Crystal Image through Imaging Innovation

PIXELPLUS

SURROUND VIEW MONITORING SYSTEM

PI5008K SVM User Guide

Rev 0.3

Last Update : 2018.11.01

6th Floor, 105, Gwanggyo-ro, Yeongtong-gu,
Suwong-si, Gyeonggi-do, 16229, Korea
Tel : +82-31-888-5300, FAX : +82-31-888-5399

Contents

1. Overview

1.1. Introduction

This documentation explains SVM((Surround View Monitoring) API.

1.2. Definitions

1.2.1. FB LUT (Front & Back LookUp Table)

Lookup table for input data of front and back channel.

1.2.2. LR LUT (Left & Right LookUp Table)

Lookup table for input data of Left and Right channel

1.2.3. BC LUT (Brightness Control LookUp Table)

Lookup table for blending of FB LUT and LR LUT and brightness control

1.2.4. BC ADD LUT (Brightness Control ADDitional LookUp Table) [Option]

Additional Lookup table to compensate curve of boundaries of BC LUT.

1.2.5. Shadow image [Option]

Rectangle image to hide invalid area where car image is drawn

2. Block Diagram

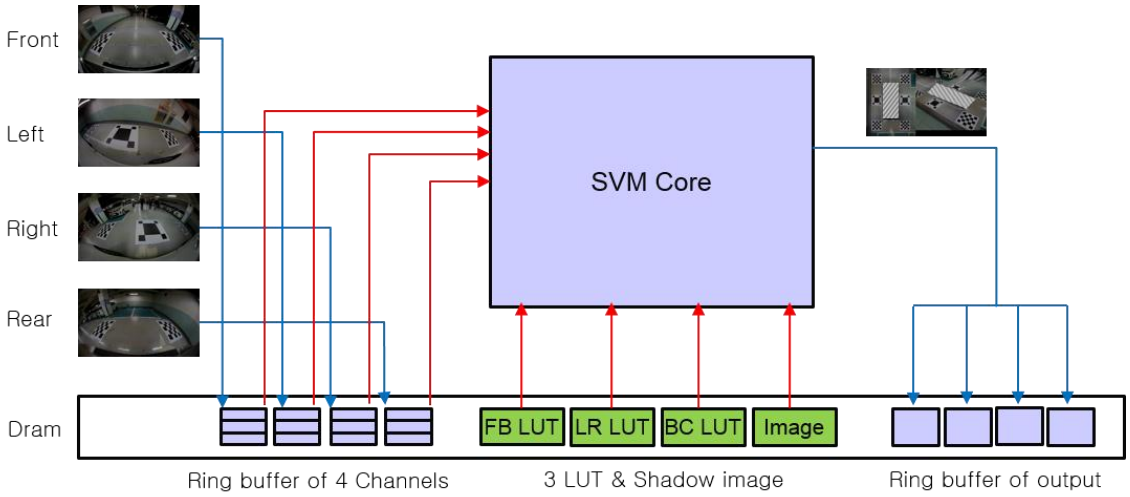

Fig 2-1. Block Diagram

Input data for each channel will be stored at the corresponding ring buffer, and SVM Core generates output data using FB, LR, BC LUT and YUV422(YUYV) Image* stored in the memory. Output data will be stored at output frame buffer. The number of output frame buffer is 2 ~4.

Notes> This image is used to display shadow image.

3. Surround view generation process

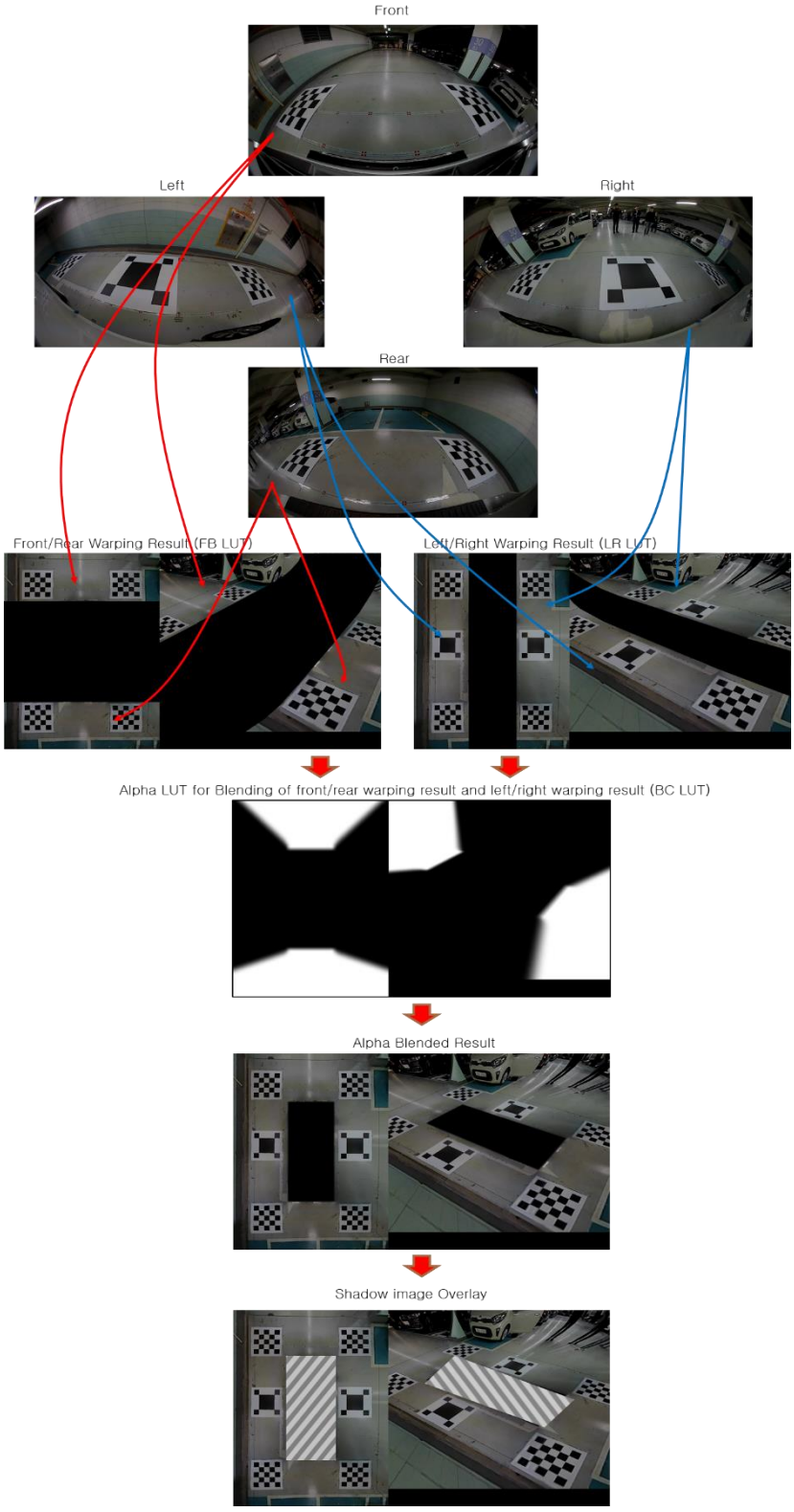

Fig 3-1. LUT & Shadow image

1. Front and Rear input data are merged by using FB LUT.
   (If morphing is used, FB LUT may not be used every frame. (see. 4.3.2))
2. Left and Right input data are merged using LR LUT.
   (If morphing is used, LR LUT may not be used every frame. (see. 4.3.2))
3. Front+Rear and Left+Right data are merged using a blending of BC LUT.
4. If there is YUV422(YUYV) image, it will be overlayed to make final output data.

4. View mode switching process

4.1. Section of view mode

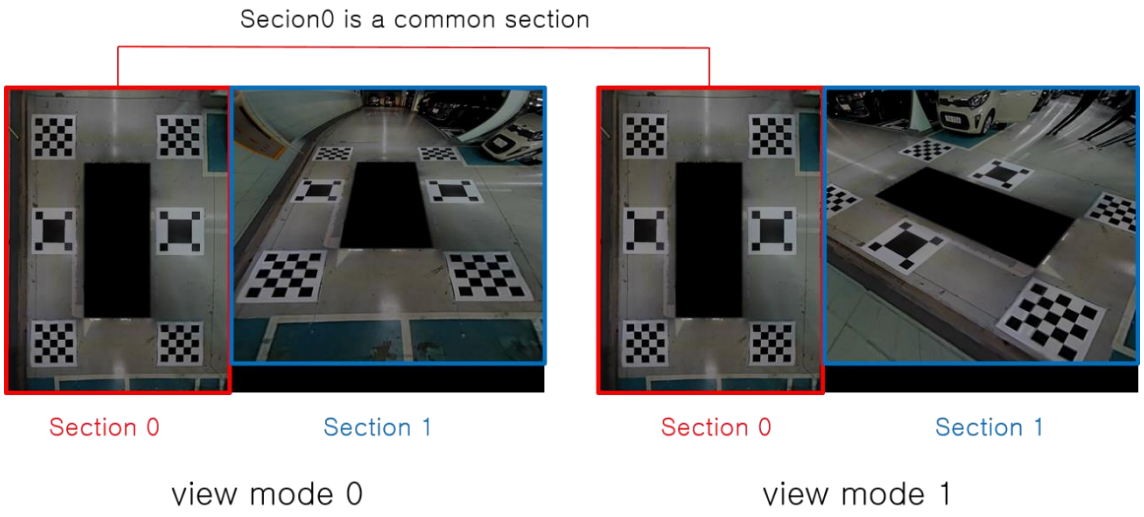

Fig 4-1. Section of view mode

Section is sub-view in view mode.

Common section is common for all view modes.

4.2. Switch to a specific view mode

4.2.1. Loading data from flash memory

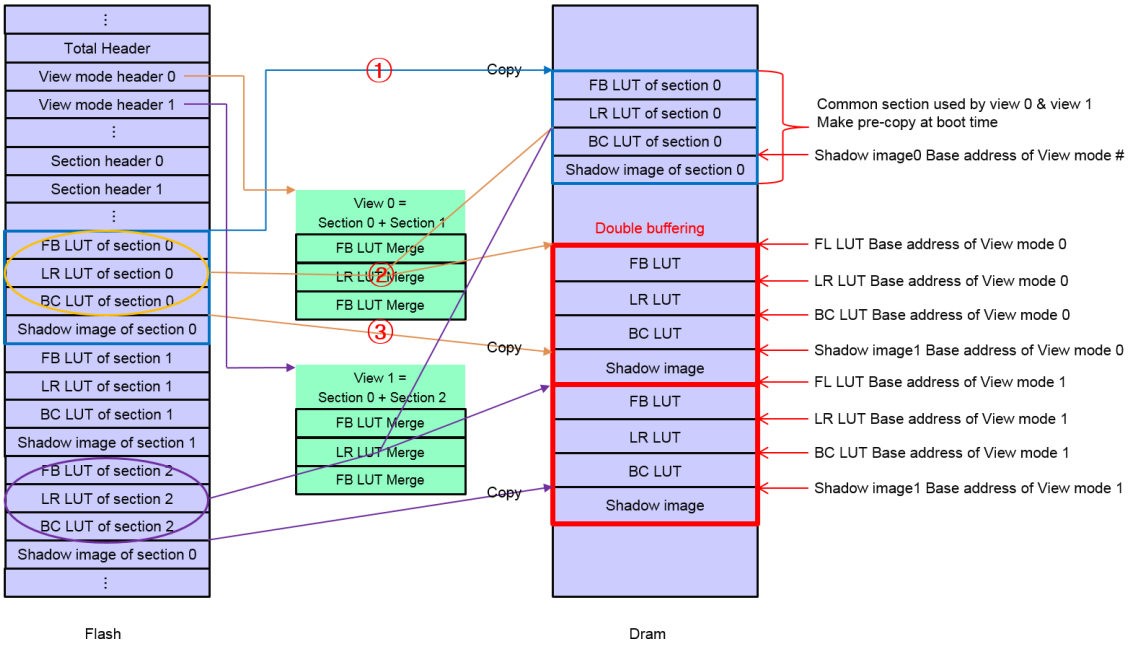

Fig 4-2. Flash memory map and loading data for single view mode change

① Common section is copied from flash to dram at boot time.

② Section is copied from flash to dram and merge with common section.

③ Shadow image of section is copied from flash to dram.

Double buffering is used to switch to a specific view mode.

Before switching view mode, data in Flash memory is copied to DRAM and merge the sections. SVM block will access this data using the base address of data.

.

4.2.2. Vsync timming

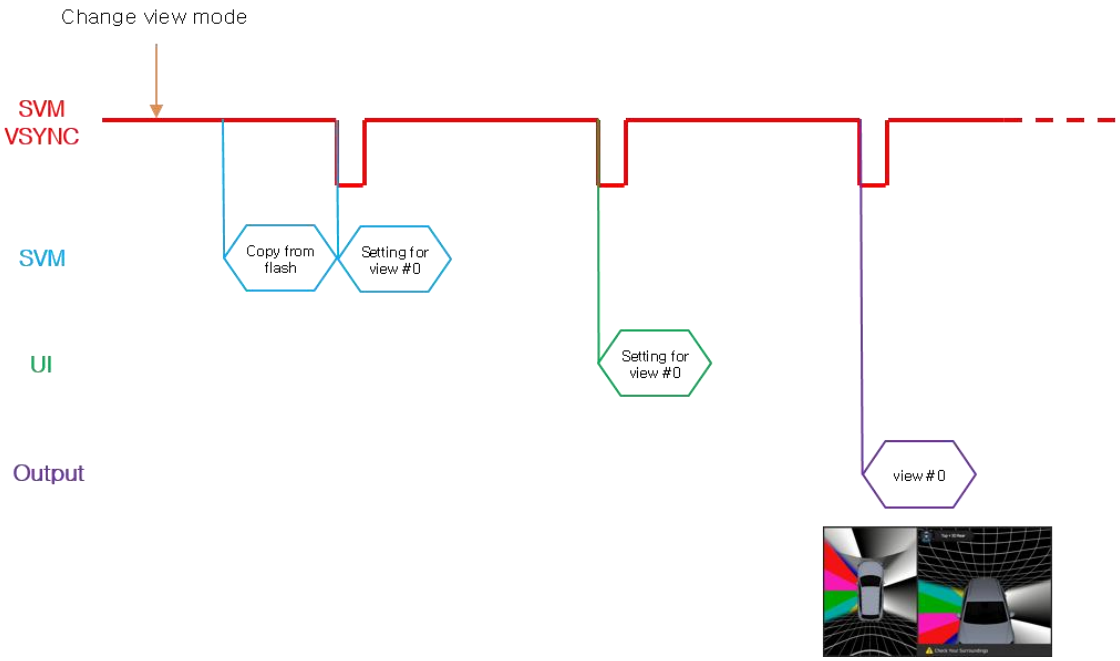

Fig 4-3. Vsync timing for switching to a specific view mode

Switching to a specific view mode will be done as follows.

1) Data stored in flash memory will be copied to DRAM.
2) SVM setting will be done at 1st Vsync after coying data
3) UI will be set at 2nd Vsync. UI resource has to be loaded into DRAM in advance will be loading Setting .
4) SVM data and UI will output at 3rd Vsync.

4.2.3. View mode switching process diagram

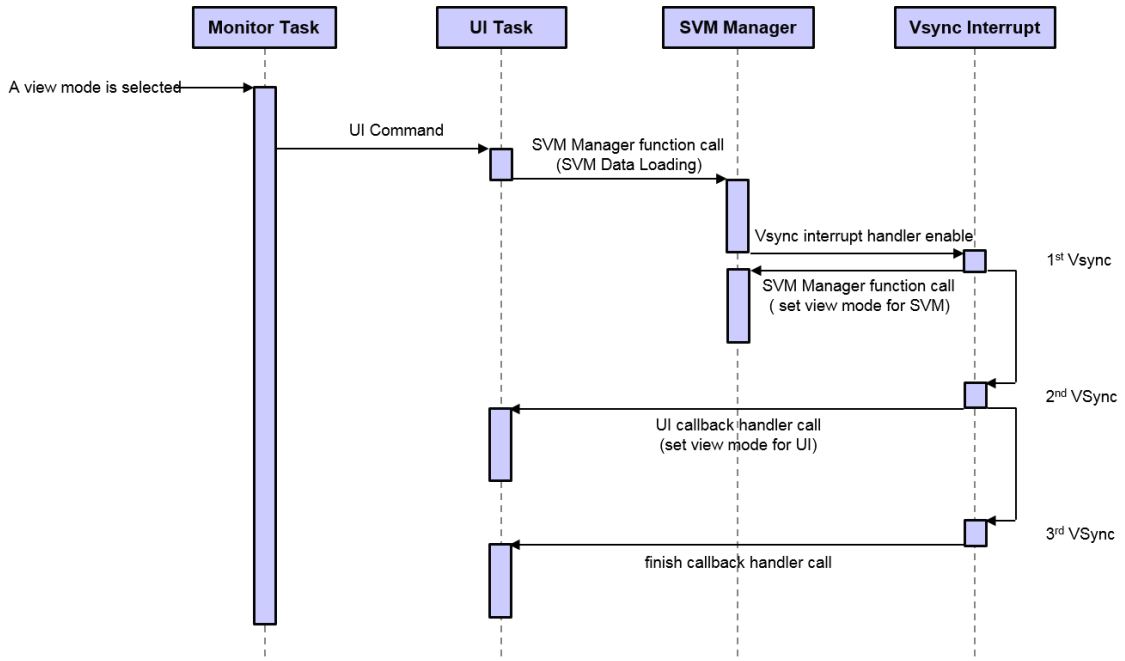

Fig 4-4. View mode switching processing diagram

4.3. Swing view mode switching

4.3.1. Flash memory map and Loading

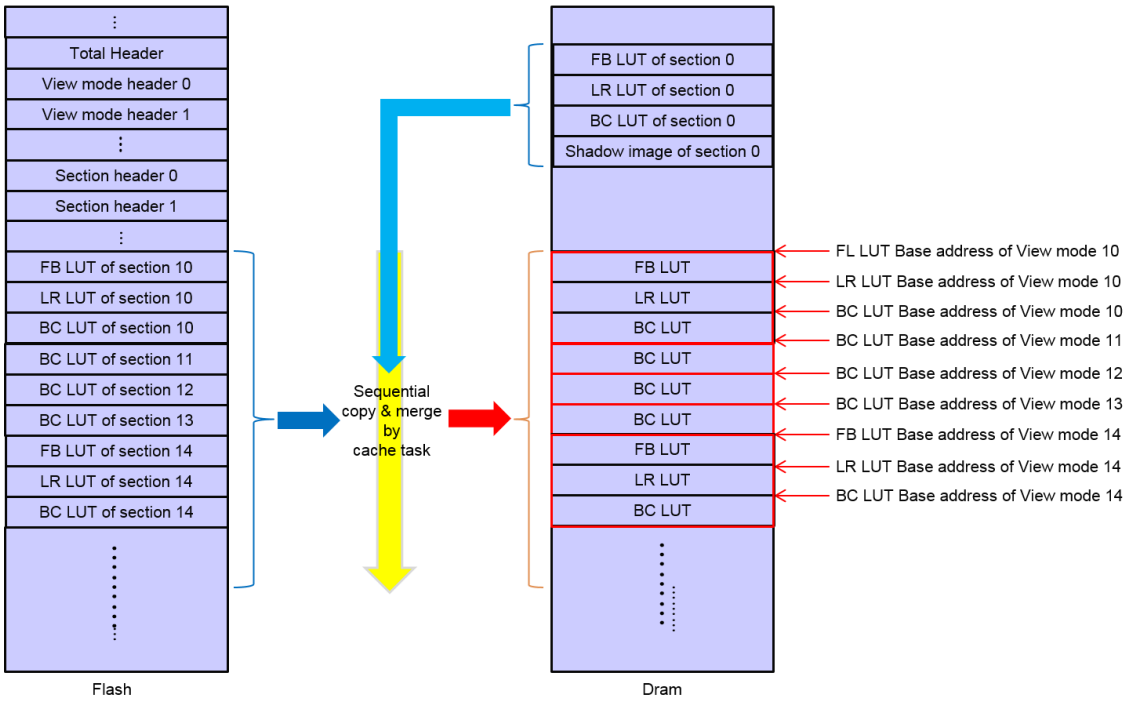

Fig 4-5. Flash memory structure and loading for swing view mode switching

To switch swing view mode, LUT data for a view has to be loaded from flash memory to DRAM before switching to the view. SVM block uses base address of loaded data to generate view mode.

Morphing function can be used for continuous the switching swing view mode.

4.3.2. Morphing

Generally one view consists of FB & LR & BC LUT.

If morphing is used, the views can be made using BC LUT between two views.

To switch swing view mode continuously, this function can be useful.

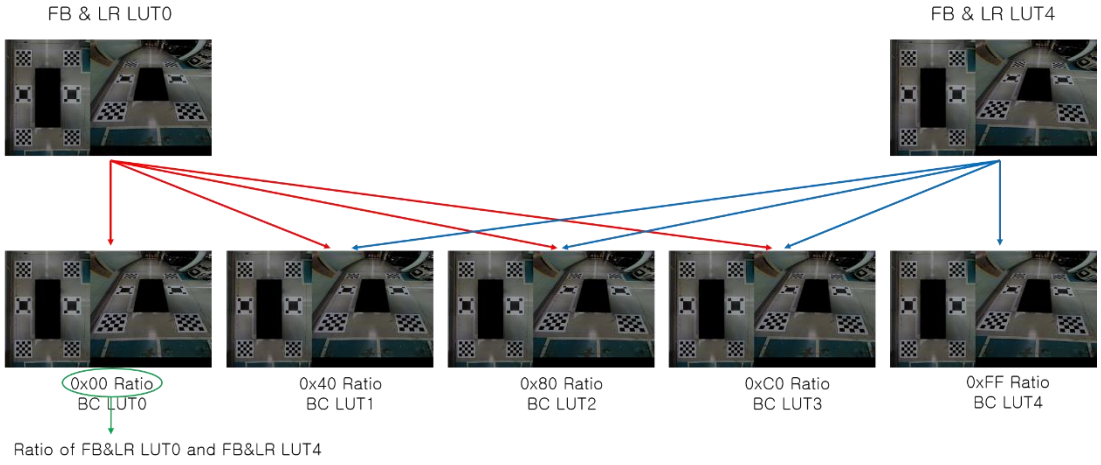

Fig 4-6. Morphing flow

4.3.3. Vsync timming

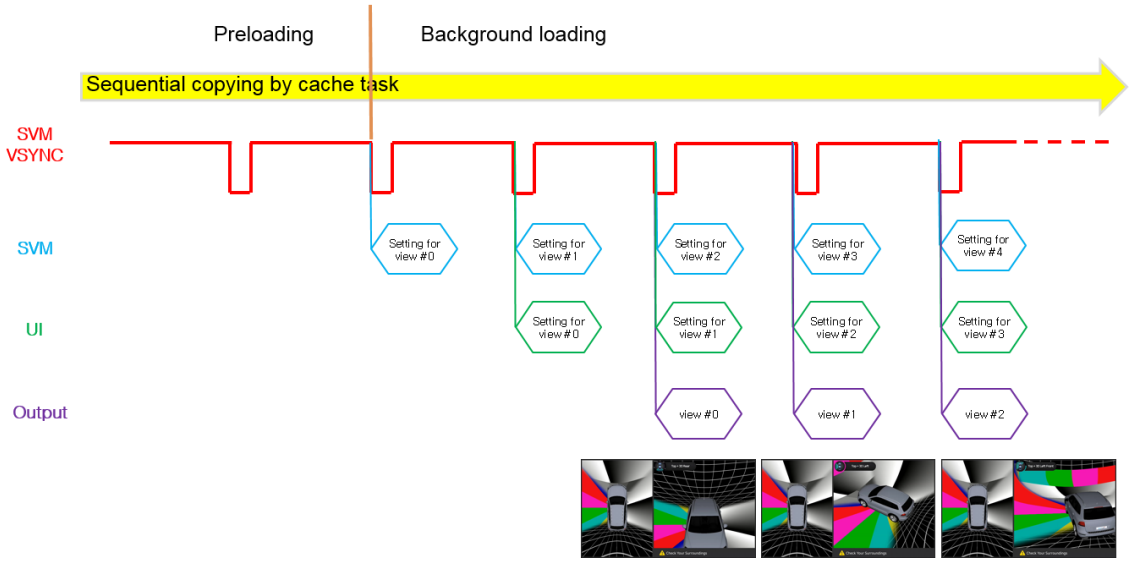

Fig 4-7. Vsync timming for swing view mode swiching

To switch swing view mode continuously, a certain amount of LUT data has to be loaded in advance. The remaining LUT data will be loaded in the background during switching view mode by cash task. Setting SVM and UI for next view mode will be done every Vsync. But UI setting must be done 1 Vsync after SVM setting.

A view will be out 2 Vsync after setting SVM for the view

5. viewmode_config.h

```
#define VIEWMODE_NULL    (0xFFFFFFFF)


typedef enum ppVIEWMODE_E
{
    eVIEWMODE_LOAD_2D_FRONT = 0,
    eVIEWMODE_LOAD_2D_REAR,
    eVIEWMODE_LOAD_3D_FRONT,
    eVIEWMODE_LOAD_3D_LEFT,
    eVIEWMODE_LOAD_3D_LEFTFRONT,
    eVIEWMODE_LOAD_3D_REAR,
    eVIEWMODE_LOAD_3D_RIGHTFRONT,
    eVIEWMODE_LOAD_3D_RIGHT,
    eVIEWMODE_LOAD_TOP2D_SWING_START,
    eVIEWMODE_LOAD_TOP2D_SWING_0=eVIEWMODE_LOAD_TOP2D_SWING_START + 0,
    eVIEWMODE_LOAD_TOP2D_SWING_44=eVIEWMODE_LOAD_TOP2D_SWING_START + 22,
    eVIEWMODE_LOAD_TOP2D_SWING_90=eVIEWMODE_LOAD_TOP2D_SWING_START + 45,
    eVIEWMODE_LOAD_TOP2D_SWING_134=eVIEWMODE_LOAD_TOP2D_SWING_START + 67,
    eVIEWMODE_LOAD_TOP2D_SWING_180=eVIEWMODE_LOAD_TOP2D_SWING_START + 90,
    eVIEWMODE_LOAD_TOP2D_SWING_224=eVIEWMODE_LOAD_TOP2D_SWING_START + 112,
    eVIEWMODE_LOAD_TOP2D_SWING_270=eVIEWMODE_LOAD_TOP2D_SWING_START + 135,
    eVIEWMODE_LOAD_TOP2D_SWING_314=eVIEWMODE_LOAD_TOP2D_SWING_START + 157,
    eVIEWMODE_LOAD_TOP2D_SWING_MAX=eVIEWMODE_LOAD_TOP2D_SWING_START + 180,

    eVIEWMODE_LOAD_MAX,

    eVIEWMODE_BASIC_FRONT_BYPASS = eVIEWMODE_LOAD_MAX,
    eVIEWMODE_BASIC_LEFT_BYPASS,
    eVIEWMODE_BASIC_RIGHT_BYPASS,
    eVIEWMODE_BASIC_REAR_BYPASS,
    eVIEWMODE_BASIC_QUAD,

    eVIEWMODE_TOTAL_MAX,
} PP_VIEWMODE_E;
```

This is enumeration for available view list. It is used for view control.

If new view mode is generated by PC tool and LUT for new view mode is stored at flash memory, view list before eVIEWMODE_LOAD_MAX also has to be changed by a user.

View list after VIEWMODE_LOAD_MAX should not be changed.

~ eVIEWMODE_LOAD_MAX	View List in the same sequence with view stored in flasg memory
eVIEWMODE_BASIC_FRONT_BYPASS	View mode to output front camera
eVIEWMODE_BASIC_LEFT_BYPASS	View mode to output left camera
eVIEWMODE_BASIC_RIGHT_BYPASS	View mode to output right camera
eVIEWMODE_BASIC_REAR_BYPASS	View mode to output rear camera
eVIEWMODE_BASIC_QUAD_BYPASS	View mode to output Front, Left, Right, Rear camera simulateneously. Each camera input is shown on the quattered screen.

Ex) If view modes are stored as below, viewmode_config.h is also changed to apply this change.

1. eVIEWMODE_LOAD_3D_FRONT // TOP + 3D FRONT
2. eVIEWMODE_LOAD_3D_LEFT // TOP + 3D LEFT
3. eVIEWMODE_LOAD_3D_RIGHT // TOP + 3D RIGHT
4. eVIEWMODE_LOAD_3D_REAR // TOP + 3D REAR

```
#define VIEWMODE_NULL            (0xFFFFFFFF)

typedef enum ppVIEWMODE_E
{
    eVIEWMODE_LOAD_3D_FRONT = 0,
    eVIEWMODE_LOAD_3D_LEFT,
    eVIEWMODE_LOAD_3D_RIGHT,
    eVIEWMODE_LOAD_3D_REAR,

    eVIEWMODE_LOAD_MAX,

    eVIEWMODE_BASIC_FRONT_BYPASS = eVIEWMODE_LOAD_MAX,
    eVIEWMODE_BASIC_LEFT_BYPASS,
    eVIEWMODE_BASIC_RIGHT_BYPASS,
    eVIEWMODE_BASIC_REAR_BYPASS,
    eVIEWMODE_BASIC_QUAD,
```

```
    eVIEWMODE_TOTAL_MAX,
} PP_VIEWMODE_E;
```

6. SVM API

6.1. PPAPI_SVM_Initialize

Prototype	PP_RESULT_E PPAPI_SVM_Initialize(PP_VOID);
Description	Initialize SVM block (API & Driver)
Argument	
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : Fail to initialize
Example	

6.2. PPAPI_SVMMEM_Initialize

Prototype	PP_RESULT_E PPAPI_SVMMEM_Initialize(PP_VIEWMODE_E IN enCacheStartView, PP_VIEWMODE_E IN enCacheEndView);
Description	Set memory for SVM block. This API has to be called between PPAPI_FLASH_Initialize() and init_dram_reserved(). This API has to be called before PPAPI_SVM_Initialize().
Argument	enCacheStartView : 1st view mode which is read by Cache task (see. 5) VIEWMODE_NULL : Do not use Cache task enCacheEndView : Last view mode which is read by Cache task (see. 5) VIEWMODE_NULL : Do not use Cache task.
Return value	eSUCCESS eERROR_NO_FLASH_MEM : There is no SVM data in Flash. eERROR_NO_MEM : Heap memory error

Example	……
	// initialization of flash
	PPAPI_FLASH_Initialize(SPI_NAND_FLASH_PAGE_SIZE,
	SPI_NAND_FLASH_ERASE_BLOCK_SIZE, eFLASH_TYPE_NAND)
	
	// read flash header
	PPAPI_FLASH_ReadHeader()
	PPAPI_FLASH_ReadFTLHeader()

	……
	PPAPI_SVMMEM_Initialize(eVIEWMODE_LOAD_TOP2D_SWING_START,
	eVIEWMODE_LOAD_TOP2D_SWING_MAX);
	……
	// initialization of reserved memory
	init_dram_reserved();
	……
	PPAPI_SVM_Initialize();

6.3. PPAPI_SVM_SetInFrameBufferAddress

Prototype	PP_RESULT_E PPAPI_SVM_SetInFrameBufferAddress(PP_CHANNEL_E IN enChannel, PP_U32* IN pu32Addr);
Description	Set base address of input buffer
Argument	enChannel : Input channel typedef enum ppCHANNEL_E eCHANNEL_FRONT eCHANNEL_LEFT, eCHANNEL_RIGHT, eCHANNEL_REAR } PP_CHANNEL_E; pu32Addr : Memory base address (16byte align)
Return value	eSUCCESS eERROR_INVALID_ARGUMENT : Argument value error eERROR_SVM_NOT_INITIALIZE : SVM is not initialized. eERROR_INVALID_ALIGN : address의 16byte align error
Example	

6.4. PPAPI_SVM_GetInFrameBufferAddress

Prototype	PP_U32* PPAPI_SVM_GetInFrameBufferAddress(PP_CHANNEL_E IN enChannel)
Description	Get the base address of input buffer
Argument	enChannel : Input channel (see. 6.3)
Return value	Base address of input buffer

	PP_NULL
Example	

6.5. PPAPI_SVM_SetOutFrameBufferAddress

Prototype	PP_RESULT_E PPAPI_SVM_SetOutFrameBufferAddress(PP_SVMMEM_OUT_FRAMEBUF_NUM_E IN enOutFBNum, PP_U32* IN pu32Addr);
Description	Set the base address of output buffer
Argument	enOutFBNum : Output buffer Number Maximum number is defined as SVMMEM_OUT_BUFFER_COUNT. typedef enum ppSVMMEM_OUT_FRAMBUF_NUM_E { eSVMMEM_OUT_FB_NUM_0 = 0, eSVMMEM_OUT_FB_NUM_1, eSVMMEM_OUT_FB_NUM_2, eSVMMEM_OUT_FB_NUM_3, eSVMMEM_OUT_FB_NUM_MAX = SVMMEM_OUT_BUFFER_COUNT, } PP_SVMMEM_OUT_FRAMEBUF_NUM_E; pu32Addr : Memory base address (16byte align)
Return value	eSUCCESS eERROR_INVALID_ARGUMENT : Argument value error eERROR_SVM_NOT_INITIALIZE : SVM is not initialized eERROR_INVALID_ALIGN : address의 16byte align error
Example	

6.6. PPAPI_SVM_GetOutFrameBufferAddress

Prototype	PP_U32* PPAPI_SVM_GetOutFrameBufferAddress(PP_SVMMEM_OUT_FRAMEBUF_NUM_E IN enOutFBNum);
Description	Get base address of output buffer
Argument	enOutFBNum : Output buffer number (see. 6.5)
Return value	Output buffer base address or PP_NULL
Example	

6.7. PPAPI_SVM_SetMirroring

Prototype	PP_VOID PPAPI_SVM_SetMirroring(PP_CHANNEL_E IN enChannel, PP_BOOL IN bHorizontal, PP_BOOL IN bVertical);
Description	Set top/bottom and left/right mirroring for each input channel
Argument	enChannel : Input channel (see. 6.3) bHorizontal : set left/right mirroring bVertical : set top/bottom mirroring
Return value	
Example	

6.8. PPAPI_SVM_SetAntiAliasing

Prototype	PP_RESULT_E PPAPI_SVM_SetAntiAliasing(PP_CHANNEL_E IN enChannel, PP_SVMDRV_ANTI_ALIASING_STRENGTH_H_E IN enHorizotal, PP_SVMDRV_ANTI_ALIASING_STRENGTH_V_E IN enVertical);
Description	Set anti-aliasing of each input channel
Argument	enChannel : input channel (see. 6.3) enHorizotal : horizontal anti-aliasing value typedef enum ppSVMDRV_ANTI_ALIASING_STRENGTH_H_E { eSVMDRV_AA_H_1 = 0, eSVMDRV_AA_H_2, eSVMDRV_AA_H_3, eSVMDRV_AA_H_4, eSVMDRV_AA_H_5, eSVMDRV_AA_H_6, eSVMDRV_AA_H_7, eSVMDRV_AA_H_MAX, } PP_SVMDRV_ANTI_ALIASING_STRENGTH_H_E; enVertical: vertical anti-aliasing value typedef enum ppSVMAPI_ANTI_ALIASING_STRENGTH_V_E {

	eSVMDRV_AA_V_1 = 0, eSVMDRV_AA_V_2, eSVMDRV_AA_V_3, eSVMDRV_AA_V_4, eSVMDRV_AA_V_MAX, } PP_SVMDRV_ANTI_ALIASING_STRENGTH_V_E;
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM is not initialized eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.9. PPAPI_SVM_SetReplaceColor

Prototype	PP_VOID PPAPI_SVM_SetReplaceColor(PP_U8 IN u8Y, PP_U8 IN u8Cb, PP_U8 IN u8Cr);
Description	Set replace color used instead of input output data
Argument	u8Y u8Cb u8Cr
Return value	
Example	

6.10. PPAPI_SVM_SetInputReplaceColorOnOff

Prototype	PP_VOID PPAPI_SVM_SetInputReplaceColorOnOff(PP_CHANNEL_E IN enChannel, PP_BOOL IN bOn);
Description	Decide whether to use replace color instead of input data
Argument	enChannel : input channel (see. 6.3) bOn : PP_TRUE or PP_FALSE
Return value	
Example	

6.11. PPAPI_SVM_GetInputReplaceColorOnOff

Prototype	PP_BOOL PPAPI_SVM_GetInputReplaceColorOnOff(PP_CHANNEL_E IN enChannel);
Description	Get the setting whether to use replace color which is used instead of input data
Argument	enChannel : Input channel (see. 6.3)
Return value	PP_TRUE PP_FALSE
Example	

6.12. PPAPI_SVM_SetOutputReplaceColorOnOff

Prototype	PP_VOID PPAPI_SVM_SetOutputReplaceColorOnOff(PP_BOOL IN bOn);
Description	Decide whether to use replace color used instead of output data Replace color : (see. 6.9)
Argument	bOn : PP_TRUE or PP_FALSE
Return value	
Example	

6.13. PPAPI_SVM_GetOutputReplaceColorOnOff

Prototype	PP_BOOL PPAPI_SVM_GetOutputReplaceColorOnOff(PP_VOID);
Description	Get the setting whether to use replace color which is used instead of output data
Argument	
Return value	PP_TRUE PP_FALSE
Example	

6.14. PPAPI_SVM_SetBackgroundColor

Prototype	PP_VOID PPAPI_SVM_SetBackgroundColor(PP_U8 IN u8Y,

	PP_U8 IN u8Cb, PP_U8 IN u8Cr);
Description	Set Background color
Argument	u8Y u8Cb u8Cr
Return value	
Example	

6.15. PPAPI_SVM_SetImageMaskColor

Prototype	PP_VOID PPAPI_SVM_SetImageMaskColor(PP_U8 IN u8Y, PP_U8 IN u8Cb, PP_U8 IN u8Cr);
Description	Set mask color of YUV422(YUYV) image Mask color is not displayed.
Argument	u8Y u8Cb u8Cr
Return value	
Example	

6.16. PPAPI_SVM_SetEdgeEnhancement

Prototype	PP_RESULT_E PPAPI_SVM_SetEdgeEnhancement(PP_S16 IN s16Edge);
Description	Set value of edge enhancement
Argument	s16Edge -1 : Edge enhancement is not applied 0 ~ 255 : Apply edge enhancement according to value
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM is not initialized
Example	

6.17. PPAPI_SVM_SetDynamicBlending

Prototype	PP_VOID PPAPI_SVM_SetDynamicBlending(PP_SVMAPI_DYNAMIC_BLENDING_E enPos, PP_U8 u8Ratio);
Description	Set ratio of Dynamic blending Dynamic blending function can change overlapped area of input image using ratio value. (see. Fig 6-1) This function is used in the MOD(Moving Object Detection).
Argument	enPos : Area typedef enum ppSVMAPI_DYNAMIC_BLENDING_E { eSVMAPI_DB_FRONTLEFT = 0, eSVMAPI_DB_FRONTRIGHT, eSVMAPI_DB_REARLEFT, eSVMAPI_DB_REARRIGHT, eSVMAPI_DB_MAX, } PP_SVMAPI_DYNAMIC_BLENDING_E; u8Ratio : Dynamic blending ratio value
Return value	
Example	

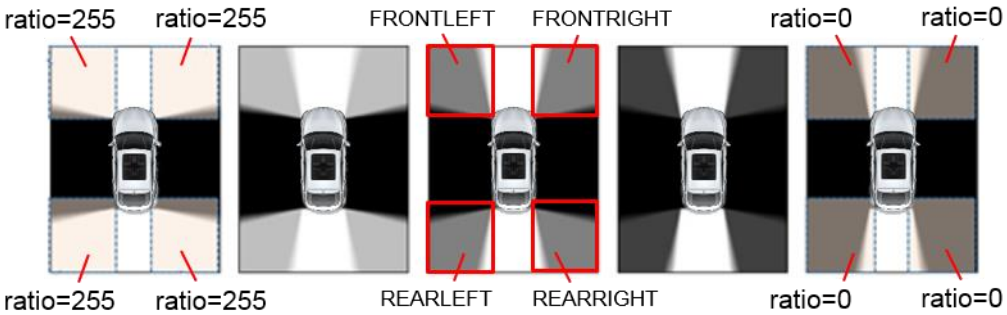

Fig 6-1. Dynamic blending

6.18. PPAPI_SVM_SetDynamicBlendingOnOff

Prototype	PP_VOID PPAPI_SVM_SetDynamicBlendingOnOff(PP_BOOL bOn);

Description	Enable/disable Dynamic blending This function is used in the MOD(Moving Object Detection).
Argument	bOn : PP_TRUE or PP_FALSE
Return value	
Example	

6.19. PPAPI_SVM_SetWindowOffset

Prototype	PP_VOID PPAPI_SVM_SetWindowOffset(PP_S8 IN s8X, PP_S8 IN s8Y);
Description	Adjust start position of output image (see. Fig 6-2)
Argument	s8X : -63 ~ 63 s8Y : -63 ~ 63
Return value	eSUCCESS eERROR_INVALID_ARGUMENT : Argument value error eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized
Example	

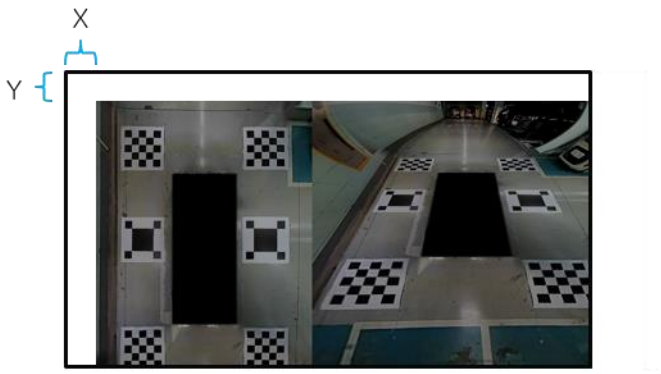

Fig 6-2. Window offset

6.20. PPAPI_SVM_SetOutputHold

Prototype	PP_VOID PPAPI_SVM_SetOutputHold(PP_BOOL IN bEnable);
Description	Pause output image
Argument	bEnable : PP_TRUE or PP_FALSE

Return value	
Example	

6.21. PPAPI_SVM_GetHoldFrameBufferAddress

Prototype	PP_U32* PPAPI_SVM_GetHoldFrameBufferAddress(PP_FIELD_E IN enField);
Description	Get memory base address of paused output image

Argument	enField : Input field
	typedef enum ppFIELD_E
	{
	eFIELD_NONE = 0, // progressive
	eFIELD_ODD = 0, // odd field of interlace
	eFIELD_EVEN, // even field of interlace
	eFIELD_MAX,
	} PP_FIELD_E;

Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not intialized
Example	

6.22. PPDRV_SVM_CTRL_GetVersion

Prototype	PP_U32 PPAPI_SVM_GetVersion(PP_VOID);
Description	SVM Block version
Argument	
Return value	SVM Block version
Example	

6.23. PPAPI_SVM_SetFBLUTAddress

Prototype	PP_RESULT_E PPAPI_SVM_SetFBLUTAddress(PP_VIEWMODE_E IN enView, PP_U32* IN pu32Addr, PP_FIELD_E IN enField);
Description	Set memory base address of FB LUT

Argument	enView : View mode (see. 5) pu32Addr : Memory base address (16byte align) enField : Field number (see. 6.21)
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initiailzed eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: Not generated view mode eERROR_INVALID_ALIGN : 16byte align error eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.24. PPAPI_SVM_GetFBLUTAddress

Prototype	PP_U32* PPAPI_SVM_GetFBLUTAddress(PP_VIEWMODE_E IN enView, PP_FIELD_E IN enField);
Description	Get memory base address of FB LUT
Argument	enView : View mode (see. 5) enField : Field number (see. 6.21)
Return value	Memory base address of FB LUT PP_NULL
Example	

6.25. PPAPI_SVM_SetLRLUTAddress

Prototype	PP_RESULT_E PPAPI_SVM_SetLRLUTAddress(PP_VIEWMODE_E IN enView, PP_U32* IN pu32Addr, PP_FIELD_E IN enField);
Description	Set memory base address of LR LUT
Argument	enView : View mode (see. 5) pu32Addr : Memory base address (16byte align) enField : Field number (see. 6.21)
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode

	eERROR_SVM_NOT_CREATED_VIEWMODE: not generated view mode eERROR_INVALID_ALIGN : 16byte align error eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.26. PPAPI_SVM_GetLRLUTAddress

Prototype	PP_U32* PPAPI_SVM_GetLRLUTAddress(PP_VIEWMODE_E IN enView, PP_FIELD_E IN enField);
Description	Get memory base address of LR LUT
Argument	enView : View mode (see. 5) enField : Field number (see. 6.21)
Return value	Memory base address of LR LUT PP_NULL
Example	

6.27. PPAPI_SVM_SetBCLUTAddress

Prototype	PP_RESULT_E PPAPI_SVM_SetBCLUTAddress(PP_VIEWMODE_E IN enView, PP_U32* IN pu32Addr, PP_FIELD_E IN enField);
Description	Set memory base address of BC LUT
Argument	enView : View mode (see. 5) pu32Addr : Memory base address (16byte align) enField : Field number (see. 6.21)
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: not generated view mode eERROR_INVALID_ALIGN : 16byte align error eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.28. PPAPI_SVM_GetBCLUTAddress

Prototype	PP_U32* PPAPI_SVM_GetBCLUTAddress(PP_VIEWMODE_E IN enView, PP_FIELD_E IN enField);
Description	Get memory base address of BC LUT
Argument	enView : View mode (see. 5) enField : Field number (see. 6.21)
Return value	Memory base address of LR LUT PP_NULL
Example	

6.29. PPAPI_SVM_SetBCAdditionalLUT [Option]

Prototype	PP_RESULT_E PPAPI_SVM_SetBCAdditionalLUT(PP_VIEWMODE_E IN enView, PP_SVMDRV_BC_ADDITIONAL_LUT_E IN enType, PP_SVMDRV_BC_ADDITIONAL_LUT_SUBCORE_E IN enSubCore, PP_U32* IN pu32Addr, PP_U32 IN u32Size, PP_FIELD_E IN enField);
Description	Set memory base address and size of BC Additional LUT
Argument	enView : View mode (see. 5) enType: BC Additional LUT type typedef enum ppSVMDRV_BC_ADDITIONAL_LUT_E { eSVMDRV_BC_ADD_LUT_ALPHA_0 = 0, eSVMDRV_BC_ADD_LUT_ALPHA_1, eSVMDRV_BC_ADD_LUT_GRADIENT, eSVMDRV_BC_ADD_LUT_MAX, } PP_SVMDRV_BC_ADDITIONAL_LUT_E; stBCAdd : BC additional LUT의 size와 memory base address typedef enum ppSVMDRV_BC_ADDITIONAL_LUT_SUBCORE_E { eSVMDRV_BC_ADD_LUT_SUBCORE_0 = 0, eSVMDRV_BC_ADD_LUT_SUBCORE_1,

	eSVMDRV_BC_ADD_LUT_SUBCORE_MAX, } PP_SVMDRV_BC_ADDITIONAL_LUT_SUBCORE_E; pu32Addr : Memory base address (16byte align) u32Size : Size of BC additional LUT enField : Field number (see. 6.21)
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: Not generated view mode eERROR_INVALID_ALIGN : 16byte align error eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.30. PPAPI_SVM_SetImageRect

Prototype	PP_RESULT_E PPAPI_SVM_SetImageRect(PP_VIEWMODE_E IN enView, PP_SVMDRV_IMG_NUMBER_E IN enImgNum, PP_RECT_S* IN stRect);
Description	Set rectangle area of YUV422(YUYV) image This function is used in shadow image output.
Argument	enView : View mode (see. 5) enImgNum : image number (Max two image) typedef enum ppSVMDRV_IMG_NUMBER_E { eSVMDRV_IMG_NUM_0 = 0, eSVMDRV_IMG_NUM_1, eSVMDRV_IMG_NUM_MAX, } PP_SVMDRV_IMG_NUMBER_E; stRect : Rectangle area if PP_NULL, Image off typedef struct ppRECT_S { PP_U16 u16X; PP_U16 u16Y; PP_U16 u16Width;

	PP_U16 u16Height; } PP_RECT_S;
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: not generated view mode eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.31. PPAPI_SVM_SetImageAlphaBlending

Prototype	PP_RESULT_E PPAPI_SVM_SetImageAlphaBlending(PP_VIEWMODE_E IN enView, PP_SVMDRV_IMG_NUMBER_E IN enImgNum, PP_U8 IN u8Alpha);
Description	Set transparency of YUV422(YUYV) image. This function is used in shadow image output.
Argument	enView : View mode (see. 5) enImgNum : image number (see. 6.30) u8Alpha : Alpha value
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: not generated view mode eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.32. PPAPI_SVM_SetImageAddress

Prototype	PP_RESULT_E PPAPI_SVM_SetBCLUTAddress(PP_VIEWMODE_E IN enView, PP_SVMDRV_IMG_NUMBER_E IN enImgNum, PP_U32* IN pu32Addr, PP_FIELD_E IN enField);
Description	Set memory base address of YUV422(YUYV) image. This function is used in shadow image output.

Argument	enView : View mode (see. 5) enImgNum : image number (see. 6.30) pu32Addr : Memory base address (16byte align) enField : Field number (see. 6.21)
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: not generate view mode eERROR_INVALID_ALIGN : 16byte align error eERROR_INVALID_ARGUMENT : Argument value error
Example	

6.33. PPAPI_SVM_GetImageAddress

Prototype	PP_U32* PPAPI_SVM_GetBCLUTAddress(PP_VIEWMODE_E IN enView, PP_SVMDRV_IMG_NUMBER_E IN enImgNum, PP_FIELD_E IN enField);
Description	Get memory base address of YUV422(YUYV) image. This function is used in shadow image output.
Argument	enView : View mode (see. 5) enImgNum : image number (see. 6.30) enField : Field number (see. 6.21)
Return value	Memory base address of YUV422(YUYV) image PP_NULL
Example	

6.34. PPAPI_SVM_SetSectionRect

Prototype	PP_RESULT_E PPAPI_SVM_SetSectionRect(PP_VIEWMODE_E IN enView, PP_SVMDRV_SECTION_NUMBER_E IN enNum, PP_RECT_S* IN stRect);
Description	Set section rectangle (see. Fig 6-3)
Argument	enView : View mode (see. 5) enSectionNumber : Section Number

	typedef enum ppSVMDRV_SECTION_NUMBER_E { eSVMDRV_SECTION_NUM_0 = 0, eSVMDRV_SECTION_NUM_1, eSVMDRV_SECTION_NUM_2, eSVMDRV_SECTION_NUM_3, eSVMDRV_SECTION_NUM_MAX, } PP_SVMDRV_SECTION_NUMBER_E; stRect : Section area (see. 6.30) If PP_NULL, Section off
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_SVM_LIMIT_VIEWMODE : Invalid view mode eERROR_SVM_NOT_CREATED_VIEWMODE: not generated view mode eERROR_INVALID_ARGUMENT : Argument value error
Example	

Fig 6-3. Setion area

6.35. PPAPI_SVM_SetAutoSectionRect

Prototype	PP_RESULT_E PPAPI_SVM_SetAutoSectionRect(PP_BOOL bOn);

Description	Set section rectangle in all view mode automatically.
Argument	bOn : PP_TRUE or PP_FALSE
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized
Example	

6.36. PPAPI_SVM_SetView

Prototype	PP_RESULT_E PPAPI_SVM_SetView(PP_VIEWMODE_E IN enView, PPAPI_SVM_SWITCHING_CALLBACK_DISPLAY IN cbDisplayCallback, PPAPI_SVM_SWITCHING_CALLBACK_FINISH IN cbFinishCallback);
Description	Swich to a specific view mode - Swing view mode : (see. 4.3) - The others view mode :After loading LUT from flash, view mode is switched. (see. 4.2)
Argument	enView : Target View mode (see. 5) cbDisplayCallback : Callback Handler called to set UI after 1 frame typedef PP_VOID (*PPAPI_SVM_SWITCHING_CALLBACK_DISPLAY) (PP_VIEWMODE_E enView); or PP_NULL cbFinishCallback : Callback Handler to be called when new view mode is displayed typedef PP_VOID (*PPAPI_SVM_SWITCHING_CALLBACK_FINISH) (PP_VOID); or PP_NULL
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_NOT_SUPPORT : enView is not swing view mode. eERROR_INVALID_ARGUMENT : Argument value error eERROR_NO_MEM : Heap memory error eERROR_SVM_NOT_LOADING : Fail to load data from flash eERROR_SVM_MORPHING : Morphing error
Example	PP_VOID EndView(PP_VOID) {

	// Module to be executed when view is switched
	}
	
	PP_VOID Display(PP_VIEWMODE_E enView)
	{
	// Set display for eVIEWMODE_LOAD_3D_FRONT
	}
	
	// Use cbDisplayCallback & cbFinishCallback
	PPAPI_SVM_SetView(eVIEWMODE_LOAD_3D_FRONT, Display, EndVie);
	
	// Use cbDisplayCallback , Not use cbFinishCallback
	PPAPI_SVM_SetView(eVIEWMODE_LOAD_3D_FRONT, Display,
	PP_NULL);
	
	// Not use cbDisplayCallback & cbFinishCallback
	PPAPI_SVM_SetView(eVIEWMODE_LOAD_3D_FRONT, PP_NULL,
	PP_NULL);

6.37. PPAPI_SVM_SetSwingView

Prototype	PP_RESULT_E PPAPI_SVM_SetSwingView(PP_VIEWMODE_E IN enDegreeStartView, PP_VIEWMODE_E IN enDegreeEndView, PP_SVMAPI_SWING_DIRECTION_E IN enDirection, PP_U32 IN u32RepeatCount, PPAPI_SVM_SWITCHING_CALLBACK_DISPLAY IN cbDisplayCallback, PPAPI_SVM_SWITCHING_CALLBACK_FINISH IN cbFinishCallback);
Description	Switch swing view mode continuously (see. 4.3) Load a certain amount of data using Cache task in advance The remaining LUT data will be loaded in the background during switching view mode by cash task.
Argument	enDegreeStartView : Start view mode (see. 5) eVIEWMODE_LOAD_TOP2D_SWING_START ~ eVIEWMODE_LOAD_TOP2D_SWING_MAX

	enDegreeEndtView : Last view mode (see. 5) eVIEWMODE_LOAD_TOP2D_SWING_START ~ eVIEWMODE_LOAD_TOP2D_SWING_MAX enDirection : Direction of rotation typedef enum ppSVMAPI_SWING_DIRECTION_E { eSVMAPI_SWING_DIRECTION_AUTO = 0, eSVMAPI_SWING_DIRECTION_CLOCKWISE, eSVMAPI_SWING_DIRECTION_COUNTERCLOCKWISE, eSVMAPI_SWING_DIRECTION_MAX, } PP_SVMAPI_SWING_DIRECTION_E; u32RepeatCount : Repeat number of switch from enStarView to enEndView SVMAPI_REPEAT_LIMITLESS : Endless repeat cbDisplayCallback : Callback Handler to be called for UI setting after 1 frame typedef PP_VOID (*PPAPI_SVM_SWITCHING_CALLBACK_DISPLAY)(PP_VIEWMODE_E enView); or PP_NULL cbFinishCallback : Callback Handler to be called when last view mode is displayed typedef PP_VOID (*PPAPI_SVM_SWITCHING_CALLBACK_FINISH)(PP_VOID); or PP_NULL
Return value	eSUCCESS eERROR_SVM_NOT_INITIALIZE : SVM Block is not initialized eERROR_NOT_SUPPORT : enDegreeStartView or enDegreeEndView is not swing view mode. eERROR_INVALID_ARGUMENT : Argument value error eERROR_NO_MEM : Heap memory error eERROR_SVM_NOT_LOADING : Fail to load data from flash eERROR_SVM_MORPHING : Morphing error
Example	

6.38. PPAPI_SVM_GetLoadedViewCount

Prototype	PP_U32 PPAPI_SVM_GetCreatedViewCount(PP_VOID);
Description	Get the number of view mode stored in flash
Argument	
Return value	
Example	

6.39. PPAPI_SVM_GetCurrentView

Prototype	PP_VIEWMODE_E PPAPI_SVM_GetCurrentView(PP_VOID);
Description	Get current view mode
Argument	
Return value	View mode
Example	

7. How to use

7.1. Configuration for swing

Set swing.cnf to control swing angle and morphing.

[PI5008KSvmPkgTool path]\data\[view data]\config\swing.cnf

```
[Information]
name=Swing Configuration
version=1.0
[Swing]
degreeInterval=2
FBLRLutInterval=8
staticView=0,44,90,134,180,224,270,314
```

- degreeInterval (default value = 2) : Set the interval of angle.
- FBLRLutInterval (default value = 8) : Set the interval of morphing.

ex) degreeInterval=2 & FBLRLutInterval=8

Degree	0	2	4	6	8	10	12	14	16	....
LUT	FB				FB				FB	....
	LR				LR				LR	....
	BC	BC	BC	BC	BC	BC	BC	BC	BC	....

- staticView

Set the angle to be applied to viewmode_config.h.

ex) staticView=0,44,90,134,180,224,270,314

```
typedef enum ppVIEWMODE_E
{
    ....................
    eVIEWMODE_LOAD_TOP2D_SWING_START,
    eVIEWMODE_LOAD_TOP2D_SWING_0=eVIEWMODE_LOAD_TOP2D_SWING_START + 0,
    eVIEWMODE_LOAD_TOP2D_SWING_44=eVIEWMODE_LOAD_TOP2D_SWING_START + 22,
    eVIEWMODE_LOAD_TOP2D_SWING_90=eVIEWMODE_LOAD_TOP2D_SWING_START + 45,
    eVIEWMODE_LOAD_TOP2D_SWING_134=eVIEWMODE_LOAD_TOP2D_SWING_START + 67,
    eVIEWMODE_LOAD_TOP2D_SWING_180=eVIEWMODE_LOAD_TOP2D_SWING_START + 90,
    eVIEWMODE_LOAD_TOP2D_SWING_224=eVIEWMODE_LOAD_TOP2D_SWING_START + 112,
```

```
    eVIEWMODE_LOAD_TOP2D_SWING_270=eVIEWMODE_LOAD_TOP2D_SWING_START + 135,
    eVIEWMODE_LOAD_TOP2D_SWING_314=eVIEWMODE_LOAD_TOP2D_SWING_START + 157,
    eVIEWMODE_LOAD_TOP2D_SWING_MAX=eVIEWMODE_LOAD_TOP2D_SWING_START + 180,
     ..................
} PP_VIEWMODE_E;
```

7.2. Make binary for Flash memory

To display surround view, view modes and LUTs for each view mode have to be generated in advance and downloaded to flash memory. PI5008KViewGenTool will be used to generate these information and convert it to the files which can be recognized by PI5008K.

After making view mode, binary file and header file will be generated by clicking <Make SVM binary> button of PI5008KSvmPkgTool.

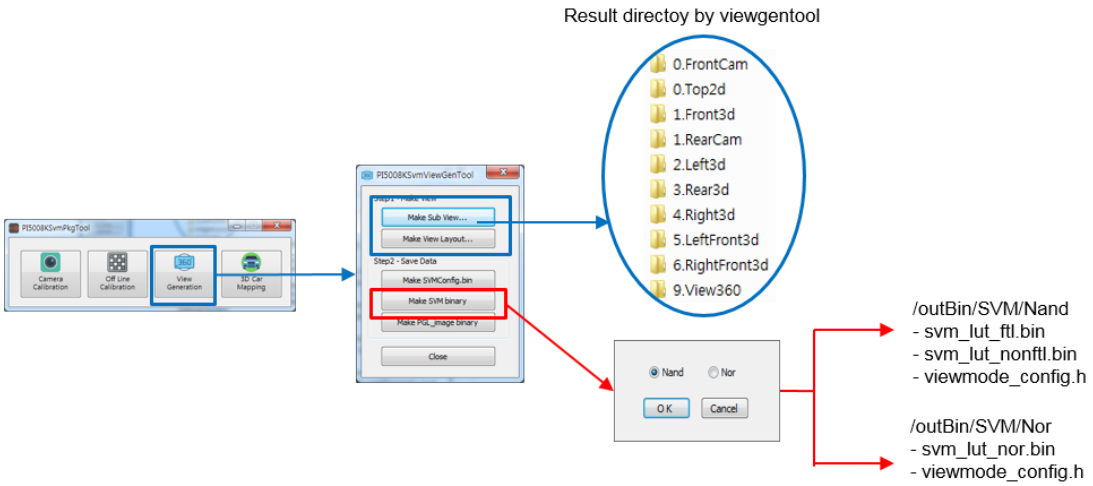

Fig 7-1. Make SVM Binary

7.3. Change PP_VIEWMODE_E (viewmode_config.h)

The generated viewmode_config.h applies to the firmware. (see. 7.2)

This enum has to have same order which is used to save view mode to flash memory. (see. 5)

7.3.1. Example

viewmode_config.h

```
#define VIEWMODE_NULL    (0xFFFFFFFF)
```

```
typedef enum ppVIEWMODE_E
{
    eVIEWMODE_LOAD_2D_FRONT          = 0,
    eVIEWMODE_LOAD_2D_REAR,
    eVIEWMODE_LOAD_3D_FRONT,
    eVIEWMODE_LOAD_3D_LEFT,
    eVIEWMODE_LOAD_3D_LEFTFRONT,
    eVIEWMODE_LOAD_3D_REAR,
    eVIEWMODE_LOAD_3D_RIGHTFRONT,
    eVIEWMODE_LOAD_3D_RIGHT,
    eVIEWMODE_LOAD_TOP2D_SWING_START,
    eVIEWMODE_LOAD_TOP2D_SWING_0=eVIEWMODE_LOAD_TOP2D_SWING_START + 0,
    eVIEWMODE_LOAD_TOP2D_SWING_44=eVIEWMODE_LOAD_TOP2D_SWING_START + 22,
    eVIEWMODE_LOAD_TOP2D_SWING_90=eVIEWMODE_LOAD_TOP2D_SWING_START + 45,
    eVIEWMODE_LOAD_TOP2D_SWING_134=eVIEWMODE_LOAD_TOP2D_SWING_START + 67,
    eVIEWMODE_LOAD_TOP2D_SWING_180=eVIEWMODE_LOAD_TOP2D_SWING_START + 90,
    eVIEWMODE_LOAD_TOP2D_SWING_224=eVIEWMODE_LOAD_TOP2D_SWING_START + 112,
    eVIEWMODE_LOAD_TOP2D_SWING_270=eVIEWMODE_LOAD_TOP2D_SWING_START + 135,
    eVIEWMODE_LOAD_TOP2D_SWING_314=eVIEWMODE_LOAD_TOP2D_SWING_START + 157,
    eVIEWMODE_LOAD_TOP2D_SWING_MAX=eVIEWMODE_LOAD_TOP2D_SWING_START + 180,

    eVIEWMODE_LOAD_MAX,

    eVIEWMODE_BASIC_FRONT_BYPASS = eVIEWMODE_LOAD_MAX,
    eVIEWMODE_BASIC_LEFT_BYPASS,
    eVIEWMODE_BASIC_RIGHT_BYPASS,
    eVIEWMODE_BASIC_REAR_BYPASS,
    eVIEWMODE_BASIC_QUAD,

    eVIEWMODE_TOTAL_MAX,
} PP_VIEWMODE_E;
```

7.4. Set resolution and frame rate

Select input/output resolution and frame rate.

7.4.1. Example

- Input : HD720P@30
- Output : HD720P@30

board_config.h

```
#define BD_VIN_FMT          ((VID_RESOL_HD720P) | (VID_FRAME_NTSC_30))
#define BD_SVM_IN_FMT       (BD_VIN_FMT)
#define BD_SVM_OUT_FMT      ((VID_RESOL_HD720P) | (VID_FRAME_NTSC_30))
#define BD_DU_IN_FMT        ((BD_SVM_OUT_FMT))
```

7.5. Initialization

Call PPAPI_SVMMEM_Initialize(). (see 6.2)

PPAPI_SVMMEM_Initialize() has to be called between PPAPI_FLASH_Initialize() and init_dram_reserved() .

PPAPI_SVMMEM_Initialize() has to be called before PPAPI_SVM_Initialize() .

Call PPAPI_SVM_Initialize(). (see. 6.1)

7.5.1. Example

main.c

```
int main(void)
{
    ……..

    if(PPAPI_FLASH_Initialize(SPI_NAND_FLASH_PAGE_SIZE,
                              SPI_NAND_FLASH_ERASE_BLOCK_SIZE,
                              eFLASH_TYPE_NAND);

    PPAPI_FTL_Initialize(NAND_DEVICE_MT29FXG01, &stStats);
    PPAPI_FLASH_ReadHeader();
```

```
    PPAPI_FLASH_ReadFTLHeader();

#ifdef CACHE_VIEW_USE
    // Memory setting for using swing view mode
    PPAPI_SVMMEM_Initialize(eVIEWMODE_LOAD_TOP2D_SWING_START,
                            eVIEWMODE_LOAD_TOP2D_SWING_MAX);
#else
    // Memory setting for not using swing view mode
    PPAPI_SVMMEM_Initialize(VIEWMODE_NULL, VIEWMODE_NULL);
#endif

    init_dram_reserved();

    ..................

    PPAPI_SVM_Initialize();

    ..................

    return 1;
}
```

7.6. View mode switching

PPAPI_SVM_SetView() (see. 6.36) or PPAPI_SVM_SetSwingView() (see. 6.37) are used to switch view mode. If UI has to be changed with view mode switching, UI callback handler(cbDisplayCallback) can be used.

If there is something to be done after view mode switching is finished, Finish callback handler(cbFinishCallback) can be used.

7.6.1. Example

7.6.1.1. Changing one view mode

```
PP_VOID FinishHandler(PP_VOID)
```

```
{
    // Setting after changing view mode
}

PP_VOID UIHandler(PP_VIEWMODE_E enView)
{
    // UI Setting of eVIEWMODE_LOAD_3D_FRONT view mode
}

PPAPI_SVM_SetView(eVIEWMODE_LOAD_3D_FRONT,
                  UIHandler,
                  FinishHandler);
```

7.6.1.2. Changing swing view mode

```
PP_VOID FinishHandler(PP_VOID)
{
    // Setting after changing all view mode
    // ex) change the other view mode after 360°  swing
}

PP_VOID UIHandler(PP_VIEWMODE_E enView)
{
    /* UI Setting of eVIEWMODE_LOAD_TOP2D_SWING_0 ~
       eVIEWMODE_LOAD_TOP2D_SWING_MAX view mode */
    switch (enView)
    {
    case eVIEWMODE_LOAD_TOP2D_SWING_0:
        // UI setting of eVIEWMODE_LOAD_TOP2D_SWING_0
        break;
    case eVIEWMODE_LOAD_TOP2D_SWING_44:
        // UI setting of eVIEWMODE_LOAD_TOP2D_SWING_44
        break;
    ……
    }
}
```

```
PPAPI_SVM_SetSwingView(eVIEWMODE_LOAD_TOP2D_SWING_0,
                       eVIEWMODE_LOAD_TOP2D_SWING_44,
                       1,
                       UIHandler,
                           FinishHandler);
```

7.7. Make Flash image and Download

main_firm.bin and generated SVM binary uses PI5008KDownload&MergeTool to create flash image and download to flash.

If the svm binary for nand flash is changed, both svm_lut_ftl.bin and svm_lut_nonftl.bin should be downloaded to nand flash.

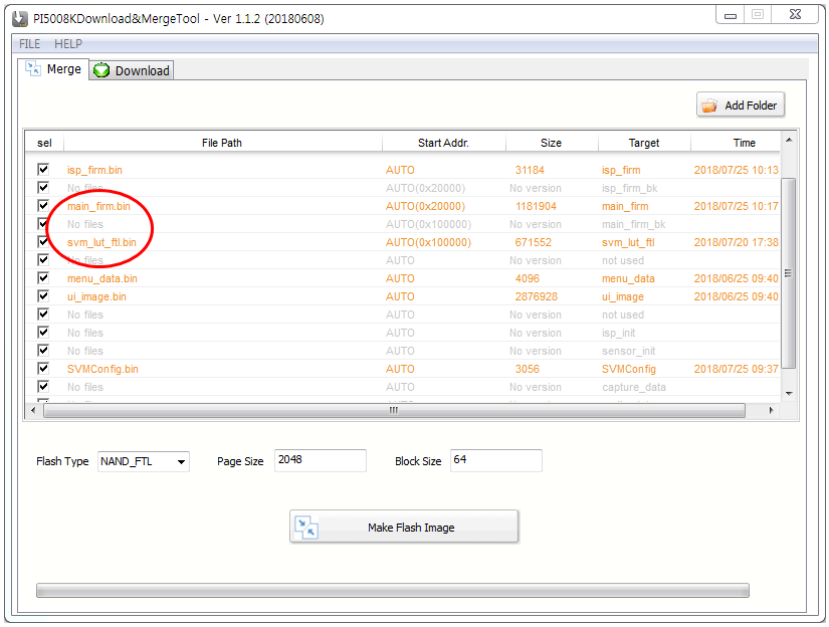

Fig 7-2. Make flashimage_FTL.bin and Download

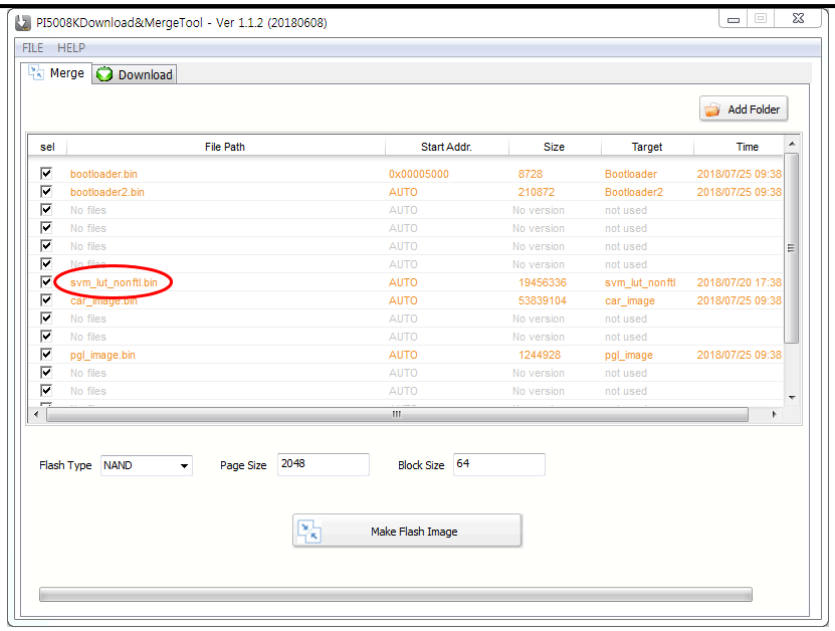

Fig 7-3. Make flashimage_nonFTL.bin and Download

8. Revision History

Version	Date	Description
V0.1	20180608	
V0.2	20180727	Apply section of view mode Add PPAPI_SVM_SetAutoSectionRect() Add PPAPI_SVM_GetLoadedViewCount() Delete PPAPI_SVM_GetCreatedViewCount()
		Delete PPAPI_SVM_SetContinuousView() Add PPAPI_SVM_SetSwingView() Update "7.How to use"