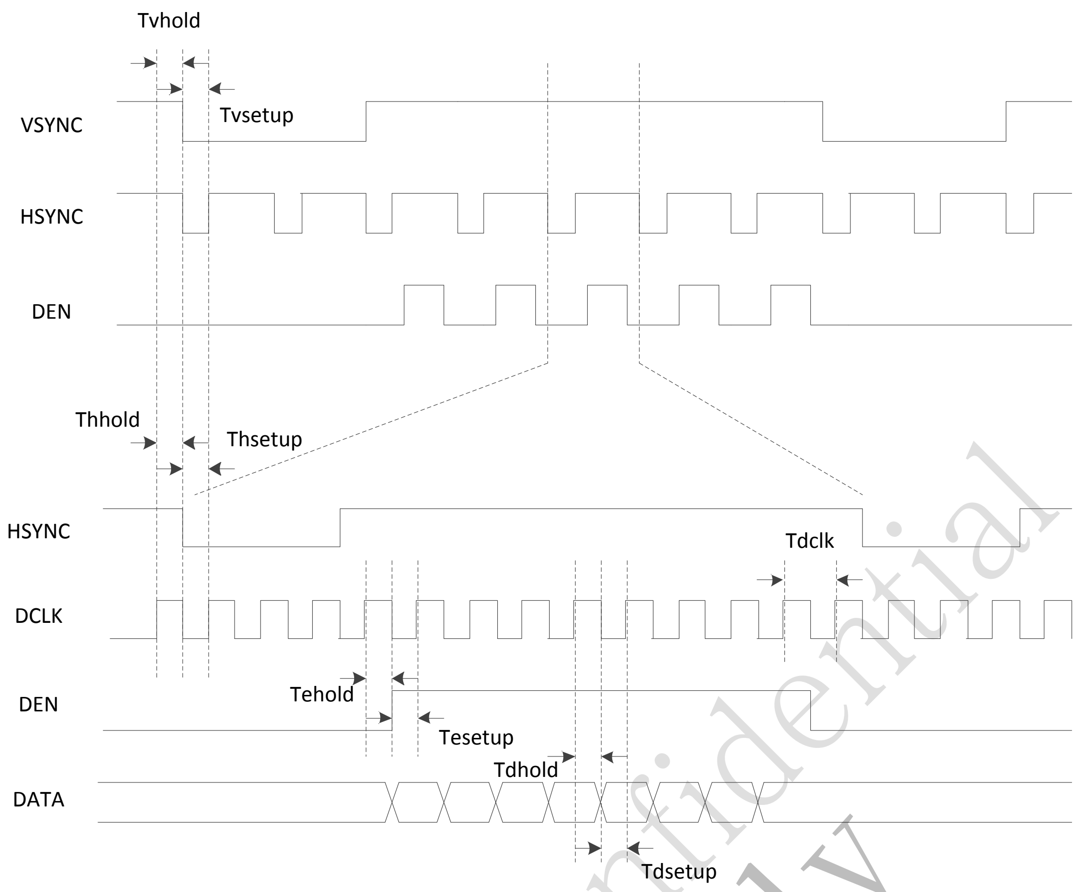


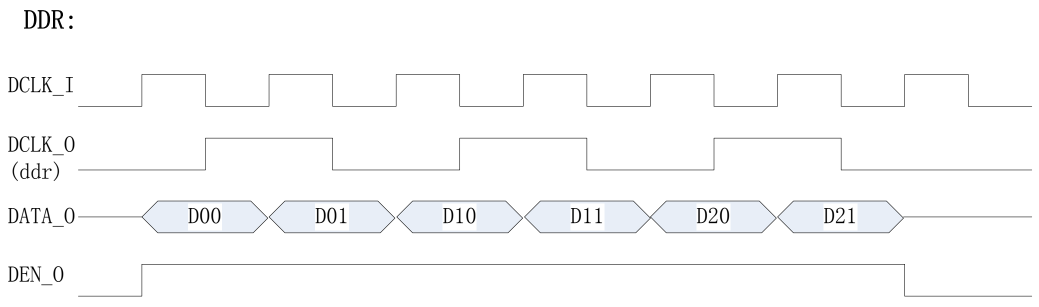
Fig. 27-17 LCDC RGB interface timing (SDR)

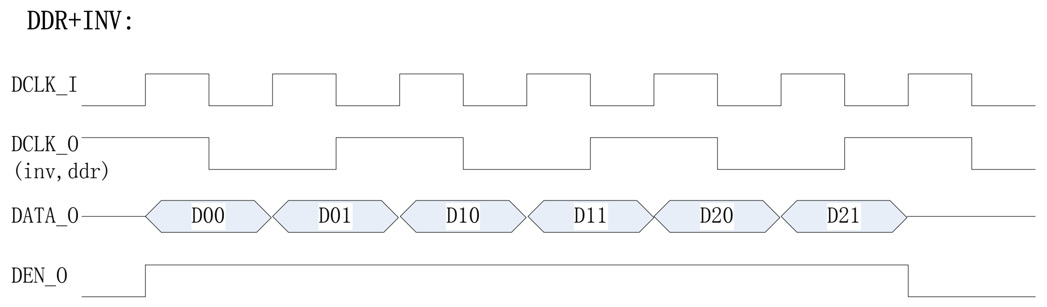
Table 27-2 LCDC0 RGB interface(SDR) signal timing constant

(VDD_core =0.9V to 1.1V, VDD_IO=3.0V to 3.6V , TA = -40℃ ot 125℃)

Item	Symbol	Min	Typ	Max	Unit
Display clock period	Tdclk				ns
VSYNC setup to DCLK falling edge	Tvsetup				ns
VSYNC hold from DCLK falling edge	Tvhold				ns
HSYNC setup to DCLK falling edge	Thsetup				ns
HSYNC hold from DCLK falling edge	Thhold				ns
DEN setup to DCLK falling edge	Tesetup				ns
DEN hold from DCLK falling edge	Tehold				ns
DATA setup to DCLK falling edge	Tdsetup				ns
DATA hold from DCLK falling edge	Tdhold				ns

3.Data timing for RGB LCD DDR interface





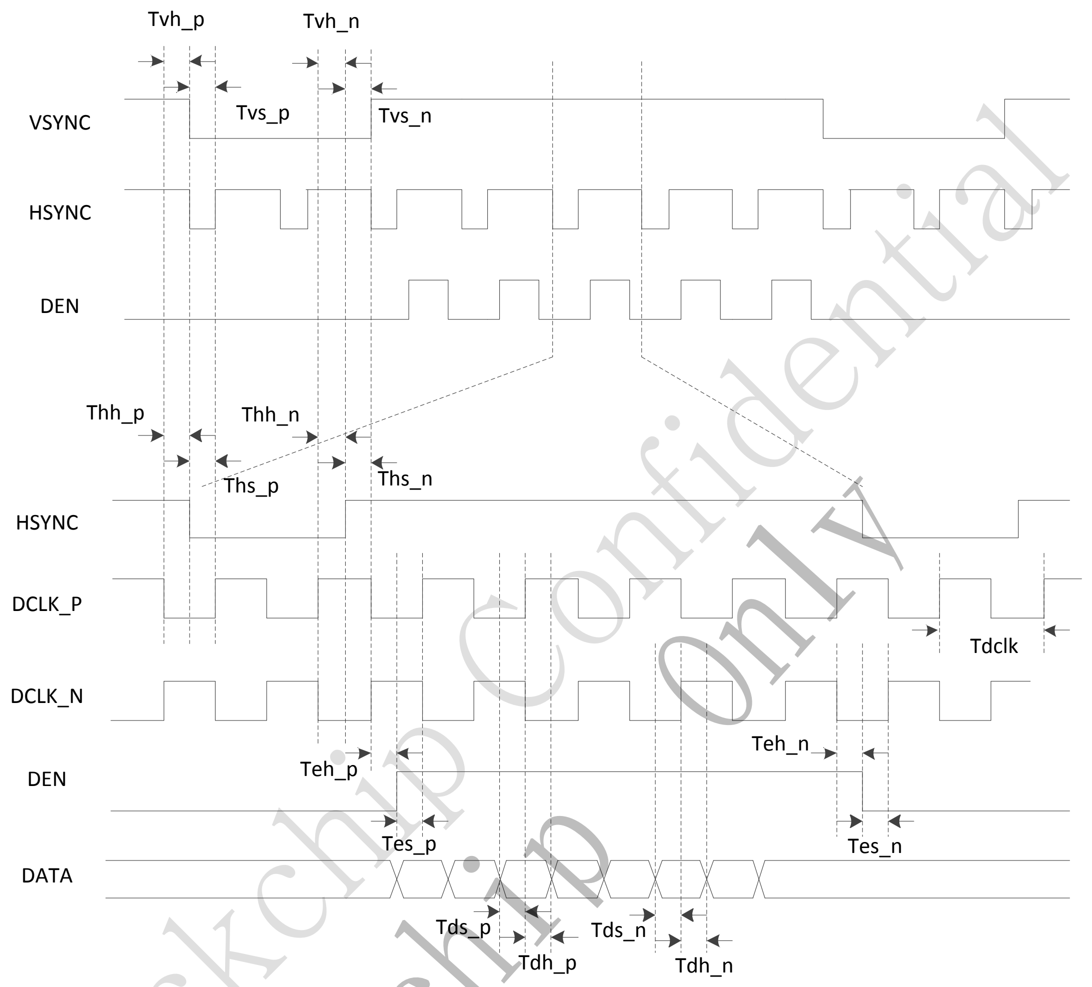


Fig. 27-18 LCDC RGB interface timing (DDR)

Table 27-3 LCDC0 RGB interface (DDR) signal timing constant
(VDD_core =0.9V to 1.1V, VDD_IO=3.0V to 3.6V , TA = -40℃ ot 125℃)

Item	Symbol	Min	Typ	Max	Unit
Display clock period	Tdclk				ns
VSYNC setup to DCLK_N falling edge	Tvs_p				ns
VSYNC hold from DCLK_N falling edge	Tvh_p				ns
HSYNC setup to DCLK_N falling edge	Ths_p				ns
HSYNC hold from DCLK_N falling edge	Thh_p				ns
DEN setup to DCLK_N falling edge	Tes_p				ns
DEN hold from DCLK_N falling edge	The_p				ns
DATA setup to DCLK_N falling edge	Tds_p				ns
DATA hold from DCLK_N falling edge	Tdh_p				ns
VSYNC setup to DCLK_P falling edge	Tvs_p				ns
VSYNC hold from DCLK_P falling edge	Tvh_p				ns
HSYNC setup to DCLK_P falling edge	Ths_p				ns

HSYNC hold from DCLK_P falling edge	Thh_p				ns
DEN setup to DCLK_P falling edge	Tes_p				ns
DEN hold from DCLK_P falling edge	The_p				ns
DATA setup to DCLK_P falling edge	Tds_p				ns
DATA hold from DCLK_P falling edge	Tdh_p				ns

Table 27-4 LCDC1 RGB interface signal timing constant
(VDD_core =0.9V to 1.1V, VDD_IO=3.0V to 3.6V , TA = -40℃ ot 125℃)

Item	Symbol	Min	Typ	Max	Unit
Display clock period	Tdclk				ns
VSYNC setup to DCLK falling edge	Tvsetup				ns
VSYNC hold from DCLK falling edge	Tvhold				ns
HSYNC setup to DCLK falling edge	Thsetup				ns
HSYNC hold from DCLK falling edge	Thhold				ns
DEN setup to DCLK falling edge	Tesetup				ns
DEN hold from DCLK falling edge	Tehold				ns
DATA setup to DCLK falling edge	Tdsetup				ns
DATA hold from DCLK falling edge	Tdhold				ns

27.5.2 MCU LCD interface timing

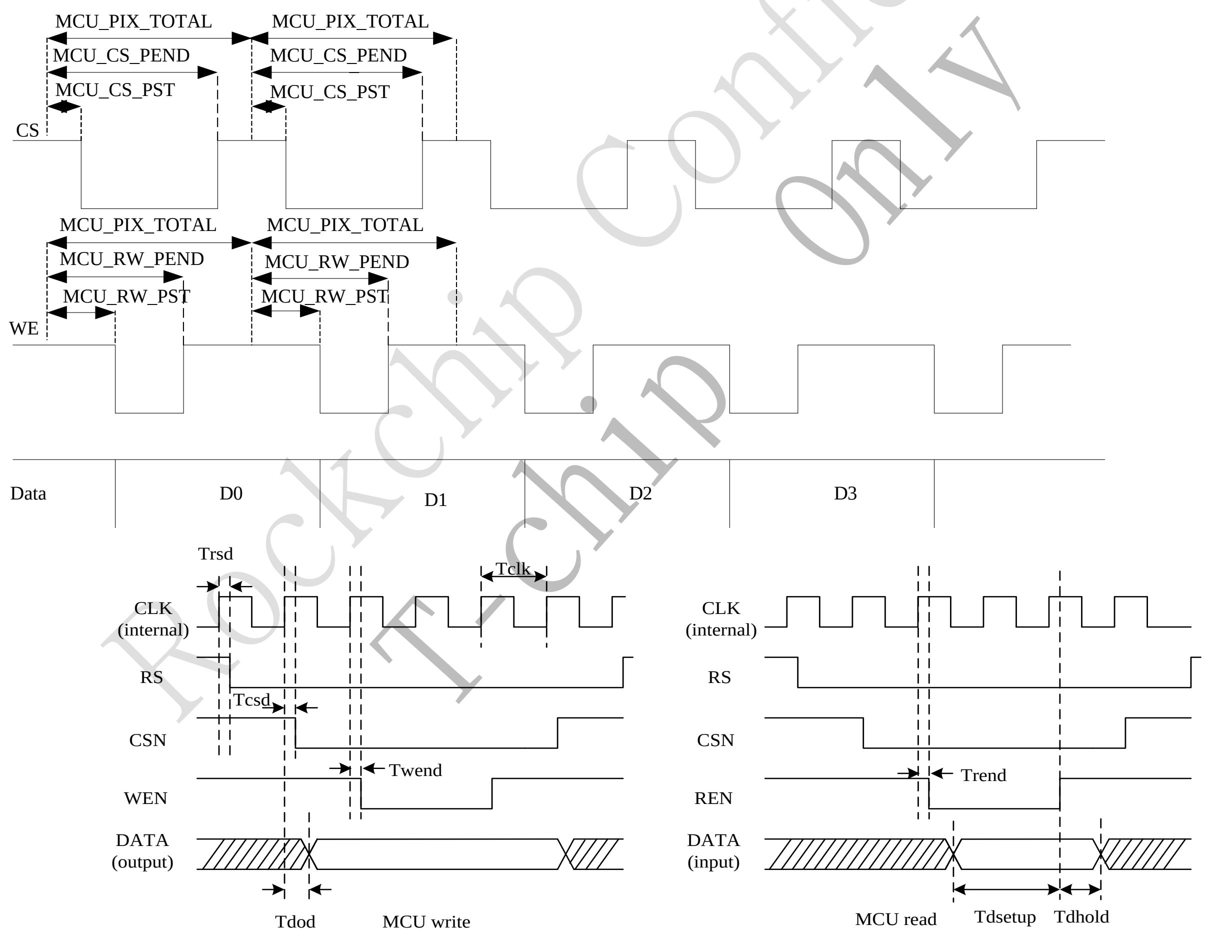


Fig. 27-19 LCDC MCU interface (i80) timing

Table 27-5 LCDC0 RGB interface signal timing constant
(VDD_core =0.9V to 1.1V, VDD_IO=3.0V to 3.6V , TA = -40℃ ot 125℃)

Item	Symbol	Min	Typ	Max	Unit
Internal clock period	Tclk				ns

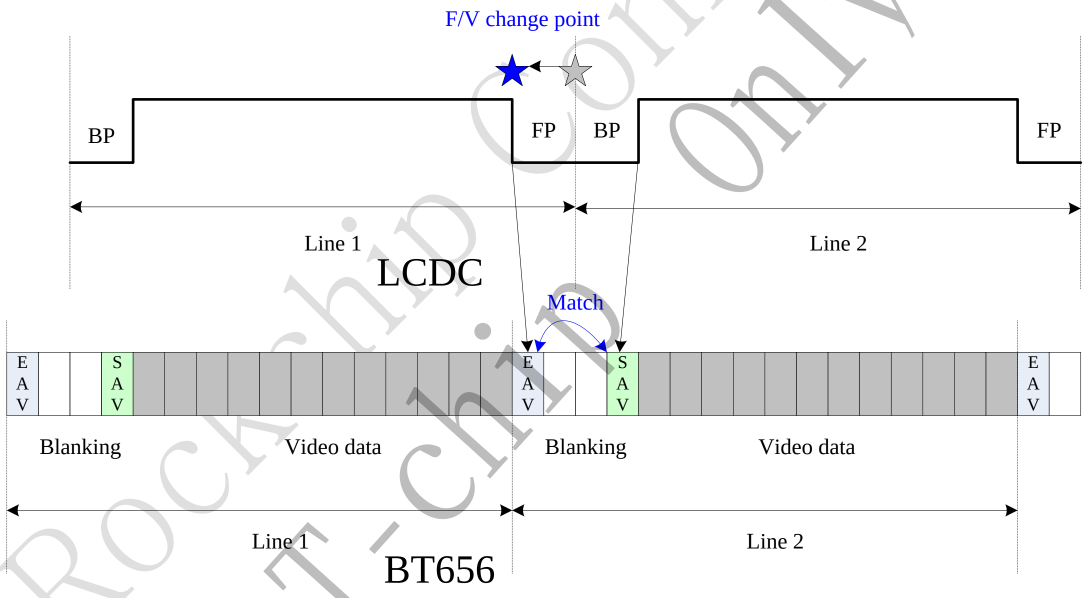
RS delay from CLK rising edge	Trsd				ns
CSN delay from CLK rising edge	Tcsd				ns
WEN delay from CLK rising edge	Twend				ns
REN delay from CLK rising edge	Trend				ns
D_out delay from CLK rising edge	Tdod				ns
D_in setup to REN rising edge	Tdsetup				ns
D_in hold from REN rising edge	Tdhold				ns

Table 27-6 LCDC1 RGB interface signal timing constant

(VDD_core =0.9V to 1.1V, VDD_IO=3.0V to 3.6V , TA = -40℃ ot 125℃)

Item	Symbol	Min	Typ	Max	Unit
Internal clock period	Tclk				ns
RS delay from CLK rising edge	Trsd				ns
CSN delay from CLK rising edge	Tcsd				ns
WEN delay from CLK rising edge	Twend				ns
REN delay from CLK rising edge	Trend				ns
D_out delay from CLK rising edge	Tdod				ns
D_in setup to REN rising edge	Tdsetup				ns
D_in hold from REN rising edge	Tdhold				ns

27.5.3 ITU656 interface timing



27.6 Interface Description

27.6.1 Display interface description

The VOP is suitable for different display mode by different usage, which is shown as follows.

Display mode	***RGB Parallel 24-bit***	***RGB Parallel 18-bit***	***RGB Parallel 16-bit***	***RGB Serial 2x12-bit***	***RGB Serial 3x8-bit***	***RGB Serial 3x8-bit + dummy***
DCLK	DCLK	DCLK	DCLK	DCLK	DCLK	DCLK

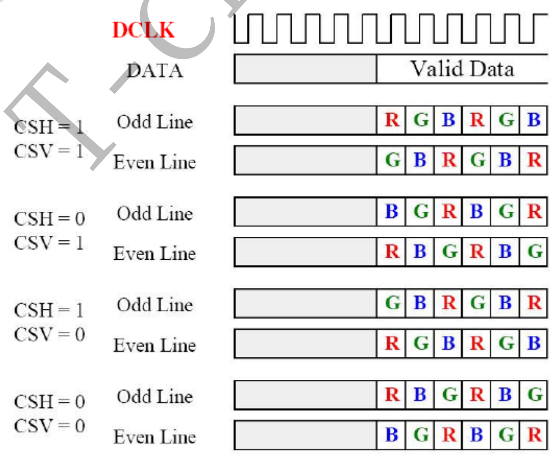
VSYNC	VSYNC	VSYNC	VSYNC	VSYNC	VSYNC	VSYNC
HSYNC	HSYNC	HSYNC	HSYNC	HSYNC	HSYNC	HSYNC
DEN	DEN	DEN	DEN	DEN	DEN	DEN
DATA	DATA[23:0]	DATA[17:0]	DATA[15:0]	DATA[11:0]	DATA[7:0]	DATA[7:0]

Display mode	***MCU Parallel 24-bit***	***MCU Parallel 18-bit***	***MCU Parallel 16-bit***	***MCU Serial 2x12-bit***	***MCU Serial 3x8-bit***
DCLK	RS	RS	RS	RS	RS
VSYNC	CSN	CSN	CSN	CSN	CSN
HSYNC	WEN	WEN	WEN	WEN	WEN
DEN	REN	REN	REN	REN	REN
DATA	DATA[23:0]	DATA[17:0]	DATA[15:0]	DATA[11:0]	DATA[7:0]

Display mode	***ITU656 Mode0***	***ITU656 Mode1***	***ITU656 Mode2***
DCLK	DCLK	DCLK	DCLK
VSYNC	-	-	-
HSYNC	-	-	-
DEN	-	-	-
DATA	DATA[7:0]	DATA[15:8]	DATA[23:16]

In the case of "RGB serial 3x8-bit", there are four scanning modes for the RGB delta data when delta swap is enable, shown as follows.

RGB delta LCD scanning mode	*delta_en*	*dsp_rg_swap*	*dsp_rb_swap*	*dsp_bg_swap*
CSH=1,CSV=1	1	0	1	0
CSH=0,CSV=1	1	0	0	0
CSH=1,CSV=0	1	0	0	1
CSH=0,CSV=0	1	0	1	1



27.6.2 IOMUX description

There are two VOPs in the chip, config GRF_SOC_CON6 register to select a VOP to IO/LVDS port.

programing flow as follow:
step1 : config GRF,iomux,io driver,lcdc select
GRF_GPIO1H_SR and GRF_GPIO1D_E register are optional.
config GRF_BASE + GRF_SOC_CON6 = (0x8<<16) | (0x0) to select vop_big output to IO/LVDS.
config GRF_BASE + GRF_SOC_CON6 = (0x8<<16) | (0x1<<3) to
select vop_lit output to IO/LVDS.
eg:

```
GRF_BASE + GRF_GPIO1D_IOMUX = ((0x55<<16)|(0x55<<0));
GRF_BASE + GRF_GPIO1H_SR      = (0x0f000f00);
GRF_BASE + GRF_GPIO1D_E        = (0x00ff00ff);
GRF_BASE + GRF_SOC_CON6       = (0x8<<16) | (0x0);
GRF_BASE + GRF_SOC_CON7       = (0x1fff<<16) | (0x1840);
```

step 2 : config LVDS PHY0 register to initial LVDS PHY0
eg:

```
LVDS_BASE + 0x00*4 = 0x7f;
LVDS_BASE + 0x01*4 = 0x40;
LVDS_BASE + 0x02*4 = 0x00;
LVDS_BASE + 0x03*4 = 0x46;
LVDS_BASE + 0x04*4 = 0x3f;
LVDS_BASE + 0x05*4 = 0x3f;
LVDS_BASE + 0x0d*4 = 0x0a;
```

step 3 : config LVDS PHY1 register to initial LVDS PHY1
eg:

```
LVDS_BASE + 0x40*4 = 0x7f;
LVDS_BASE + 0x41*4 = 0x40;
LVDS_BASE + 0x42*4 = 0x00;
LVDS_BASE + 0x43*4 = 0x46;
LVDS_BASE + 0x44*4 = 0x3f;
LVDS_BASE + 0x45*4 = 0x3f;
LVDS_BASE + 0x4d*4 = 0x0a;
```

27.7 Application Notes

27.7.1 DMA transfer mode

There are three DMA transfer modes for loading win0 or win1 frame data determined by following parameters(X=0,1,2,3):

dma_burst_length
winX_no_outstanding
winX_gather_en
winX_gather_thres

- **auto outstanding transfer mode(random transfer)**

When winX_no_outstanding is 0, multi-bursts transfer command could be sent out to AXI master interface continuously if the internal memory has enough space to store new data. The continuous random burst number is in the range of 1 to 4, mainly depending on the empty level of internal memory, dma_burst_length, data format and active image width.

- **configured outstanding transfer mode(fixed transfer)**

When winX_gather_en is 1, fixed-number of bursts transfer command should be sent out to AXI master interface continuously if the internal memory has enough space to store new data. The fixed-number is determined by winX_gather_thres. Since the internal memory size is limited, there is some restriction for the winX_gather_thres as follows.

Table 27-7 Gather configuration for all format

Gather Threshold	**dma_burst_length =2'b00(burst16)**	**dma_burst_length =2'b01(burst8)**	**dma_burst_length =2'b10(burst4)**
YUV420 YUV422 YUV444	0	0,1,2	0,1,2,3
ARGB888 RGB888 RGB565	0,1,2,3	0,1,2,3	0,1,2,3
8BPP 4BPP 2BPP 1BPP	0,1,2,3	0,1,2,3	0,1,2,3

27.7.2 Win0/Win1 dma load mode

If you want to improve the efficiency of accessing external memory for loading winX frame data, you could assert winX_dma_load. When winX_dma_load is high, winX frame data is loaded in the unit of line composing with one or more burst transfers; otherwise, loaded in the unit of burst transfer. However, it is not suitable for data format YUV420, no-scaling and active width less than 256.

27.7.3 IEP direct path

There are two data source for win0/win1win2/win3: external memory and IEP internal memory. However, the IEP data is just active for one layer at one frame time when IEP data path is enable, determined by dsp_layer0/1/2/3_sel. Moreover, it is not suitable for win0/1 with scaling, reverse display.

Direct path interface (DPI) can be used for LCDC to display images from other image processing IPs, which also has DPI (slave).

There are programming flows for both DPI display on sequence and DPI display off sequence.

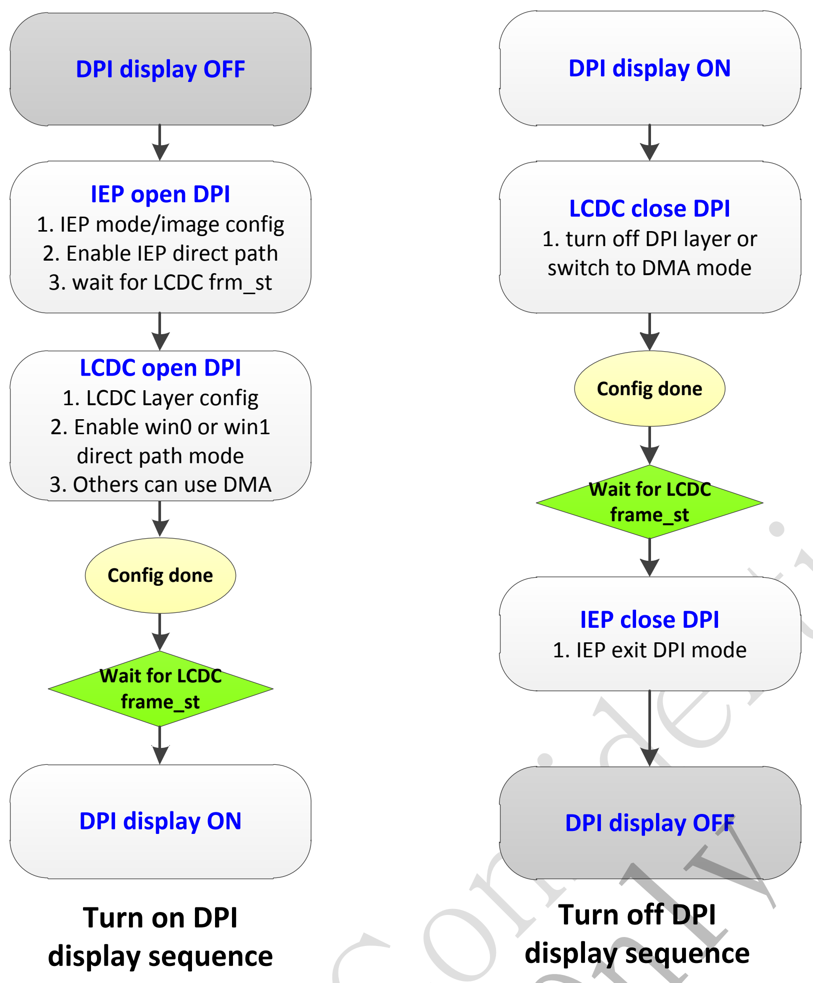


Fig. 27-20 LCDC DPI Programming flow

1.Turn on DPI display

First, configure IEP into DPI mode. After doing image information and image processing mode configuration, IEP DPI mode can be turn on for display. IEP is in idle mode only if LCDC's frame start input signal is valid.

Second, configure LCDC for DPI display. Note that only one layer (Win0 or Win1) can use DPI in same frame. Other layers still can use internal DMA.

Finally, set LCDC "config_done" to confirm all the new configuration and waiting frame sync to start DPI display actually.

2.Turn off DPI display

First, close LCDC layer's DPI mode by turning off DPI layer or switching it to DMA mode. Then set LCDC "config_done" to confirm new configuration.

Second, wait for LCDC's frame synchronization to close DPI display in LCDC.

Finally, turn off IEP's DPI mode.

27.7.4 WIN BPP LUT/GAMMA LUT

WIN1 LUT/DSP LUT should be configured before displaying if win2/3_lut_en/dsp_lut_en is high. You could only update these LUTs by software.

When win1_lut_load_en is 0, the WIN LUT data should be refreshed by software,i.e, writing win1 lut data to the internal memory with the start address WIN1_LUT_MST. The memory size is 256x25, i.e, lower 25bits valid, and the writing data number is determined by software, .

When dsp_lut_load_en is 0, the DSP LUT data should be refreshed by software,i.e, writing dsp lut data to the internal memory with the start address DSP_LUT_MST. The memory size is 256x24, i.e, lower 25bits valid, and the writing data number is determined by software.

27.7.5 DMA QoS request

If you want to get higher priority for VOP to access external memory when the frame data is urgent, a QoS request can be generated and sent out basing on the configured values:

noc_hurry_en
noc_hurry_value
noc_qos_en
noc_win_qos

If noc_qos_en is enable, a win0/1_qos_req is asserted when the empty level of win0/1's linebuffer is greater than the threshold configured in noc_win_qos. And it will be disserted when the empty level is smaller than the threshold or noc_qos_en is disable.

If noc_qos_en is enable, a win0/1_hurry_req is asserted when the empty level of win2/3's fifo is greater than the threshold configured in noc_win_qos. And it will be disserted when the empty level is smaller than the threshold or noc_qos_en is disable.

Either win0/1_qos_req or win2/3_hurry_req is high, a QoS request will be sent out for VOP.

27.7.6 Mirror display

If Y-Mirror display is enable, the frame data is loaded from last line to first line, where the start address of first pixel in last line is defined in

WIN0/1_YRGB0_MST/WIN0/1_CBR0_MST/WIN0/1_YRGB1_MST/WIN0/1_CBR_MST/WIN2/3_MST for win0/1/2/3 respectively.

Otherwise, the win's frame line data width and virtual stride should be 64bit-aligned for 8bit-RGB/YUV or 128bit-aligned for 10bit YUV if X-Mirror or Y-Mirror display is enable.

27.7.7 DDR interface

LCD DDR interface is just suitable for Parallel RGB LCD panel and Serial RGB LCD 2x12 panel. If LCD DDR interface is enable, the timing parameters for LCD panel should be even.

Otherwise, you can synchronize output clock with VSYNC or HSYNC depending on dclk_ddr_sync.

27.7.8 Interrupt

VOP interrupt is comprised of 12 interrupt sources:

frame start interrupt
line flag interrupt
bus error interrupt
win0 empty interrupt
win1 empty interrupt
win2 empty interrupt
win3 empty interrupt
hwc empty interrupt
post empty interrupt
pwm gen interrupt
irq_mmu

Every interrupt has independent interrupt enable (VOP_INT_EN), interrupt clear (VOP_INT_CLR), interrupt status (VOP_INT_STATUS).

27.7.9 RGB display mode

RGB display mode is used for RGB panel display and CCIR656 output. It is a continuous frames display mode.

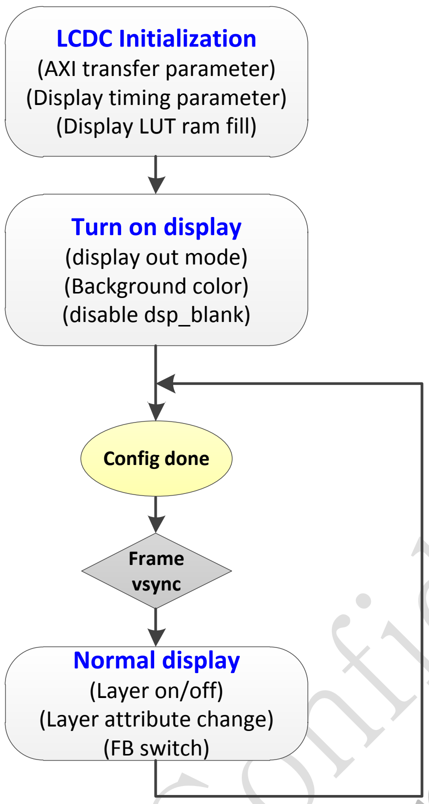


Fig. 27-21 LCDC RGB mode Programming flow

1.LCDC initialization

LCDC initialization should be done before turning display on.

First, AXI bus parameter (LCDC_SYS_CTRL) should be set for DMA transfer.

Second, display panel/interface timing should be set for display output. The registers are: LCDC_DSP_HTOTAL_HS_END/ LCDC_DSP_HACT_ST_END/ LCDC_DSP_VTOTAL_HS_END/ LCDC_DSP_VACT_ST_END/ LCDC_DSP_VS_ST_END_F1/ LCDC_DSP_VACT_ST_END_F1

2.Background display

Before normal display, the background display could be turn on.

First, set display output mode (LCDC_DSP_CTRL0/1) according to display device.

Second, disable dsp_blank mode, which would not be enable until frame synchronization.

Finally, writing '1' to "LCDC_REG_CFG_DONE" register then all the frame-sync registers will be enable at the beginning of next frame.

3.Normal display

In normal display, all the display layers' attribute could be different according display scenario. So there is a programming loop in this mode.

First, configure all the display layers' attribute registers for the change of image format, location, size, scaling factor, alpha and overlay and so on. Those register would not be enable until frame synchronization.

Finally, write '1' to "LCDC_REG_CFG_DONE" register then all the frame-sync registers will be enable at the beginning of next frame.

27.7.10 MCU display mode

MCU display mode is used for MCU panel display or MCU I80 local bus. It is a single frame display mode.

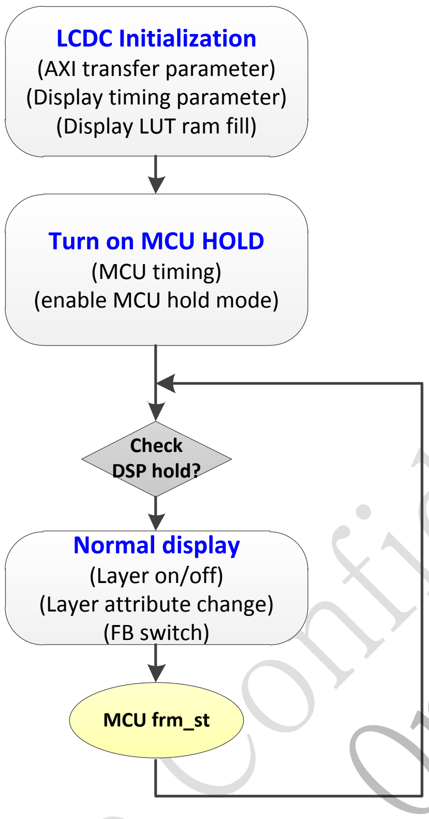


Fig. 27-22 LCDC RGB mode Programming flow

1.LCDC initialization

LCDC initialization should be done before turning display on.

First, AXI bus parameter (LCDC_SYS_CTRL) should be set for DMA transfer.

Second, display panel/interface timing should be set for display output. The registers are: LCDC_DSP_HTOTAL_HS_END/ LCDC_DSP_HACT_ST_END/ LCDC_DSP_VTOTAL_HS_END/ LCDC_DSP_VACT_ST_END/ LCDC_DSP_VS_ST_END_F1/ LCDC_DSP_VACT_ST_END_F1

Finally, fill the display LUT ram if color LUT function enable.

2.Turn on MCU hold mode

First setting MCU timing parameter (LCDC_MCU_CTRL[26:0]).

Turn on MCU hold mode (LCDC_MCU_CTRL[31] and LCDC_MCU_CTRL[27]), then wait for MCU display hold (read LCDC_MCU_CTRL[28] if it's value is '1', or set display hold valid interrupt)

3.Single display

If MCU display hold status is valid, single MCU display frame could be start by setting mcu_frame_st (LCDC_MCU_CTRL[28])

First, configure all the display layers' attribute registers for the change of image format, location, size, scaling factor, alpha and overlay and so on. Those register would not be enable until MCU frame start.

Second, write '1' to start one MCU frame.

Finally, wait for MCU display hold status.

27.7.11 MIPI control

1.double channel

MIPI double channel display is supported in the version(only left-right type).

Config doub_channel_en register in DSP_CTRL0 to adapt double channel display .

When doub_channel mode ,the vop will output two data bus to MIPI PHY,data0 is left panel data,data1 is right panel data.

Double channel overlap display is supported at the same time.

normal mode:

left panel data is from 0 to (width/2 -1).

right panel data is from width/2 to width-1.

overlap mode:

left panel data is from 0 to (width/2 -1+overlap number).

right panel data is from width/2-overlap number to width-1.

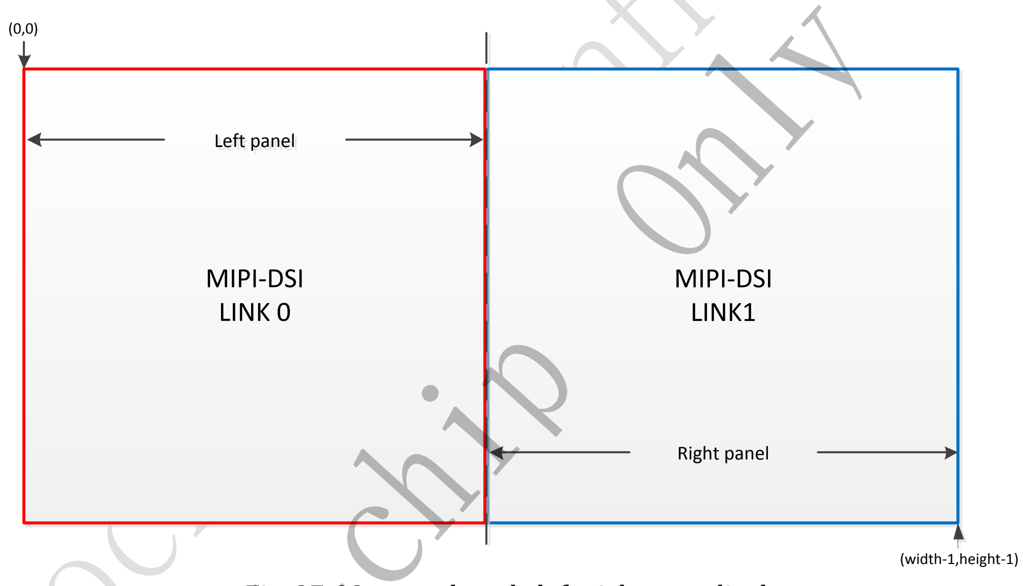


Fig. 27-23 normal mode left-right type display

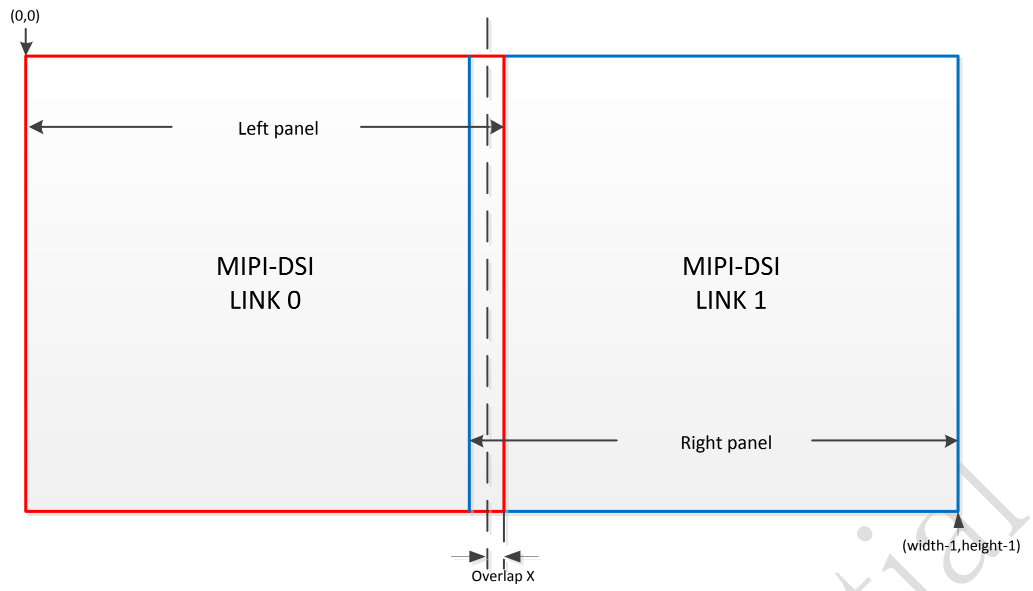


Fig. 27-24 overlap mode left-right type display

The overlap number equal double_ch_overlap_num value *2,in the range of 0~16.

2.halt mode

Mipi halt fuction is supported in this version, detail configuration reference MIPI-DSI chapter.

3.command mode flow

Mipi command mode is supported in this version .

There is programming flows for command mode .

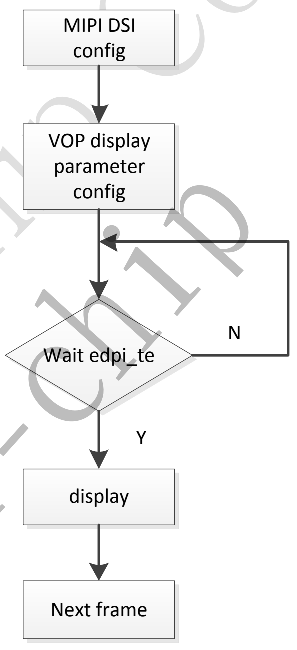


Fig. 27-25 command mode flow

27.7.12 Immediately control register

There are two type register in VOP , one type is effective immediately,the other is effective by frame sync.
Effective immediately registers list as follows,other registers are all effective by frame sync.

Table 27-8 effective immediately register table

register address	description

0x008[23:21],0x008[15:8]	some dsp ctrl function bit
0x00c	sys ctrl1 register
0x018	background color register
0x01c	mcu ctrl register
0x038	win0 color key register
0x078	win1 color key register
0x0cc	win2 color key register
0x11c	win3 color key register
0x188~0x19c	dsp_timing ctrl registers
0x1a0~0x1a8	pwm ctrl registers
0x1c8~0x1dc	cabc_gauss_parameter registers
0x1ec~0x1f4	frc pattern parameter registers