Crystal Image through Imaging Innovation

PIXELPLUS

SURROUND VIEW MONITORING SYSTEM

PI5008K Video I/O User Guide

Rev 0.4

Last Update : 2018.12.27

6th Floor, 105, Gwanggyo-ro, Yeongtong-gu,

Suwong-si, Gyeonggi-do, 16229, Korea

Tel : +82-31-888-5300, FAX : +82-31-888-5399

Contents

Figure

1. Video I/O User Guide

1.1. Video Input

PI5008K SDK makes it possible to select various video inputs.

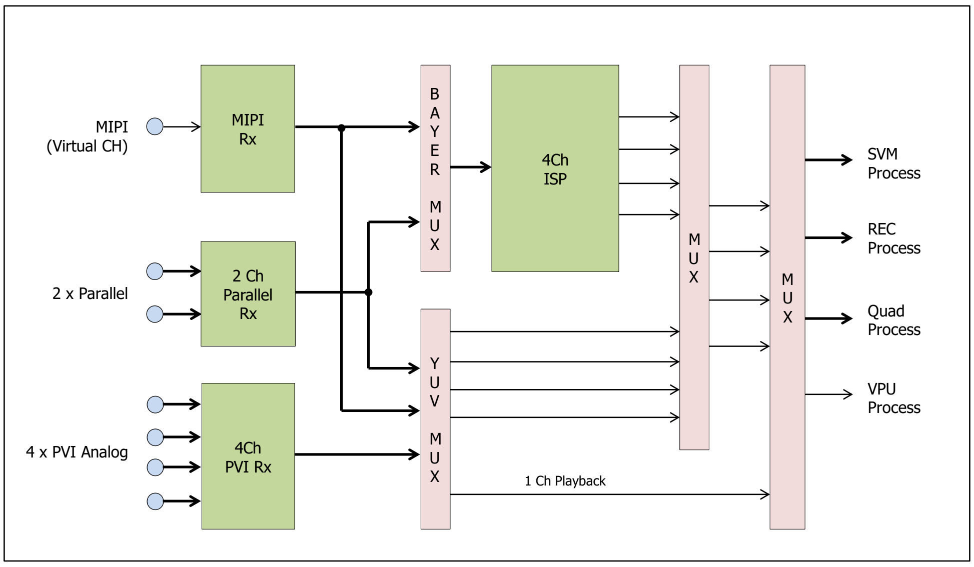


Figure 1 Block Diagram of Video Input Interface

Notes> Video input cannot be changed during run-time

1.1.1. Configure Video input:

1.1.1.1. Options for each feature

A. Video input type

: Camera video input type

Video input type	Description
VIDEO_IN_TYPE_MIPI_BAYER	Video input type is MIPI bayer type
VIDEO_IN_TYPE_MIPI_YUV	Video input type is MIPI yuv type
VIDEO_IN_TYPE_PVI	Video input type is PVI(analog)

B.MIPI input format

: MIPI bayer/yuv raw bit format

MIPI input format	Description
MIPI_VID_BIT_RAW6	Bayer raw 6bit
MIPI_VID_BIT_RAW7	Bayer raw 7bit
MIPI_VID_BIT_RAW8	Bayer raw 8bit
MIPI_VID_BIT_RAW10	Bayer raw 10bit
MIPI_VID_BIT_RAW12	Bayer raw 12bit
MIPI_VID_BIT_RAW14	Bayer raw 14bit
MIPI_VID_BIT_YUV8_2XRAW8	YUV 8bit or Bayer raw 16bit
MIPI_VID_BIT_2XRAW10	Bayer raw 20bit

C. Video signal format

: Video signal format: yuv 8/16, embedded/external sync, RGB, bayer 8/10

Video signal format	Description
VID_TYPE_YC8_EMB	YUV 8bit embedded sync
VID_TYPE_YC8_EXT	YUV 8bit external sync
VID_TYPE_YC16_EMB	YUV 16bit embedded sync
VID_TYPE_YC16_EXT	YUV 16bit external sync
VID_TYPE_RGB24	RGB 24bit
VID_TYPE_BAYER_8BIT	Bayer 8bit
VID_TYPE_BAYER_10BIT	Bayer 10bit

D. Video frame

: Video frame 25/30/50/60

Video frame	Description
VID_FRAME_NTSC_30	30frame(NTSC)
VID_FRAME_PAL_25	25frame(PAL)
VID_FRAME_NTSC_60	60frame(NTSC)
VID_FRAME_PAL_50	50frame(PAL)

E. Video resolution

: Video resolution

Video resolution	Description

VID_RESOL_SD720H	CVBS 720x480i, 720x576i
VID_RESOL_SD960H	CVBS 960x480i, 960x576i
VID_RESOL_HD720P	1280x720p
VID_RESOL_HD960P	1280x960p
VID_RESOL_HD1080P	1920x1080p
VID_RESOL_HD800_480P	800x480p
VID_RESOL_HD1024_600P	1024x600p

F. Analog Video Standard

: Analog video standard

Analog video standard	Description
VID_STANDARD_CVBS	CVBS (720x480i, 720x576i, 960x480i, 960x576i)
VID_STANDARD_PVI	Analog HD PVI(pixelplus) standard.
VID_STANDARD_CVI	Analog HD CVI(Dahwa) standard.
VID_STANDARD_HDT	Analog HD TVI(HikVision) standard.
VID_STANDARD_HDA	Analog HD AHD(Nextchip) standard.

G. PARALLEL Video Type

: Parallel digital video type.

PARALLEL video type	Description
VID_PARALLEL_TYPE_VIN_BAYER	Bayer video type
VID_PARALLEL_TYPE_VIN_YUV	YUV video type

H. Analog Tx source

: Analog tx (PVITX) source type

Analog Tx source	Description
PVITX_SRC_NONE	Don’t use Analog TX(PVITX)
PVITX_SRC_DU	Analog TX(PVITX) source is DU.
PVITX_SRC_QUAD	Analog TX(PVITX) source is QUAD.

1.1.1.2. Set Video input type

Select video input type. There are 3 types; MIPI_BAYER, MIPI_YUV, PVI[Analog HD Camera (with SD)]

Ex> Camera is MIPI Bayer type.

SDK/config/board_config.h

```
#define VIDEO_IN_TYPE                    (VIDEO_IN_TYPE_MIPI_BAYER)
```

1.1.1.3. Set Video input format

Select video input frame rate and resolution.

Ex> Camera is 1280x720p25.

SDK/config/board_config.h

```
#define BD_VIN_FMT (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
```

1.1.1.4. Set Camera input format

Select camera input format. If camera is MIPI, set raw bit. If camera is PVI, set standard.

Ex> If camera is MIPI_Bayer, raw 12bit :

SDK/config/board_config.h

```
#define BD_CAMERA_IN_FMT (BD_VIN_FMT | MIPI_VID_BIT_RAW12)
```

Ex> If camera is PVI, HDA :

SDK/config/board_config.h

```
#define BD_CAMERA_IN_FMT (BD_VIN_FMT | VID_STANDARD_HDA)
```

1.1.1.5. Set SVM output format

Select SVM h/w module output frame rate and resolution.

Ex> SVM h/w module output is 1280x720p25.

SDK/config/board_config.h

```
#define BD_SVM_OUT_FMT (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
```

1.1.2. Set camera input port

1.1.2.1. Set camera input port

Select camera input port. It is supported from SDK v1.11 above.

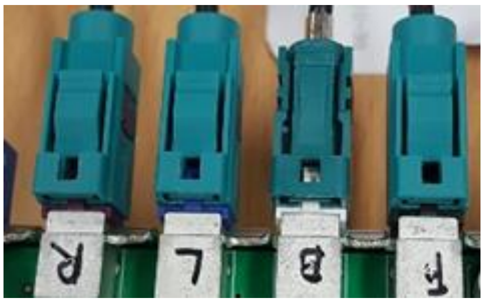
Ex> Front-port0, Left-port1, Right-port2, Back(rear)-port3.

SDK/drivers/vin/vin_user_config.c

```
pVin->vidPort[0] = vinVidInPort0;
pVin->vidPort[1] = vinVidInPort1;
pVin->vidPort[2] = vinVidInPort2;
```

```
pVin->vidPort[3] = vinVidInPort3;
```

Ex> Front-port0, Left-port2, Right-port3, Back(rear)-port1.



SDK/drivers/vin/vin_user_config.c

```
pVin->vidPort[0] = vinVidInPort0;
pVin->vidPort[1] = vinVidInPort2;
pVin->vidPort[2] = vinVidInPort3;
pVin->vidPort[3] = vinVidInPort1;
```

1.1.3. APIs

1.1.3.1. VIN

1.1.3.1.1.PPAPI_VIN_Initialize

Prototype	PP_VOID PPAPI_VIN_Initialize(PP_VOID);
Description	Initialize VIN h/w module
Argument	None
Return value	None
Remark	

1.1.3.1.2.PP_VOID PPAPI_VIN_SetGenlockParam

Prototype	PP_VOID PPAPI_VIN_SetGenlockParam(PP_VOID);
Description	Set genlock parameters of VIN h/w module
Argument	None
Return value	None
Remark	

1.1.3.1.3.PPAPI_VIN_GetResol

Prototype	PP_RESULT_E PPAPI_VIN_GetResol(const PP_S32 defVideoFmt, PP_S32 *pRetWidth, PP_S32 *pRetHeight, _VID_RESOL *peRetResol);
Description	Get video resolution (Width, Height, resolution) from configuration define.
Argument	defVideoFmt : Board configuration define. pRetWidth : Size of width. pRetHeight : Size of height. pRetResol : resolution of video.
Return value	eERROR_NOT_SUPPORT: eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	Not used

[Data Types]

_VID_RESOL

[Description]

Define the video resolution.

[Syntax]

```
typedef enum {
      vres_720x480i60 = 0,  // 0
      vres_720x576i50,              // 1
      vres_960x480i60,              // 2
      vres_960x576i50,              // 3

      vres_1280x720p60,             // 8
      vres_1280x720p50,             // 9
      vres_1280x720p30,             //10
      vres_1280x720p25,             //11
      vres_1280x960p30,             //12
      vres_1280x960p25,             //13
      vres_1920x1080p30,            //14
      vres_1920x1080p25,            //15
      vres_800x480p60,              //16  ----- don't suppot pvi
```

```
        vres_800x480p50,                //17  ----- don't suppot pvi
        vres_1024x600p60,               //18  ----- don't suppot pvi
        vres_1024x600p50,               //19  ----- don't suppot pvi
        max_vid_resol
}_VID_RESOL;
```

Member	Description
vres_720x480i60	720x480i60.
vres_720x576i50	720x576i50.
vres_960x480i60	960x480i60.
vres_960x576i50	960x576i50.
vres_1280x720p60	1280x720p60
vres_1280x720p50	1280x720p50
vres_1280x720p30	1280x720p30
vres_1280x720p25	1280x720p25
vres_1280x960p30	1280x960p30
vres_1280x960p25	1280x960p25
vres_1920x1080p30	1920x1080p30.
vres_1920x1080p25	1920x1080p25.
vres_800x480p60	800x480p60. Don't support on PVI-Tx
vres_800x480p50	800x480p50. Don't support on PVI-Tx
vres_1024x600p60	1024x600p60. Don't support on PVI-Tx
vres_1024x600p50	1024x600p50. Don't support on PVI-Tx

1.1.3.1.4.PPAPI_VIN_SetQuadViewMode

Prototype	PP_RESULT_E PPAPI_VIN_SetQuadViewMode(const PP_S32 defVideoFmt, const PP_U8 bQuadView, const PP_S32 chSel, const PP_S32 pathSel);
Description	Set Quad h/w module. Quad h/w module be used to Quad view mode or full Image channel capture.
Argument	defVideoFmt :Board configuration define. bQuadView : 1: Quad view mode, 0: Full image channel capture. chSel : Quad start channel or capture channel. If bQuadView == 0, capture channel. If bQuadView == 1, select quad view mode.

	pathSel : Select Quad input path.(default 0)
Return value	eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.5.PPAPI_VIN_SetCaptureMode

Prototype	PP_RESULT_E PPAPI_VIN_SetCaptureMode(const PP_S32 defVideoFmt, const PP_S32 chSel, const PP_S32 pathSel);
Description	Set capture mode and be ready capture status.
Argument	defVideoFmt: Board configuration define. chSel: Capture channel. pathSel: Select Quad input path.(default 0)
Return value	eERROR_TIMEOUT: eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.6.PPAPI_VIN_GetCaptureImage

Prototype	PP_RESULT_E PPAPI_VIN_GetCaptureImage(const PP_S32 defVideoFmt, const PP_S32 bYOnly, PP_U32 u32BufPAddr, PP_U32 *pRetBufSize);
Description	Get capture image.
Argument	defVideoFmt: Board configuration define. bYOnly:Select Y only or YUV image. pRetBufSize: pointer of capture image buffer.
Return value	eERROR_TIMEOUT: eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.1.PPAPI_VIN_SetCaptureUserMode

Prototype	PP_RESULT_E PPAPI_VIN_SetCaptureUserMode(const PP_S32 defVideoFmt, const PP_U32 sclWidth, const PP_U32 sclHeight, const PP_U8 chSelBit);
Description	Set capture mode and be ready capture status with scale down image.
Argument	defVideoFmt: Board configuration define. sclWidth: scale size of width. Only support down scale. sclHeight: scale size of height. Only support down scale. chSelBit: Capture channel bit. 0001b:ch0, 0010b:ch1, 0100b:ch2..
Return value	eERROR_TIMEOUT: eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.2.PPAPI_VIN_GetCaptureUserImage

Prototype	PP_RESULT_E PPAPI_VIN_GetCaptureUserImage(const PP_S32 defVideoFmt, const PP_U32 sclWidth, const PP_U32 sclHeight, const PP_U8 chSelBit, PP_U32 *pu32BufPAddr, PP_U32 *pRetBufSize);
Description	Get capture image with scale down.
Argument	defVideoFmt: Board configuration define. sclWidth: scale size of width. Only support down scale. sclHeight: scale size of height. Only support down scale. pu32BufPAddr : physical address of capture buffer. pRetBufSize: pointer of capture image buffer.
Return value	eERROR_TIMEOUT: eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.3.PPAPI_VIN_EnableQuad

Prototype	PP_VOID PPAPI_VIN_EnableQuad(const PP_BOOL bEnable);
Description	Start / Stop Quad h/w module.
Argument	bEnable:TRUE : enable, FALSE:disable
Return value	None
Remark	

1.1.3.1.1.PPAPI_VIN_SetVIDPort

Prototype	PP_RESULT_E PPAPI_VIN_SetVIDPort(const PP_S8 s8VinPort0, const PP_S8 s8VinPort1, const PP_S8 s8VinPort2, const PP_S8 s8VinPort3, const PP_S8 s8VinPort4);
Description	Set Camera input VID port.
Argument	s8VinPort0: VID port0 of camera channel number. s8VinPort1: VID port1 of camera channel number. s8VinPort2: VID port2 of camera channel number. s8VinPort3: VID port3 of camera channel number. s8VinPort4: VID port4 of playback channel number.
Return value	eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.1.PPAPI_VIN_SetSVMChannel

Prototype	PP_RESULT_E PPAPI_VIN_SetSVMChannel(const PP_S8 s8SvmChannel, const PP_S8 s8SvmPath, const PP_S8 s8SvmPort);
Description	Set SVM Channel input.
Argument	s8SvmChannel: SVM Input channel(0~3). s8SvmPath: SVM Input path. 0: Video, 1:TestPattern s8SvmPort: SVM Input port. If s8SvmPath = 0(Video), Video port(0~3),

	If s8SvmPath = 1(TestPattern), 2
Return value	eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.1.PPAPI_VIN_SetROChannel

Prototype	PP_RESULT_E PPAPI_VIN_SetROChannel(const PP_U8 u8ROPath, const PP_U8 b8Bit, const PP_S8 s8OutCh0, const PP_S8 s8OutCh1, const PP_S8 s8OutCh2, const PP_S8 s8OutCh3);
Description	Set RO output channel.
Argument	u8ROPath: Select RO path(0 or 1). b8Bit: Select 8bit or 16bit output. If 16bit output, ignore u8ROPath value. s8OutCh0: output channel0 number. Ignore: -1 s8OutCh1: output channel1 number. Ignore: -1 s8OutCh2: output channel2 number. Ignore: -1 s8OutCh3: output channel3 number. Ignore: -1
Return value	eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.1.1.PPAPI_VIN_GetInputInfo

Prototype	PP_RESULT_E PPAPI_VIN_GetInputInfo(const PP_U8 u8Channel, PP_U32 *pRetInputInfo);
Description	Get Video input information.
Argument	u8Channel: Input Video number(0~3). pRetInputInfo: pointer of return information.
Return value	eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

[Data Types]

pRetInputInfo

[Description]

Return input Video information.

[Syntax]

```
pRetInputInfo[0] //VIN_SYNC_VIN_HSIZE_INFO_CONFIG_U
 pRetInputInfo[1] //VIN_SYNC_VIN_FSIZE_INFO_CONFIG_U
 pRetInputInfo[2] //VIN_SYNC_VIN_HVACT_INFO_CONFIG_U
```

1.1.3.1.2.PPAPI_VIN_DiagCameraInput

Prototype	PP_RESULT_E PPAPI_VIN_DiagCameraInput(const PP_U8 u8Channel);
Description	Diagnosis camera input.
Argument	u8Channel: Input camera number(0~3).
Return value	eERROR_INVALID_ARGUMENT: eSUCCESS
Remark	

1.1.3.2. Analog Rx(PVI)

1.1.3.2.1.PPAPI_PVIRX_CheckChipID

Prototype	PP_RESULT_E PPAPI_PVIRX_CheckChipID(const PP_U8 IN chanAddr, PP_U16 OUT *pRetChipID, PP_U8 OUT *pRetRevID, PP_S32 OUT *pRetRWVerify);
Description	Get current Chip pvi rx ID and register access verify status.
Argument	chanAddr: channel ID(0~3) pRetChipID: Return chip ID (0x2000) pRetRevID: Return Rev ID (1) pRetRWVerify: 1: success, else:failure.
Return value	eERROR_FAILURE: eSUCCESS:
Remark	

1.1.3.2.2.PPAPI_PVIRX_SetAttrChip

Prototype	PP_RESULT_E PPAPI_PVIRX_SetAttrChip(const PP_U8 IN chanAddr, const _stAttrChip IN *pstPviRxAttrChip);
Description	Set pvi rx h/w input attribute parameters.
Argument	chanAddr: channel ID(0~3) pstPviRxAttrChip:pointer of attribute parameters.
Return value	eERROR_FAILURE: eSUCCESS:
Remark	

[Data Types]

_stAttrChip

[Description]

Define the pvi rx h/w module attribute.

[Syntax]

```
typedef struct
{
      uint8_t chanAddr;
      uint8_t vinMode;
}_stAttrChip;
```

Member	Description
chanAddr	Channel ID (0~3)
vinMode	Select PVI-Rx input pin: 1:Single VinP pin(default), 3:Single VinN pin, 0:Differential VinPN pin.

1.1.3.2.3.PPAPI_PVIRX_SetTableStdResol

Prototype	PP_RESULT_E PPAPI_PVIRX_SetTableStdResol(const PP_U8 IN chanAddr, const enum _eCameraStandard IN cameraStandard, const enum _eCameraResolution IN cameraResolution, const enum _eVideoResolution IN videoResolution, const PP_S32 IN bWaitStableStatus);
Description	Set pvi rx h/w module as standard, resolution.
Argument	chanAddr: channel ID(0~3)

	cameraStandard: Camera standard type. cameraResolution: Camera resolution type. videoResolution: Video resolution type. bWaitStableStatus: Wait stable status after set parameters.
Return value	eERROR_FAILURE: eSUCCESS:
Remark	

[Data Types]

_eCameraStandard

[Description]

Define the pvi rx standard format.

[Syntax]

```
enum _eCameraStandard {
        CVBS = 0,
        PVI,
        CVI,
        HDA,
        HDT_OLD,
        HDT_NEW,
        max_camera_standard
};
```

Member	Description
CVBS	Cvbs (NTSC, PAL)
PVI	Analog HD PVI(pixelplus) standard.
CVI	Analog HD CVI(Dahwa) standard.
HAD	Analog HD AHD(Nextchip) standard.
HDT_OLD	Analog HD TVI(HikVision) Old standard(reserved)
HDT_NEW	Analog HD TVI(HikVision) New standard.

[Data Types]

_eCameraResolution

[Description]

Define the camera resolution.

[Syntax]

```
enum _eCameraResolution {
        camera_ntsc = 0,
        camera_pal,

        camera_1280x720p60,
        camera_1280x720p50,
        camera_1280x720p30,
        camera_1280x720p25,
        camera_1920x1080p30,
        camera_1920x1080p25,

        camera_1280x960p30,
        camera_1280x960p25,

        max_camera_resolution
};;
```

Member	Description
`Camera_ntsc`	NTSC
`Camera_pal`	PAL
`camera_1280x720p60`	1280x720p60
`camera_1280x720p50`	1280x720p50
`camera_1280x720p30`	1280x720p30
`camera_1280x720p25`	1280x720p25
`camera_1920x1080p30`	1920x1080p30
`camera_1920x1080p25`	1920x1080p25
`camera_1280x960p30`	1280x960p30
`camera_1280x960p25`	1280x960p25

[Data Types]

_eVideoResolution

[Description]

Define the video resolution.

[Syntax]

```
enum _eVideoResolution {
        video_720x480i60 = 0,
        video_720x576i50,
        video_960x480i60,
        video_960x576i50,
        video_1280x720p60,
        video_1280x720p50,
        video_1280x720p30,
        video_1280x720p25,
        video_1920x1080p30,
        video_1920x1080p25,

        video_1280x960p30,
        video_1280x960p25,

        max_video_resolution
};
```

Member	Description
`video_720x480i60`	720x480i60
`video_720x576i50`	720x576i50
`video_960x480i60`	960x480i60
`video_960x576i50`	960x576i50
`video_1280x720p60`	1280x720p60
`video_1280x720p50`	1280x720p50
`video_1280x720p30`	1280x720p30
`video_1280x720p25`	1280x720p25
`video_1920x1080p30`	1920x1080p30
`video_1920x1080p25`	1920x1080p25
`video_1280x960p30`	1280x960p30
`video_1280x960p25`	1280x960p25

1.1.3.2.4.PPAPI_PVIRX_SetNovidInitIRQ

Prototype	PP_RESULT_E PPAPI_PVIRX_SetNovidInitIRQ(const PP_U8 IN chanAddr);
Description	Initialize pvi rx novideo irq parameters.
Argument	chanAddr: channel ID(0~3)
Return value	eERROR_FAILURE: eSUCCESS:
Remark	

1.1.3.2.5.PPAPI_PVIRX_SetInit

Prototype	PP_RESULT_E PPAPI_PVIRX_SetInit(const PP_U8 IN chanAddr);
Description	Initialize pvi rx channel. Standard, resolution, UTC, IRQ. Etc.
Argument	chanAddr: channel ID(0~3)
Return value	eERROR_FAILURE: eSUCCESS:
Remark	

1.1.3.2.6.PPAPI_PVIRX_ReadVidStatusReg

Prototype	PP_RESULT_E PPAPI_PVIRX_ReadVidStatusReg(const PP_U8 IN chanAddr, _stPVIRX_VidStatusReg OUT *pstVidStatusReg);
Description	Get current pvi rx status registers information.
Argument	chanAddr: channel ID(0~3) pstVidStatusReg:
Return value	eERROR_FAILURE: eSUCCESS:
Remark	

[Data Types]

_stPVIRX_VidStatusReg

[Description]

Pvi rx stuatus registers.

[Syntax]

```
typedef union
{
        uint8_t reg[3];
        struct
        {
                uint8_t det_ifmt_res:3;
                uint8_t det_video:1;
                uint8_t det_ifmt_ref:2;
                uint8_t det_ifmt_std:2;
                uint8_t det_chroma:1;
                uint8_t lock_chroma:1;
                uint8_t lock_c_fine:1;
                uint8_t lock_hpll:1;
                uint8_t lock_hperiod:1;
                uint8_t lock_clamp:1;
                uint8_t lock_gain:1;
                uint8_t lock_std:1;

                uint8_t reserved0:2;
                uint8_t det_std_hda:1;
                uint8_t det_std_hdt_h0:1;
                uint8_t det_std_hdt_h1:1;
                uint8_t det_std_hdt_v:1;
                uint8_t reserved1:2;
        }b;
}_stPVIRX_VidStatusReg;
```

Member	Description
`det_ifmt_res`	Status Information of Detected Video Input Resolution 0 : SD 480i 1 : SD 576i 2 : HD720p 3 : HD1080p 4 : HD960p or HD800p
`det_video`	Status Information of Video Detection 0 : Not Detected 1 : Detected
`det_ifmt_ref`	Status Information of Detected Video Input Refresh Rate 0 : 25Hz 1 : 30Hz 2 : 50Hz 3 : 60Hz
`det_ifmt_std`	Status Information of Detected Video Input Standard 0 : HD-PVI 1 : HD-CVI 2 : HDA

Member	Description
	3 : HDT
`det_chroma`	Status of Chroma Detection 0 : Not Detected 1 : Detected
`lock_chroma`	Coarse Lock Status of Chroma Phase Tracking Loop 0 : Not Locked 1 : Locked
`lock_c_fine`	Fine Lock Status of Chroma Phase Tracking Loop 0 : Not Locked 1 : Locked
`lock_hpll`	Lock Status of Horizontal PLL Loop 0 : Not Locked 1 : Locked
`lock_hperiod`	
`lock_clamp`	Lock Status of Clamp Loop 0 : Not Locked 1 : Locked
`lock_gain`	Lock Status of Gain Loop 0 : Not Locked 1 : Locked
`lock_std`	Lock Status of Video Standard Detection 0 : Not Detected 1 : Detected
`det_std_hda`	Detect HDA standard format
`det_std_hdt_h0`	Detect HDT standard format specific horizontal signal feature.
`det_std_hdt_h1`	Detect HDT standard format specific horizontal signal feature.
`det_std_hdt_v`	Detect HDT standard format specific vertical signal feature.

1.1.3.2.7.PPAPI_PVIRX_MonitorCurVidStatusReg

Prototype	PP_RESULT_E PPAPI_PVIRX_MonitorCurVidStatusReg(const PP_U8 IN chanAddr, _stPVIRX_VidStatusReg OUT *pstVidStatusReg);
Description	Get currnet pvi rx status registers and parsing information.

Argument	chanAddr: channel ID(0~3) pstVidStatusReg:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.8.PPAPI_PVIRX_ReadStdResol

Prototype	PP_RESULT_E PPAPI_PVIRX_ReadStdResol(const PP_U8 IN chanAddr, const _stPVIRX_VidStatusReg IN *pstVidStatusReg, enum _eCameraStandard OUT *pCameraStandard, enum _eCameraResolution OUT *pCameraResolution, enum _eVideoResolution OUT *pVideoResolution);
Description	Get basic information value as standard, resolution from _stPVIRX_VidStatusReg.
Argument	chanAddr: channel ID(0~3) pstVidStatusReg pCameraStandard: Camera standard type. pCameraResolution: Camera resolution type. pVideoResolution: Video resolution type.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.9.PPAPI_PVIRX_GetStdResol

Prototype	PP_RESULT_E PPAPI_PVIRX_GetStdResol(const PP_U8 IN chanAddr, _stPVIRX_VidStatusReg IN *pstVidStatusReg, enum _eCameraStandard OUT *pCameraStandard, enum _eCameraResolution OUT *pCameraResolution, enum _eVideoResolution OUT *pVideoResolution, int OUT *pReJudge);
Description	Processing and Correct information value as standard, resolution from _stPVIRX_VidStatusReg.
Argument	chanAddr: channel ID(0~3)

	pstVidStatusReg: pCameraStandard: Camera standard type. pCameraResolution: Camera resolution type. pVideoResolution: Video resolution type. pRejudge: Rejudge camera data.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.10.PPAPI_PVIRX_VID_SetChnAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetChnAttr(const PP_U8 IN chanAddr, const _stChnAttr IN *pstChnAttr);
Description	Set video input control parameter.
Argument	chanAddr: channel ID(0~3) pstChnAttr:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Types]

_stChnAttr

[Description]

Define the video H/V active size and delay.

[Syntax]

```
typedef struct
{
       uint16_t u16HActive; //b[12:0]
       uint16_t u16HDelay; //b[12:0]
       uint16_t u16VActive; //b[10:0]
       uint16_t u16VDelay; //b[10:0]
       uint16_t u16HSCLRatio; //b[15:0] 0:skip write

}_stChnAttr;
```

Member	Description
`u16HActive`	Video horizontal active size. b[12:0]

Member	Description
`u16Hdelay`	Video horizontal delay size. b[12:0]
`u16Vactive`	Video vertical active size. b[10:0]
`u16Vdelay`	Video vertical delay size. b[10:0]
`u16HSCLRatio`	Video horizontal scale ratio. b[15:0]. 0:skip write. use default.

1.1.3.2.11.PPAPI_PVIRX_VID_GetChnAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetChnAttr(const PP_U8 IN chanAddr, _stChnAttr OUT *pstChnAttr);
Description	Get video input control parameter.
Argument	chanAddr: channel ID(0~3) pstChnAttr:
Return value	eERROR_FAILURE: eSUCCESS
Remark	Reference [Data Types] _stChnAttr

1.1.3.2.12.PPAPI_PVIRX_VID_SetCscAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetCscAttr(const PP_U8 IN chanAddr, const _stCscAttr IN *pstCscAttr);
Description	Set video input Cb/Cr parameter.
Argument	chanAddr: channel ID(0~3) pstCscAttr:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Types]

_stCscAttr

[Description]

Define the video Cb/Cr gain and offset.

[Syntax]

```
typedef struct
{
        uint8_t u8CbGain;
        uint8_t u8CrGain;
        uint8_t u8CbOffset;
        uint8_t u8CrOffset;
}_stCscAttr;
```

Member	Description
`u8CbGain`	Video Cb gain.
`u8CrGain`	Video Cr gain.
`u8CbOffset`	Video Cb offset.
`u8CrOffset`	Video Cr offset.

1.1.3.2.13.PPAPI_PVIRX_VID_GetCscAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetCscAttr(const PP_U8 IN chanAddr, _stCscAttr OUT *pstCscAttr);
Description	Get video input Cb/Cr parameter.
Argument	chanAddr: channel ID(0~3) pstCscAttr:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.14.PPAPI_PVIRX_VID_SetContrast

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetContrast(const PP_U8 IN chanAddr, const _stContrast IN *pstContrast);
Description	Set video contrast value.
Argument	chanAddr: channel ID(0~3) pstContrast:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Tyeps]

_stContrast

[Description]

Define the video contrast value.

[Syntax]

```
typedef struct
{
        uint8_t u8Contrast;
}_stContrast;
```

Member	Description
`u8Contrast`	Video contrast value.(0~255)

1.1.3.2.15.PPAPI_PVIRX_VID_GetContrast

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetContrast(const PP_U8 IN chanAddr, _stContrast OUT *pstContrast);
Description	Get video contrast value.
Argument	chanAddr: channel ID(0~3) pstContrast:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.16.PPAPI_PVIRX_VID_SetBright

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetBright(const PP_U8 IN chanAddr, const _stBright IN *pstBright);
Description	Set video brightness value.
Argument	chanAddr: channel ID(0~3) pstBright:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Types]

_stBright

[Description]

Define the video brightness value.

[Syntax]

```
typedef struct
```

```
{
       uint8_t u8Bright;
}_stBright;
```

Member	Description
`u8Bright`	Video brightness value.(0~255)

1.1.3.2.17.PPAPI_PVIRX_VID_GetBright

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetBright(const PP_U8 IN chanAddr, _stBright OUT *pstBright);
Description	Get video brightness value.
Argument	chanAddr: channel ID(0~3) pstBright:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.18.PPAPI_PVIRX_VID_SetSaturation

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetSaturation(const PP_U8 IN chanAddr, const _stSaturation IN *pstSaturation);
Description	Set video saturation value.
Argument	chanAddr: channel ID(0~3) pstSaturation:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Tyeps]

_stSaturation

[Description]

Define the video saturation value.

[Syntax]

```
typedef struct
{
       uint8_t u8Saturation;
}_stSaturation;
```

Member	Description
u8Saturation	Video saturation value.(0~255)

1.1.3.2.19.PPAPI_PVIRX_VID_GetSaturation

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetSaturation(const PP_U8 IN chanAddr, _stSaturation OUT *pstSaturation);
Description	Get video saturation value.
Argument	chanAddr: channel ID(0~3) pstSaturation:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.20.PPAPI_PVIRX_VID_SetHue

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetHue(const PP_U8 IN chanAddr, const _stHue IN *pstHue);
Description	Set video hue value.
Argument	chanAddr: channel ID(0~3) pstHue:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Types]

_stHue

[Description]

Define the video hue value.

[Syntax]

```
typedef struct
{
        uint8_t u8Hue;
}_stHue;
```

Member	Description
u8Hue	Video hue value.(0~255)

1.1.3.2.21.PPAPI_PVIRX_VID_GetHue

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetHue(const PP_U8 IN chanAddr, _stHue OUT *pstHue);
Description	Get video hue value.
Argument	chanAddr: channel ID(0~3) pstHue:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.22.PPAPI_PVIRX_VID_SetSharpness

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetSharpness(const PP_U8 IN chanAddr, const _stSharpness IN *pstSharpness);
Description	Set video sharpness value.
Argument	chanAddr: channel ID(0~3) pstSharpness:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

[Data Types]

_stSharpness

[Description]

Define the video sharpness value.

[Syntax]

```
typedef struct
{
        uint8_t u8Sharpness;
}_stSharpness;
```

Member	Description

Member	Description
u8Sharpness	Video sharpness value. b[3:0]

1.1.3.2.23.PPAPI_PVIRX_VID_GetSharpness

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_GetSharpness(const PP_U8 IN chanAddr, _stSharpness OUT *pstSharpness);
Description	Get video sharpness value.
Argument	chanAddr: channel ID(0~3) pstSharpness:
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.24.PPAPI_PVIRX_VID_SetBlank

Prototype	PP_RESULT_E PPAPI_PVIRX_VID_SetBlank(const PP_U8 IN chanAddr, const PP_S32 IN bEnable, const PP_S32 IN blankColor);
Description	Enable blank video and blank color when no video status.
Argument	chanAddr: channel ID(0~3) bEnable: TRUE enable blankColor: 0: black, 1: blue
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.25.PPAPI_PVIRX_GetNovidStatus

Prototype	PP_RESULT_E PPAPI_PVIRX_GetNovidStatus(const PP_U8 IN chanAddr, PP_U8 OUT *pStatus);
Description	Get camera plug-in or out status(novideo status).
Argument	chanAddr: channel ID(0~3) pStatus: 1: novideo(plug-out), else video(plut-in).
Return value	eERROR_FAILURE:

	eSUCCESS
Remark	

1.1.3.2.26.PPAPI_PVIRX_UTC_SetTable

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_SetTable(const PP_U8 IN chanAddr, const enum _eCameraStandard IN cameraStandard, const enum _eCameraResolution IN cameraResolution);
Description	Set UTC configuration by camera standard, resolution.
Argument	chanAddr: channel ID(0~3) cameraStandard: camera standard type. cameraResolution: camera resolution type.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.27.PPAPI_PVIRX_UTC_StartRX

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_StartRX(const PP_U8 IN chanAddr, const PP_S32 IN bStart);
Description	Start utc rx process.
Argument	chanAddr: channel ID(0~3) bStart: TRUE:start
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.28.PPAPI_PVIRX_UTC_StartTX

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_StartTX(const PP_U8 IN chanAddr, const PP_S32 IN bStart);
Description	Start utc tx process.
Argument	chanAddr: channel ID(0~3) bStart: TRUE start.

Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.29.PPAPI_PVIRX_UTC_SendData

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_SendData(const PP_U8 IN chanAddr, const enum _eCameraStandard IN cameraStandard, const enum _eCameraResolution IN cameraResolution, const PP_S32 IN dataSize, const PP_U8 IN *pData);
Description	Send utc txdata.
Argument	chanAddr: channel ID(0~3) cameraStandard: camera standard format cameraResolution: camera resolution type. dataSize: Size of UTC byte pData: pointer utc data buffer.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.30.PPAPI_PVIRX_UTC_GetRxAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_GetRxAttr(const PP_U8 IN chanAddr, _stUTCRxAttr OUT *pstUTCRxAttr);
Description	Get utc rx attribute paramters.
Argument	chanAddr: channel ID(0~3) pstUTCRxAttr: UTC Rx registers
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.31.PPAPI_PVIRX_UTC_GetTxAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_GetTxAttr(const PP_U8 IN

	chanAddr, _stUTCTxAttr OUT *pstUTCTxAttr);
Description	Get utc tx attribute paramters.
Argument	chanAddr: channel ID(0~3) pstUTCTxAttr: UTC Tx registers.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.32.PPAPI_PVIRX_UTC_GetHVStartAttr

Prototype	PP_RESULT_E PPAPI_PVIRX_UTC_GetHVStartAttr(const PP_U8 IN chanAddr, _stUTCHVStartAttr OUT *pstUTCHVStartAttr);
Description	Get utc H/V start attribute paramters.
Argument	chanAddr: channel ID(0~3) pstUTCHVStartAttr: UTC HV paramters registers.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.3.2.33.PPAPI_PVIRX_Initialize

Prototype	void PPAPI_PVIRX_Initialize(void)
Description	Initialize PVI Rx h/w module from Board configuration.
Argument	None
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.1.4. How to Use

1.1.4.1. Set Board configuration

Set below of “SDK/config/board_config.h” file.

# Example Video In/Out Feature setting

1) Input: MIPI 720p25 bayer raw12bit camera, Output: 720p25 YUV8bit Embedded Sync

```
#define VIDEO_IN_TYPE          (VIDEO_IN_TYPE_MIPI_BAYER)
#define BD_VIN_FMT             (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
#define BD_CAMERA_IN_FMT       (BD_VIN_FMT | MIPI_VID_BIT_RAW12)
#define BD_SVM_IN_FMT          (BD_VIN_FMT)
#define BD_SVM_OUT_FMT         (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
#define BD_DU_IN_FMT           (BD_SVM_OUT_FMT)
#define BD_DU_OUT_FMT          (BD_DU_IN_FMT | VID_TYPE_YC8_EMB)
#define BD_QUAD_OUT_FMT        (BD_VIN_FMT)
#define BD_RO_OUT_FMT          (BD_VIN_FMT | VID_TYPE_YC8_EMB)
#define BD_VPU_IN_FMT          (BD_QUAD_OUT_FMT)
#define BD_PVITX_OUT_FMT       (BD_DU_IN_FMT | PVITX_SRC_NONE)
```

2) Input: MIPI 960p25 bayer raw12bit camera, Output: 720p25 YUV16bit External Sync

```
#define VIDEO_IN_TYPE          (VIDEO_IN_TYPE_MIPI_BAYER)
#define BD_VIN_FMT             (VID_FRAME_PAL_25 | VID_RESOL_HD960P)
#define BD_CAMERA_IN_FMT       (BD_VIN_FMT | MIPI_VID_BIT_RAW12)
#define BD_SVM_IN_FMT          (BD_VIN_FMT)
#define BD_SVM_OUT_FMT         (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
#define BD_DU_IN_FMT           (BD_SVM_OUT_FMT)
#define BD_DU_OUT_FMT          (BD_DU_IN_FMT | VID_TYPE_YC16_EXT)
#define BD_QUAD_OUT_FMT        (BD_SVM_OUT_FMT)
#define BD_RO_OUT_FMT          (BD_SVM_OUT_FMT | VID_TYPE_YC8_EMB)
#define BD_VPU_IN_FMT          (BD_QUAD_OUT_FMT)
#define BD_PVITX_OUT_FMT       (BD_DU_IN_FMT | PVITX_SRC_NONE)
```

3) Input: HDA 720p25 camera, Output: Digital [720p25 YUV8bit External Sync], Analog HD[HDA 720p25 Du]

```
#define VIDEO_IN_TYPE          (VIDEO_IN_TYPE_PVI)
#define BD_VIN_FMT             (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
#define BD_CAMERA_IN_FMT       (BD_VIN_FMT | VID_STANDARD_HDA)
#define BD_SVM_IN_FMT          (BD_VIN_FMT)
#define BD_SVM_OUT_FMT         (VID_FRAME_PAL_25 | VID_RESOL_HD720P)
#define BD_DU_IN_FMT           (BD_SVM_OUT_FMT)
#define BD_DU_OUT_FMT          (BD_DU_IN_FMT | VID_TYPE_YC16_EXT)
#define BD_QUAD_OUT_FMT        (BD_VIN_FMT)
#define BD_RO_OUT_FMT          (BD_VIN_FMT | VID_TYPE_YC8_EMB)
```

#define BD_VPU_IN_FMT (BD_QUAD_OUT_FMT)

#define BD_PVITX_OUT_FMT (BD_DU_IN_FMT | PVITX_SRC_DU | VID_STANDARD_HDA)

1.1.4.2. Call API Function.

1.1.4.2.1.Call “PPAPI_VIN_Initialize()”.

In function of “PPAPI_VIN_Initialize():”, call below sub functions.

- ➔ vin_initialize()

1.1.4.2.2.Call “PPAPI_PVIRX_Initialize()”.

In function of “PPAPI_PVIRX_Initialize():”, call below sub functions.

- ➔ PPAPI_PVIRX_SetInit()
  - ■ PPAPI_PVIRX_SetTableStdResol()
  - ■ PPAPI_PVIRX_SetAttrChip()
  - ■ PVIRX_SetTableIRQ()
  - ■ PPAPI_PVIRX_UTC_SetTable()

1.1.4.2.3.Call “PPAPI_PVITX_Initialize()”.

In function of “PPAPI_PVITX_Initialize():”, call below sub functions.

- ➔ PPAPI_PVITX_SetInit()
  - ■ PPAPI_PVITX_Set()

1.2. Video Output

PI5008K SDK makes it possible to select various video output format.

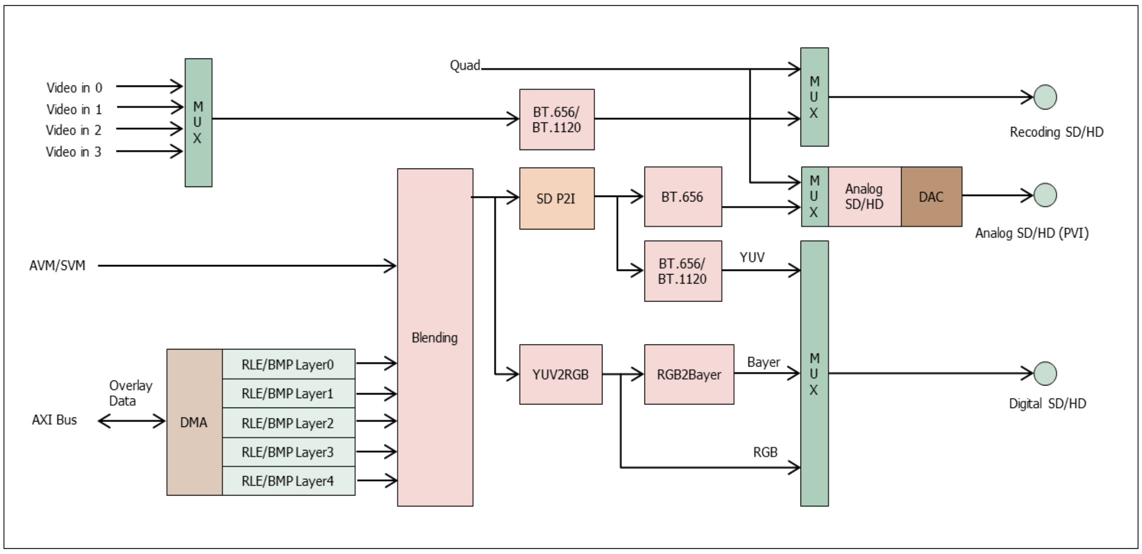


Figure 2 Video Output Flow Chart

1.2.1. Configure Video output:

1.2.1.1. Set DU output format

Select DU output frame rate, resolution and output video signal format.

Ex> DU output format same with SVM h/w module output. And YUV 8bit embedded sync.

```
SDK/config/board_config.h
    #define BD_DU_IN_FMT (BD_SVM_OUT_FMT)
    #define BD_DU_OUT_FMT (BD_DU_IN_FMT | VID_TYPE_YC8_EMB)
```

1.2.1.2. Set RO(Record Output) format

Select video output frame rate, resolution and video signal format.

Ex> RO output format same with VIN format. And YUV 8bit embedded sync.

```
SDK/config/board_config.h
    #define BD_RO_OUT_FMT ((BD_VIN_FMT) | (VID_TYPE_YC8_EMB))
```

1.2.1.3. Set Analog Tx(PVI) format

Select analog output format. Select source DU or Quad. Set standard format.

Ex> If Source is DU. (same with DU output display):

```
SDK/config/board_config.h
    #define BD_PVITX_OUT_FMT (BD_DU_IN_FMT | PVITX_SRC_DU |
```

VID_STANDARD_HDA)

Ex> If source is QUAD :

SDK/config/board_config.h

```
#define BD_PVITX_OUT_FMT (BD_DU_IN_FMT | PVITX_SRC_QUAD |
VID_STANDARD_HDA)
```

1.2.2. APIs

1.2.2.1. Data Types

1.2.2.1.1.enum _pvi_tx_table_type_format

```
enum _pvi_tx_table_type_format {
        pvi_tx_table_format_SD720 = 0,
        pvi_tx_table_format_SD960,
        pvi_tx_table_format_PVI,
        pvi_tx_table_format_HDA,
        pvi_tx_table_format_CVI,
        pvi_tx_table_format_HDT,
        max_pvi_tx_table_type_format
};
```

Member	Description
pvi_tx_table_format_SD720	720x480i60, 720x576i50
pvi_tx_table_format_SD960	960x480i60, 960x576i50
pvi_tx_table_format_PVI	Analog HD PVI(pixelplus) standard.
pvi_tx_table_format_HDA	Analog HD AHD(Nextchip) standard.
pvi_tx_table_format_CVI	Analog HD CVI(Dahwa) standard.
pvi_tx_table_format_HDT	Analog HD TVI(HikVision) standard.

1.2.2.1.1.enum _pvi_tx_table_resol_format

```
enum _pvi_tx_table_resol_format {
        pvi_tx_table_format_720x480i60 = 0,
        pvi_tx_table_format_720x576i50,
        pvi_tx_table_format_960x480i60,
        pvi_tx_table_format_960x576i50,
        pvi_tx_table_format_1280x720p60,
        pvi_tx_table_format_1280x720p50,
        pvi_tx_table_format_1280x720p30,
        pvi_tx_table_format_1280x720p25,
        pvi_tx_table_format_1280x960p30,
        pvi_tx_table_format_1280x960p25,
        pvi_tx_table_format_1920x1080p30,
        pvi_tx_table_format_1920x1080p25,
        max_pvi_tx_table_resol_format
}
```

Member	Description
pvi_tx_table_format_720x480i60	720x480i60.
pvi_tx_table_format_720x576i50	720x576i50.
pvi_tx_table_format_960x480i60	960x480i60.
pvi_tx_table_format_960x576i50	960x576i50.
pvi_tx_table_format_1280x720p60	1280x720p60
pvi_tx_table_format_1280x720p50	1280x720p50
pvi_tx_table_format_1280x720p30	1280x720p30
pvi_tx_table_format_1280x720p25	1280x720p25
pvi_tx_table_format_1280x960p30	1280x960p30
pvi_tx_table_format_1280x960p25	1280x960p25
pvi_tx_table_format_1920x1080p30	1920x1080p30.
pvi_tx_table_format_1920x1080p25	1920x1080p25.

1.2.2.2. Analog Tx(PVI)

1.2.2.2.1.PPAPI_PVITX_CheckChipID

Prototype	PP_RESULT_E PPAPI_PVITX_CheckChipID(PP_U16 OUT *pRetChipID, PP_S32 OUT *pRetRWVerify);
Description	Get current Chip pvi rx ID and register access verify status.
Argument	pRetChipID: Return chip ID (0x1000) pRetRWVerify: 1: success, else:failure
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.2.2.2.2.PPAPI_PVITX_Set

Prototype	void PPAPI_PVITX_Set(const enum _pvi_tx_table_type_format IN typeFormat, const enum _pvi_tx_table_resol_format IN resolFormat);
Description	Set pvi tx h/w module as standard, resolution.
Argument	typeFormat: PVI Tx standard type. resolFormat: PVI Tx resolution type.
Return value	None
Remark	

[Data Types]

_pvi_tx_table_type_format

[Description]

Define the pvi tx standard format.

[Syntax]

```
enum _pvi_tx_table_type_format {
      pvi_tx_table_format_SD720 = 0,
      pvi_tx_table_format_SD960,
      pvi_tx_table_format_PVI,
      pvi_tx_table_format_HDA,
      pvi_tx_table_format_CVI,
      pvi_tx_table_format_HDT,
      max_pvi_tx_table_type_format
};
```

Member	Description
pvi_tx_table_format_SD720	Cvbs (NTSC, PAL) 720x480i60, 720x576i50
pvi_tx_table_format_SD960	Cvbs 960x480i60, 960x576i50
pvi_tx_table_format_PVI	Analog HD PVI(pixelplus) standard.
pvi_tx_table_format_HDA	Analog HD AHD(Nextchip) standard.
pvi_tx_table_format_CVI	Analog HD CVI(Dahwa) standard.
pvi_tx_table_format_HDT	Analog HD TVI(HikVision) Old standard.

[Data Types]

_pvi_tx_table_resol_format

[Description]

Define the camera resolution.

[Syntax]

```
enum _pvi_tx_table_resol_format {
      pvi_tx_table_format_720x480i60 = 0,
      pvi_tx_table_format_720x576i50,
      pvi_tx_table_format_960x480i60,
      pvi_tx_table_format_960x576i50,
      pvi_tx_table_format_1280x720p60,
      pvi_tx_table_format_1280x720p50,
      pvi_tx_table_format_1280x720p30,
      pvi_tx_table_format_1280x720p25,
      pvi_tx_table_format_1280x960p30,
      pvi_tx_table_format_1280x960p25,
      pvi_tx_table_format_1920x1080p30,
      pvi_tx_table_format_1920x1080p25,
      max_pvi_tx_table_resol_format
};
```

Member	Description
pvi_tx_table_format_720x480i60	720x480i60.
pvi_tx_table_format_720x576i50	720x576i50.
pvi_tx_table_format_960x480i60	960x480i60.
pvi_tx_table_format_960x576i50	960x576i50.
pvi_tx_table_format_1280x720p60	1280x720p60
pvi_tx_table_format_1280x720p50	1280x720p50
pvi_tx_table_format_1280x720p30	1280x720p30
pvi_tx_table_format_1280x720p25	1280x720p25
pvi_tx_table_format_1280x960p30	1280x960p30
pvi_tx_table_format_1280x960p25	1280x960p25
pvi_tx_table_format_1920x1080p30	1920x1080p30.
pvi_tx_table_format_1920x1080p25	1920x1080p25.

1.2.2.2.3.PPAPI_PVITX_SetInit

Prototype	PP_RESULT_E PPAPI_PVITX_SetInit(const enum _pvi_tx_table_type_format IN pviTxType, const enum _pvi_tx_table_resol_format IN pviTxResol);

Description	Set parameters. VIN h/w module
Argument	pviTxType: PVI Tx standard type. pviTxResol: PVI Tx resolution type.
Return value	eERROR_FAILURE: eSUCCESS
Remark	

1.2.2.2.4.PPAPI_PVITX_Initialize

Prototype	void PPAPI_PVITX_Initialize(void)
Description	Initialize Analog Tx(PVI Tx)
Argument	None
Return value	None
Remark	Not used

Version	Date	Description
V0.0	20180510	
V0.1	20180608	
V0.2	20180725	
V0.3	20181115	