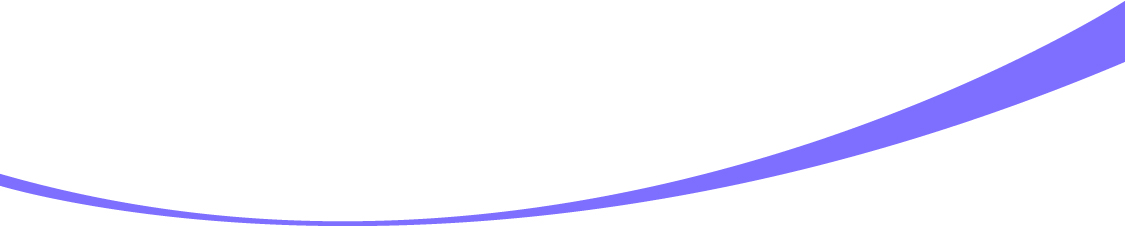
API Reference

PI5008 API Reference

Version 2

Mar. 2017



Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author(s) |
| 0.10 | Feb. 10, 2017 | API 초안  VPU API 추가 | yhcho |
| 0.20 | Mar. 13, 2017 | System API 추가  SVM API 추가  VPU API 수정  - DMA copy 추가 | ihkong  yhcho  mijin |
|  |  |  |  |

Contents

[Document History 2](#_Toc477870913)

[Contents 3](#_Toc477870914)

[1 Type 6](#_Toc477870915)

[2 Error code 7](#_Toc477870916)

[3 SYSTEM API 8](#_Toc477870917)

[3.1 Data Type 8](#_Toc477870918)

[3.1.1 UART\_BAUDRATE\_E 8](#_Toc477870919)

[3.1.2 UART\_DATABIT\_E 8](#_Toc477870920)

[3.1.3 UART\_PARITY\_E 9](#_Toc477870921)

[3.1.4 UART\_STOP\_BIT\_E 9](#_Toc477870922)

[3.1.5 DMA\_EVENT\_RESULT\_E 9](#_Toc477870923)

[3.1.6 SYSDMA\_CALLBACK 10](#_Toc477870924)

[3.2 API 10](#_Toc477870925)

[3.2.1 SYSAPI\_initialize 10](#_Toc477870926)

[3.2.2 SYSAPI\_UART\_initialize 11](#_Toc477870927)

[3.2.3 SYSAPI\_DMA\_initialize 11](#_Toc477870928)

[3.2.4 SYSAPI\_DMA\_m2m\_word 12](#_Toc477870929)

[3.2.5 SYSAPI\_DMA\_check\_result 13](#_Toc477870930)

[3.2.6 SYSAPI\_DMA\_abort 13](#_Toc477870931)

[3.2.7 SYSAPI\_CACHE\_writeback 14](#_Toc477870932)

[3.2.8 SYSAPI\_CACHE\_invalidate 14](#_Toc477870933)

[4 VPU(Video Processing Unit) API 15](#_Toc477870934)

[4.1 Data Type 15](#_Toc477870935)

[4.1.1 VPU\_CHANNEL\_E 15](#_Toc477870936)

[4.1.2 VPU\_FRAME\_E 15](#_Toc477870937)

[4.1.3 VPU\_SIZE\_T 16](#_Toc477870938)

[4.1.4 VPU\_RECT\_T 16](#_Toc477870939)

[4.1.5 VPU\_FAST\_PARAM\_T 17](#_Toc477870940)

[4.1.6 VPU\_BRIEF\_LUT\_VALUE\_T 17](#_Toc477870941)

[4.1.7 VPU\_BRIEF\_LUT\_T 18](#_Toc477870942)

[4.1.8 VPU\_FAST\_RESULT\_POS\_T 18](#_Toc477870943)

[4.1.9 VPU\_BRIEF\_RESULT\_DESC\_T 19](#_Toc477870944)

[4.2 API 19](#_Toc477870945)

[4.2.1 VPUAPI\_FB\_initialize 19](#_Toc477870946)

[4.2.2 VPUAPI\_FB\_deInitialize 20](#_Toc477870947)

[4.2.3 VPUAPI\_FB\_setChannel 20](#_Toc477870948)

[4.2.4 VPUAPI\_FB\_setScale 21](#_Toc477870949)

[4.2.5 VPUAPI\_FB\_setScaleEnable 21](#_Toc477870950)

[4.2.6 VPUAPI\_FB\_setROI 21](#_Toc477870951)

[4.2.7 VPUAPI\_FB\_setROIEnable 22](#_Toc477870952)

[4.2.8 VPUAPI\_FB\_setZone 22](#_Toc477870953)

[4.2.9 VPUAPI\_FAST\_setParam 23](#_Toc477870954)

[4.2.10 VPUAPI\_BRIEF\_setLut 23](#_Toc477870955)

[4.2.11 VPUAPI\_BRIEF\_getLut 24](#_Toc477870956)

[4.2.12 VPUAPI\_FB\_start 24](#_Toc477870957)

[4.2.13 VPUAPI\_FAST\_getResult 25](#_Toc477870958)

[4.2.14 VPUAPI\_BRIEF\_getResult 25](#_Toc477870959)

[4.2.15 VPUAPI\_getHammingDistance 26](#_Toc477870960)

[4.2.16 VPUAPI\_getHammingDistance\_mem 27](#_Toc477870961)

[4.2.17 VPUAPI\_FB\_DMA\_copy 27](#_Toc477870962)

[5 SVM(Surround View Monitoring) API 29](#_Toc477870963)

[5.1 Error code 29](#_Toc477870964)

[5.2 Data Type 29](#_Toc477870965)

[5.2.1 SVM\_USE\_LUT 29](#_Toc477870966)

[5.2.2 SVM\_UNUSE\_LUT 29](#_Toc477870967)

[5.2.3 SVM\_USE\_CAR 29](#_Toc477870968)

[5.2.4 SVM\_UNUSE\_CAR 29](#_Toc477870969)

[5.2.5 SVM\_CHANNEL\_E 30](#_Toc477870970)

[5.2.6 SVM\_BURST\_LENGTH\_E 30](#_Toc477870971)

[5.2.7 SVM\_OUTPUTMODE \_E 31](#_Toc477870972)

[5.2.8 SVM\_COMPRESSION\_RATIO\_E 32](#_Toc477870973)

[5.2.9 SVM\_BOUNDARYSCALE\_E 32](#_Toc477870974)

[5.2.10 SVM\_BOUNDARYAEAR\_E 33](#_Toc477870975)

[5.2.11 SVM\_EDGE\_MODE\_E 33](#_Toc477870976)

[5.2.12 SVM\_COLOR\_E 34](#_Toc477870977)

[5.2.13 SVM\_OUTPUT\_PART\_E 35](#_Toc477870978)

[5.2.14 SVM\_SUBPART\_E 35](#_Toc477870979)

[5.2.15 SVM\_FB\_PART\_E 36](#_Toc477870980)

[5.2.16 SVM\_LR\_PART\_E 36](#_Toc477870981)

[5.2.17 SVM\_PAGEFLIPPING\_NUMBER\_E 37](#_Toc477870982)

[5.2.18 SVM\_SECTION\_NUMBER\_E 37](#_Toc477870983)

[5.2.19 SVM\_ROI\_AREA\_T 38](#_Toc477870984)

[5.2.20 SVM\_BOUNDARY\_RANGE\_T 38](#_Toc477870985)

[5.2.21 SVM\_BOUNDARY\_T 39](#_Toc477870986)

[5.2.22 SVM\_COLORCOEF\_R2Y\_T 39](#_Toc477870987)

[5.2.23 SVM\_COLORCOEF\_Y2R\_T 40](#_Toc477870988)

[5.2.24 SVM\_COLORCOEF\_T 41](#_Toc477870989)

[5.2.25 SVM\_RECT\_T 41](#_Toc477870990)

[5.2.26 SVM\_COLORCOEF\_T 42](#_Toc477870991)

[5.2.27 SVM\_RGB\_STATISTICS\_T 43](#_Toc477870992)

[5.3 API 44](#_Toc477870993)

[5.3.1 SVMAPI\_initialize 44](#_Toc477870994)

[5.3.2 SVMAPI\_deInitialize 45](#_Toc477870995)

[5.3.3 SVMAPI\_createViewMode 45](#_Toc477870996)

[5.3.4 SVMAPI\_deleteAllViewMode 46](#_Toc477870997)

[5.3.5 SVMAPI\_setViewMode 46](#_Toc477870998)

[5.3.6 SVMAPI\_setChannelInputAddr 47](#_Toc477870999)

[5.3.7 SVMAPI\_getChannelInputAddr 48](#_Toc477871000)

[5.3.8 SVMAPI\_setChannelFilpping 48](#_Toc477871001)

[5.3.9 SVMAPI\_setCompression 49](#_Toc477871002)

[5.3.10 SVMAPI\_setCompressionEnable 49](#_Toc477871003)

[5.3.11 SVMAPI\_setROI 50](#_Toc477871004)

[5.3.12 SVMAPI\_setROIEnable 51](#_Toc477871005)

[5.3.13 SVMAPI\_setHorizontalScale 51](#_Toc477871006)

[5.3.14 SVMAPI\_setHoldTime 52](#_Toc477871007)

[5.3.15 SVMAPI\_setVerticalHalfScale 53](#_Toc477871008)

[5.3.16 SVMAPI\_setInputBurstLength 53](#_Toc477871009)

[5.3.17 SVMAPI\_setInputTestModeEnable 54](#_Toc477871010)

[5.3.18 SVMAPI\_setInputWrappingInitEnable 54](#_Toc477871011)

[5.3.19 SVMAPI\_setInputEnable 55](#_Toc477871012)

[5.3.20 SVMAPI\_setSync 56](#_Toc477871013)

[5.3.21 SVMAPI\_setBGColor 56](#_Toc477871014)

[5.3.22 SVMAPI\_setMaskColor 57](#_Toc477871015)

[5.3.23 SVMAPI\_setColorCoef 57](#_Toc477871016)

[5.3.24 SVMAPI\_setCar 57](#_Toc477871017)

[5.3.1 SVMAPI\_setCarReadBurstLength 58](#_Toc477871018)

[5.3.2 SVMAPI\_updateCar 58](#_Toc477871019)

[5.3.3 SVMAPI\_setCarAddr 59](#_Toc477871020)

[5.3.4 SVMAPI\_getCarAddr 59](#_Toc477871021)

[5.3.5 SVMAPI\_setCarEnable 60](#_Toc477871022)

[5.3.6 SVMAPI\_setEdgeEnhancementMode 60](#_Toc477871023)

[5.3.7 SVMAPI\_setEdgeEnhancementFixedGain 61](#_Toc477871024)

[5.3.8 SVMAPI\_setEdgeEnhancementEnable 61](#_Toc477871025)

[5.3.9 SVMAPI\_setDynamicblendingCoef 62](#_Toc477871026)

[5.3.10 SVMAPI\_setDynamicblendingEnable 62](#_Toc477871027)

[5.3.11 SVMAPI\_setMorphingSpeed 63](#_Toc477871028)

[5.3.12 SVMAPI\_updateBCCoef 63](#_Toc477871029)

[5.3.13 SVMAPI\_updateBCRatio 64](#_Toc477871030)

[5.3.14 SVMAPI\_getPageNum4Lut 64](#_Toc477871031)

[5.3.15 SVMAPI\_updateFBLUT 65](#_Toc477871032)

[5.3.16 SVMAPI\_updateLRLUT 65](#_Toc477871033)

[5.3.17 SVMAPI\_updateBCLUT 66](#_Toc477871034)

[5.3.18 SVMAPI\_setFBLUTAddr 66](#_Toc477871035)

[5.3.19 SVMAPI\_setLRLUTAddr 67](#_Toc477871036)

[5.3.20 SVMAPI\_setBCLUTAddr 67](#_Toc477871037)

[5.3.21 SVMAPI\_getFBLUTAddr 68](#_Toc477871038)

[5.3.22 SVMAPI\_getLRLUTAddr 68](#_Toc477871039)

[5.3.23 SVMAPI\_getBCLUTAddr 69](#_Toc477871040)

[5.3.24 SVMAPI\_getRGBStatistics 69](#_Toc477871041)

[5.3.25 SVMAPI\_setSVMEnable 70](#_Toc477871042)

[5.3.26 SVMAPI\_setBlank 70](#_Toc477871043)

[5.3.27 SVMAPI\_setSection 71](#_Toc477871044)

[5.3.28 SVMAPI\_setSectionEnable 72](#_Toc477871045)

[5.3.29 SVMAPI\_setOutputBurstLength 72](#_Toc477871046)

[5.3.30 SVMAPI\_setInvalidLUTEnable 73](#_Toc477871047)

[5.3.31 SVMAPI\_setOutputEnable 73](#_Toc477871048)

[5.3.32 SVMAPI\_setCBOutputVsync 73](#_Toc477871049)

# Type

|  |
| --- |
| typedef char int8;  typedef unsigned char uint8;  typedef short int16;  typedef unsigned short uint16;  typedef int int32;  typedef unsigned int uint32;  typedef long long int int64;  typedef unsigned long long int uint64;  typedef volatile unsigned int vuint32;  typedef enum {FALSE, TRUE} bool;  #define APICALL extern  #define APIENTRY  #define IN  #define OUT |

# Error code

|  |
| --- |
| enum tagERROR\_E  {  eSUCCESS = 0,  eERROR\_INVALIED\_ARGUMENT, eERROR\_MEM\_ALLOC,  eERROR\_SVM\_NOT\_INITIALIZE = 100, eERROR\_SVM\_VIEWMODE\_COUNT\_ZERO, eERROR\_SVM\_VIEWMODE\_NOT\_SELECTED,  eERROR\_SVM\_TOO\_MAANY\_LIMIT\_VIEWMODE,  eERROR\_SVM\_UNUSE\_VIEWMODE,  eERROR\_SVM\_UNUSE\_LUT\_VIEWMODE,  eERROR\_SVM\_UNUSE\_CAR\_VIEWMODE,  eERROR\_SVM\_RUNNING,  eERROR\_VPU\_NOT\_INITIALIZE = 200, eERROR\_VPU\_INVALIED\_ARGUMENT\_TOO\_MAANY\_LIMIT\_FEATURE\_COUNT, eERROR\_VPU\_TOO\_MAANY\_LIMIT\_FEATURE\_COUNT\_1ST\_FRAME, eERROR\_VPU\_TOO\_MAANY\_LIMIT\_FEATURE\_COUNT\_2ND\_FRAME, eERROR\_VPU\_TOO\_MAANY\_LIMIT\_FEATURE\_COUNT\_BOTH\_FRAME,  eERROR\_SYS\_DMA\_INVALID\_CHANNEL = 300, eERROR\_SYS\_DMA\_CHANNEL\_IN\_USE, eERROR\_SYS\_DMA\_ABORT, eERROR\_SYS\_DMA\_ERROR,  }; |

# SYSTEM API

## Data Type

### UART\_BAUDRATE\_E

[Syntax]

|  |
| --- |
| enum tagUART\_BAUDRATE\_E  {  eBAUDRATE\_2400 = 2400,  eBAUDRATE\_4800 = 4800,  eBAUDRATE\_9600 = 9600,  eBAUDRATE\_19200 = 19200,  eBAUDRATE\_38400 = 38400,  eBAUDRATE\_57600 = 57600,  eBAUDRATE\_115200 = 115200,  eBAUDRATE\_230400 = 230400,  eBAUDRATE\_460800 = 460800,  eBAUDRATE\_921600 = 921600,  }; |

[Description]

UART baud rate enumeration.

### UART\_DATABIT\_E

[Syntax]

|  |
| --- |
| enum tagUART\_DATABIT\_E  {  eDATA\_BIT\_5 = 0,  eDATA\_BIT\_6 = 1,  eDATA\_BIT\_7 = 2,  eDATA\_BIT\_8 = 3,  }; |

[Description]

UART data bit enumeration.

### UART\_PARITY\_E

[Syntax]

|  |
| --- |
| enum tagUART\_PARITY\_E  {  ePARITY\_NONE = 0x0,  ePARITY\_ODD = 0x8,  ePARITY\_EVEN = 0x18,  }; |

[Description]

UART parity enumeration.

### UART\_STOP\_BIT\_E

[Syntax]

|  |
| --- |
| enum tagUART\_STOP\_BIT\_E  {  eSTOP\_BIT\_1 = 0x0,  eSTOP\_BIT\_2 = 0x4,  }; |

[Description]

UART stop bit enumeration.

### DMA\_EVENT\_RESULT\_E

[Syntax]

|  |
| --- |
| enum tagDMA\_EVENT\_RESULT\_E  {  eDMA\_EVENT\_DONE = 0,  eDMA\_EVENT\_ABORT,  eDMA\_EVENT\_ERROR,  }; |

[Description]

DMA event result enumeration.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eDMA\_EVENT\_DONE | DMA 완료 이벤트 |
| eDMA\_EVENT\_ABORT | DMA 취소 이벤트 |
| eDMA\_EVENT\_ERROR | DMA 오류 이벤트 |

### SYSDMA\_CALLBACK

[Syntax]

|  |
| --- |
| typedef void (\*SYSDMA\_CALLBACK)(uint8 ch, uint32 event) |

[Description]

DMA 결과를 받을 수 있는 callback 함수

[Member]

|  |  |
| --- | --- |
| Member | Description |
| ch | DMA를 수행한 channel |
| event | enum DMA\_EVENT\_RESULT\_E |

## API

### SYSAPI\_initialize

[Syntax]

|  |
| --- |
| int32 APIENTRY SYSAPI\_initialize(void); |

[Description]

System 초기화

[Parameter]

[Return]

|  |  |
| --- | --- |
| Value | Description |
| eSUCCESS | 성공 |
|  |  |

### SYSAPI\_UART\_initialize

[Syntax]

|  |
| --- |
| int32 APIENTRY SYSAPI\_UART\_initialize(uint8 IN ch, int32 IN baudRate, int32 IN dataBit, int32 IN stopBit, int32 IN parity); |

[Description]

UART 초기화.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| channel | UART channel |
| baudRate | Baud rate |
| dataBit | Data bit |
| stopBit | Stop bit |
| parity | Parity |

[Return]

|  |  |
| --- | --- |
| Value | Description |
| eSUCCESS | 성공 |
|  |  |

### SYSAPI\_DMA\_initialize

[Syntax]

|  |
| --- |
| void APIENTRY SYSAPI\_DMA\_initialize(SYSDMA\_CALLBACK IN callback); |

[Description]

DMA 초기화. DMA engine을 software reset 시키고, 이벤트를 받기 위한 callback 함수를 등록한다. 등록한 callback함수를 통해 DMA 완료, 취소, 오류 이벤트를 받을 수 있다. 이벤트를 받지 않으려면 NULL 을 입력 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| callback | DMA 결과를 받을 수 있는 callback 함수. |

[Return]

### SYSAPI\_DMA\_m2m\_word

[Syntax]

|  |
| --- |
| int32 APIENTRY SYSAPI\_DMA\_m2m\_word(uint8 IN ch, uint32 IN \*pPhySrc, uint32 IN \*pPhyDst, int32 IN size, bool IN isBlock); |

[Description]

Source memory 에서 destination memory 로 word 단위의 DMA를 수행한다. Address 와 size는 word 단위로 정렬되어야 한다. isBlock 변수가 TRUE 으로 설정될 경우 DMA 가 완료될 때까지 기다린 후 리턴 되지만 그렇지 않을 경우 함수는 즉시 리턴 된다. DMA 의 완료는 SYSAPI\_DMA\_initialize 함수를 통해 등록한 이벤트 callback 함수를 통해서 확인 하거나 SYSAPI\_DMA\_check\_result 함수를 폴링해서 확인 가능하다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| ch | DMA 를 수행할 channel |
| pPhySrc | Source memory 의 physical address |
| pPhyDst | Destination memory 의 physical address |
| size | DMA를 수행할 크기 바이트 |
| isBlock | DMA 수행 완료를 기다릴지 결정 |

[Return]

|  |  |
| --- | --- |
| Value | Description |
| eSUCCESS | 성공 |
| eERROR\_SYS\_DMA\_INVALID\_CHANNEL | 잘못된 채널 |
| eERROR\_SYS\_DMA\_CHANNEL\_IN\_USE | DMA 진행중 |
| eERROR\_SYS\_DMA\_ABORT | DMA 취소 |
| eERROR\_SYS\_DMA\_ERROR | DMA 오류 |

### SYSAPI\_DMA\_check\_result

[Syntax]

|  |
| --- |
| int32 APIENTRY SYSAPI\_DMA\_check\_result(uint8 IN ch); |

[Description]

DMA 수행 결과를 확인.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| ch | DMA 를 수행한 channel |

[Return]

|  |  |
| --- | --- |
| Value | Description |
| eSUCCESS | 성공 |
| eERROR\_SYS\_DMA\_INVALID\_CHANNEL | 잘못된 채널 |
| eERROR\_SYS\_DMA\_CHANNEL\_IN\_USE | DMA 진행중 |
| eERROR\_SYS\_DMA\_ABORT | DMA 취소 |
| eERROR\_SYS\_DMA\_ERROR | DMA 오류 |

### SYSAPI\_DMA\_abort

[Syntax]

|  |
| --- |
| int32 APIENTRY SYSAPI\_DMA\_abort(uint8 IN ch); |

[Description]

수행중인 DMA를 취소.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| ch | DMA 를 수행한 channel |

[Return]

|  |  |
| --- | --- |
| Value | Description |
| eSUCCESS | 성공 |
| eERROR\_SYS\_DMA\_INVALID\_CHANNEL | 잘못된 채널 |

### SYSAPI\_CACHE\_writeback

[Syntax]

|  |
| --- |
| void APIENTRY SYSAPI\_CACHE\_writeback(uint32 IN pAddr, int32 IN size); |

[Description]

Cache writeback

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| pAddr | Memory address |
| size | Memory size |

[Return]

### SYSAPI\_CACHE\_invalidate

[Syntax]

|  |
| --- |
| void APIENTRY SYSAPI\_CACHE\_invalidate(uint32 IN pAddr, int32 IN size); |

[Description]

Cache invalidate

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| pAddr | Memory address |
| Size | Memory size |

[Return]

# VPU(Video Processing Unit) API

## Data Type

### VPU\_CHANNEL\_E

[Syntax]

|  |
| --- |
| enum tagVPU\_CHANNEL\_E  {  eVPU\_CHANNEL\_FRONT = 0,  eVPU\_CHANNEL\_LEFT,  eVPU\_CHANNEL\_RIGHT,  eVPU\_CHANNEL\_BACK,  eVPU\_CHANNEL\_SVM,  eVPU\_CHANNEL\_OUTPUT,  eVPU\_CHANNEL\_MAX,  }; |

[Description]

Channel의 종류

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eVPU\_CHANNEL\_FRONT | Front camera |
| eVPU\_CHANNEL\_LEFT | Left camera |
| eVPU\_CHANNEL\_RIGHT | Right camera |
| eVPU\_CHANNEL\_BACK | Back camera |
| eVPU\_CHANNEL\_SVM | SVM output |
| eVPU\_CHANNEL\_OUTPUT | Output video |

### VPU\_FRAME\_E

[Syntax]

|  |
| --- |
| enum tagVPU\_FRAME\_E  {  eVPU\_FRAME\_1ST = 0,  eVPU\_FRAME\_2ND,  eVPU\_FRAME\_MAX,  }; |

[Description]

FAST와 BREIF 수행하는 1st frame 또는 2nd frame

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eVPU\_FRAME\_1ST | 1st frame |
| eVPU\_FRAME\_2ND | 2nd frame |

### VPU\_SIZE\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_SIZE\_T  {  uint16 width;  uint16 height;  } VPU\_SIZE\_T, \*pVPU\_SIZE\_T; |

[Description]

Size 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| width |  |
| height |  |

### VPU\_RECT\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_RECT\_T  {  uint16 x;  uint16 y;  uint16 width;  uint16 height;  } VPU\_RECT\_T, \*pVPU\_RECT\_T; |

[Description]

사각형 영역 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| x |  |
| y |  |
| width |  |
| height |  |

### VPU\_FAST\_PARAM\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_FAST\_PARAM\_T  {  uint8 number;  uint8 threshold;  } VPU\_FAST\_PARAM\_T, \*pVPU\_FAST\_PARAM\_T; |

[Description]

FAST 설정을 위한 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| number | FAST의 number parameter |
| threshold | FAST의 threshold |

### VPU\_BRIEF\_LUT\_VALUE\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_BRIEF\_LUT\_VALUE\_T  {  uint8 x1;  uint8 y1;  uint8 x2;  uint8 y2;  } VPU\_BRIEF\_LUT\_VALUE\_T, \*pVPU\_BRIEF\_LUT\_VALUE\_T; |

[Description]

BRIEF LUT 업데이트를 위한 Parameter 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| x1 | BRIEF LUT의 x1 parameter |
| y1 | BRIEF LUT의 y1 parameter |
| x2 | BRIEF LUT의 x2 parameter |
| y2 | BRIEF LUT의 y2 parameter |

### VPU\_BRIEF\_LUT\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_BRIEF\_LUT\_T  {  VPU\_BRIEF\_LUT\_VALUE\_T lut[128];  } VPU\_BRIEF\_LUT\_T, \*pVPU\_BRIEF\_LUT\_T; |

[Description]

BRIEF LUT 업데이트를 위한 128개의 parameter들의 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| lut | BRIEF LUT parameter의 배열 |

### VPU\_FAST\_RESULT\_POS\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_FAST\_RESULT\_POS\_T  {  uint16 x;  uint16 y;  } VPU\_FAST\_RESULT\_POS\_T, \*pVPU\_FAST\_RESULT\_POS\_T; |

[Description]

FAST feature point의 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| x | x feature point |
| Y | y feature point |

### VPU\_BRIEF\_RESULT\_DESC\_T

[Syntax]

|  |
| --- |
| typedef struct tagVPU\_BRIEF\_RESULT\_DESC\_T  {  uint32 descriptor[4];  } VPU\_BRIEF\_RESULT\_DESC\_T, \*pVPU\_BRIEF\_RESULT\_DESC\_T; |

[Description]

BRIEF descriptor의 구조체

[Member]

|  |  |
| --- | --- |
| Member | Description |
| descriptor | descriptor는 128bit |

## API

### VPUAPI\_FB\_initialize

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_initialize(VPU\_SIZE\_T IN inputSize, uint16 IN limitFeatureCount); |

[Description]

FAST와 BRIEF 초기화. FAST와 BRIEF를 수행하기 위해서는 제일 처음으로 호출해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| inputSize | 입력 사이즈 (width, height) |
| limitFeatureCount | Feature point가 생성되는 최대 개수 |

[Return]

Error code

### VPUAPI\_FB\_deInitialize

[Syntax]

|  |
| --- |
| void APIENTRY VPUAPI\_FB\_deInitialize(void); |

[Description]

FAST와 BRIEF 초기화 해제

[Parameter]

[Return]

### VPUAPI\_FB\_setChannel

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_setChannel(uint8 IN channel); |

[Description]

channel를 설정

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| channel | enum VPU\_CHANNEL\_E |

[Return]

Error code

### VPUAPI\_FB\_setScale

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_setScale(VPU\_SIZE\_T IN scaleSize); |

[Description]

입력 영상에 대한 scale을 설정한다. VPU는 최대 HD(1280x720)까지 지원하므로 그 이상이면 scale을 설정해야 한다. Scale, ROI, Zone은 동시에 사용 가능하며 수행 순서는Scale->ROI->Zone이다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| scaleSize | Scale size |

[Return]

Error code

### VPUAPI\_FB\_setScaleEnable

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_setScaleEnable(bool IN isEnable); |

[Description]

Scale 사용 여부

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | TRUE or FALSE |

[Return]

Error code

### VPUAPI\_FB\_setROI

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_setROI(VPU\_RECT\_T IN roi); |

[Description]

입력 영상에 대한 ROI영역을 설정한다. VPU는 최대 HD(1280x720)까지 지원하므로 그 이상이면 ROI 영역을 설정해야 한다. Scale, ROI, Zone은 동시에 사용 가능하며 수행 순서는Scale->ROI->Zone이다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| roi | roi 영역 |

[Return]

Error code

### VPUAPI\_FB\_setROIEnable

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_setROIEnable(bool IN isEnable); |

[Description]

ROI 영역의 사용여부

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | TRUE or FALSE |

[Return]

Error code

### VPUAPI\_FB\_setZone

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_setZone(uint8 IN zoneNumber, bool IN isEnable, VPU\_RECT\_T IN zone); |

[Description]

입력 영상에 대한 Zone영역을 설정한다. 4부분의 사각영역인 Zone을 설정하여 해당 영역에 대해서만 FAST와 BRIEF를 수행하도록 한다. Scale, ROI, Zone은 동시에 사용 가능하며 수행 순서는Scale->ROI->Zone이다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| zoneNumber | Zone 영역의 번호 (0~3) |
| isEnable | 사용 여부 |
| zone | Zone의 사각영역. isEnable이 TRUE일때만 사용된다. |

[Return]

Error code

### VPUAPI\_FAST\_setParam

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FAST\_setParam(VPU\_FAST\_PARAM\_T IN fastParam); |

[Description]

FAST parameter를 설정. FAST 수행 전에 설정해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| fastParma | FAST parameter |

[Return]

Error code

### VPUAPI\_BRIEF\_setLut

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_BRIEF\_setLut(pVPU\_BRIEF\_LUT\_T IN briefLut); |

[Description]

BRIEF LUT를 업데이트 한다. 기본적으로 저장되어 있는 LUT와 다른 값을 사용시 BRIEF 수행 전에 업데이트 해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| briefLut | BRIEF LUT 배열 |

[Return]

Error code

### VPUAPI\_BRIEF\_getLut

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_BRIEF\_getLut(pVPU\_BRIEF\_LUT\_T OUT briefLut); |

[Description]

설정되어 있는 BRIEF LUT를 읽어온다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| briefLut | BRIEF LUT 배열 |

[Return]

Error code

### VPUAPI\_FB\_start

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_start(uint8 IN frameInterval,  uint16\* OUT featureNumOf1st, uint16\* OUT featureNumof2nd); |

[Description]

FAST와 BRIEF를 수행한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| frameInterval | 수행할 2frame 간의 간격 |
| featureNumOf1st | 수행 후 첫번째 frame의 feature의 개수 |
| featureNumof2nd | 수행 후 두번째 frame의 feature의 개수 |

[Return]

Error code

### VPUAPI\_FAST\_getResult

[Syntax]

|  |
| --- |
| pVPU\_FAST\_RESULT\_POS\_T APIENTRY VPUAPI\_FAST\_getResult(uint8 IN frameNumber); |

[Description]

FAST 수행 후의 결과 값인 feature point을 얻는다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| frameNumber | frame 번호 ( 0 or 1) |

[Return]

|  |  |
| --- | --- |
| Member | Description |
| pVPU\_FAST\_RESULT\_POS\_T | Feature point 배열 |
| error | NULL |

### VPUAPI\_BRIEF\_getResult

[Syntax]

|  |
| --- |
| pVPU\_BRIEF\_RESULT\_DESC\_T APIENTRY VPUAPI\_BRIEF\_getResult(uint8 IN frameNumber); |

[Description]

BRIEF 수행 후의 결과 값인 descritor를 얻는다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| frameNumber | frame 번호 ( 0 or 1) |

[Return]

|  |  |
| --- | --- |
| Member | Description |
| pVPU\_BRIEF\_RESULT\_DESC\_T | Descriptor 배열 |
| error | NULL |

### VPUAPI\_getHammingDistance

[Syntax]

|  |
| --- |
| uint8\* APIENTRY VPUAPI\_getHammingDistance(pVPU\_BRIEF\_RESULT\_DESC\_T IN desc1,  uint32 IN descCount1, pVPU\_BRIEF\_RESULT\_DESC\_T IN desc2, uint8 IN descCount2); |

[Description]

N:M의 Hamming distance를 수행한다. srcDesc 배열의 index와 dstDesc 배열의 index를 순차적으로 증가시켜서 수행한다.

srcDesc[0] ------------- dstDesc[0] ---------> returnVale[0]

dstDesc[1] ---------> returnVale[1]

……..

dstDesc[(M-1)] ----------> returnVale[(M-1)]

srcDesc[1] ------------- dstDesc[0] ----------> returnVale[(M\*1)+0]

dstDesc[1] ----------> returnVale[(M\*1)+1]

……..

dstDesc[(M-1)] ----------> returnVale[(M\*1)+(M-1)]

srcDesc[(N-1)] ---------- dstDesc[0] ----------> returnVale[(M\*(N-1))+0]

dstDesc[1] ----------> returnVale[(M\*(N-1))+1]

……..

dstDesc[(M-1)] ---------> returnVale[(M\*(N-1))+(M-1)]

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| desc1 | 1st frame의 descriptor 배열 |
| descCount1 | 1st frame의 descriptor의 개수 |
| desc2 | 2nd frame의 descriptor 배열 |
| descCount2 | 2nd frame의 descriptor의 개수 |

[Return]

|  |  |
| --- | --- |
| Member | Description |
| uint8\* | hamming distance 배열 |
| error | NULL |

### VPUAPI\_getHammingDistance\_mem

[Syntax]

|  |
| --- |
| uint8\* APIENTRY VPUAPI\_getHammingDistance\_mem(pVPU\_BRIEF\_RESULT\_DESC\_T IN desc1,  pVPU\_BRIEF\_RESULT\_DESC\_T IN desc2, uint8 IN descCount); |

[Description]

N:N의 Hamming distance를 수행한다. srcDesc 배열의 index와 dstDesc 배열의 index를 순차적으로 증가시켜서 수행한다.

srcDesc[0] ------------- dstDesc[0] ---------> returnVale[0]

srcDesc[1] ------------- dstDesc[1] ----------> returnVale[1]

srcDesc[(N-1)] ---------- dstDesc[(N-1)] ----------> returnVale[(N-1)]

…..

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| desc1 | 1st frame의 descriptor 배열 |
| desc2 | 2nd frame의 descriptor 배열 |
| descCount | descriptor의 개수 |

[Return]

|  |  |
| --- | --- |
| Member | Description |
| uint8\* | hamming distance 배열 |
| error | NULL |

### VPUAPI\_FB\_DMA\_copy

[Syntax]

|  |
| --- |
| uint8 APIENTRY VPUAPI\_FB\_DMA\_copy (uint32\* IN src, uint32\* OUT dst, uint16 IN width, uint16 IN height) |

[Description]

VPU의 2D DMA를 이용한 copy를 수행한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| src | copy 하려는 source address |
| dst | copy 하려는 destination address |
| width | copy 하려는 width size |
| height | copy 하려는 height size |

[Return]

Error code

# SVM(Surround View Monitoring) API

## Error code

|  |  |
| --- | --- |
| Member | Description |
| eSUCCESS | 성공 |
| eERROR\_INVALIED\_ARGUMENT | 잘못된 인자 값을 사용 |
| eERROR\_SVM\_NOT\_INITIALIZE | 초기화를 수행하지 않았다. (SVMAPI\_initialize API 참조) |
| eERROR\_SVM\_VIEWMODE\_NOT\_SELECTED | View mode를 선택하지 않았다. |
| eERROR\_SVM\_TOO\_MAANY\_LIMIT\_VIEWMODE | 최대 View mode handler 이상을 생성했다. |
| eERROR\_SVM\_UNUSE\_VIEWMODE | 생성하지 않은 view mode를 사용했다. |
| eERROR\_SVM\_UNUSE\_LUT\_VIEWMODE | LUT를 사용하지 않는 view mode이다. |
| eERROR\_SVM\_UNUSE\_CAR\_VIEWMODE | Car image를 사용하지 않는 view mode이다. |
| eERROR\_SVM\_RUNNING | SVM이 수행 중일 때는 사용할 수 없다. |

## Data Type

### SVM\_USE\_LUT

[Description]

LUT를 사용한다.

### SVM\_UNUSE\_LUT

[Description]

LUT를 사용하지 않는다.

### SVM\_USE\_CAR

[Description]

자동차 이미지를 사용한다.

### SVM\_UNUSE\_CAR

[Description]

자동차 이미지를 사용하지 않는다.

### SVM\_CHANNEL\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_CHANNEL\_E  {  eSVM\_CHANNEL\_FRONT = 0,  eSVM\_CHANNEL\_LEFT,  eSVM\_CHANNEL\_RIGHT,  eSVM\_CHANNEL\_BACK,  eSVM\_CHANNEL\_MAX,  }; |

[Description]

Channel

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_CHANNEL\_FRONT | Front camera |
| eSVM\_CHANNEL\_LEFT | Left camera |
| eSVM\_CHANNEL\_RIGHT | Right camera |
| eSVM\_CHANNEL\_BACK | Back camera |

### SVM\_BURST\_LENGTH\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_BURST\_LENGTH  {  eSVM\_BURST\_LENGTH\_8 = 4,  eSVM\_BURST\_LENGTH\_16,  eSVM\_BURST\_LENGTH\_32,  eSVM\_BURST\_LENGTH\_64,  eSVM\_BURST\_LENGTH\_128,  eSVM\_BURST\_LENGTH\_MAX,  }; |

[Description]

입출력의 burst length 설정에 사용될 index 값이다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_BURST\_LENGTH\_8 | 8 burst length |
| eSVM\_BURST\_LENGTH\_16 | 16 burst length |
| eSVM\_BURST\_LENGTH\_32 | 32 burst length |
| eSVM\_BURST\_LENGTH\_64 | 64 burst length |
| eSVM\_BURST\_LENGTH\_128 | 128 burst length |

### SVM\_OUTPUTMODE \_E

[Syntax]

|  |
| --- |
| enum tagSVM\_OUTPUTMODE\_E  {  eSVM\_OUTPUTMODE\_BYPASS\_FRONT = 0,  eSVM\_OUTPUTMODE\_BYPASS\_LEFT,  eSVM\_OUTPUTMODE\_BYPASS\_RIGHT,  eSVM\_OUTPUTMODE\_BYPASS\_BACK,  eSVM\_OUTPUTMODE\_LUT\_HALF\_BLEND,  eSVM\_OUTPUTMODE\_LUT\_BCLUT\_ONLY\_ALPHA,  eSVM\_OUTPUTMODE\_LUT\_BCLUT,  eSVM\_OUTPUTMODE\_MAX,  }; |

[Description]

출력 타입을 지정한다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_OUTPUTMODE\_BYPASS\_FRONT | Front channel만 출력한다. |
| eSVM\_OUTPUTMODE\_BYPASS\_LEFT | Left channel만 출력한다. |
| eSVM\_OUTPUTMODE\_BYPASS\_RIGHT | Right channel만 출력한다. |
| eSVM\_OUTPUTMODE\_BYPASS\_BACK | Back channel만 출력한다. |
| eSVM\_OUTPUTMODE\_LUT\_HALF\_BLEND | FB & LR LUT 사용  FB & LR이 겹치는 부분은 1/2 blending |
| eSVM\_OUTPUTMODE\_LUT\_BCLUT\_ONLY\_ALPHA | FB & LR LUT 사용  Brightness control LUT에서 Alpha값만 사용 |
| eSVM\_OUTPUTMODE\_LUT\_BCLUT | FB & LR LUT 사용  Brightness control LUT 사용 |

### SVM\_COMPRESSION\_RATIO\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_COMPRESSION\_RATIO\_E  {  eSVM\_COMPRESSION\_RATIO\_HALF = 0,  eSVM\_COMPRESSION\_RATIO\_QUARTER,  eSVM\_COMPRESSION\_MAX,  }; |

[Description]

입력 채널들에 대한 압축률의 종류이다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_COMPRESSION\_RATIO\_HALF | 1/2 압축 |
| eSVM\_COMPRESSION\_RATIO\_QUARTER | 1/4 압축 |

### SVM\_BOUNDARYSCALE\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_BOUNDARYSCALE\_E  {  eSVM\_BOUNDARYSCALE\_ONE = 0,  eSVM\_BOUNDARYSCALE\_ONE\_OVER\_TWO,  eSVM\_BOUNDARYSCALE\_ONE\_OVER\_FOUR,  eSVM\_BOUNDARYSCALE\_ONE\_OVER\_EIGHT,  eSVM\_BOUNDARYSCALE\_MAX,  }; |

[Description]

가로 scale down의 scale 종류이다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_BOUNDARYSCALE\_ONE | Scale down을 하지 않는다. |
| eSVM\_BOUNDARYSCALE\_ONE\_OVER\_TWO | 1/2 scale down |
| eSVM\_BOUNDARYSCALE\_ONE\_OVER\_FOUR | 1/4 scale down |
| eSVM\_BOUNDARYSCALE\_ONE\_OVER\_EIGHT | 1/8 scale down |

### SVM\_BOUNDARYAEAR\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_BOUNDARYAEAR\_E  {  eSVM\_BOUNDARYAEAR\_0 = 0,  eSVM\_BOUNDARYAEAR\_1,  eSVM\_BOUNDARYAEAR\_2,  eSVM\_BOUNDARYAEAR\_3,  eSVM\_BOUNDARYAEAR\_MAX,  }; |

[Description]

가로 scale down의 세로로 나누어 지는 4부분이다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_BOUNDARYAEAR\_0 | 0번째 부분 |
| eSVM\_BOUNDARYAEAR\_1 | 1번째 부분 |
| eSVM\_BOUNDARYAEAR\_2 | 2번째 부분 |
| eSVM\_BOUNDARYAEAR\_3 | 3번째 부분 |

### SVM\_EDGE\_MODE\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_EDGE\_MODE\_E  {  eSVM\_EDGE\_MODE\_LUT = 0,  eSVM\_EDGE\_MODE\_FIXED\_GAIN,  eSVM\_EDGE\_MODE\_MAX,  }; |

[Description]

Edge enhancement mode.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_EDGE\_MODE\_LUT | Brightness control LUT를 사용한다. |
| eSVM\_EDGE\_MODE\_FIXED\_GAIN | 고정된 값을 사용한다. |

### SVM\_COLOR\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_COLOR\_E  {  eSVM\_COLOR\_R = 0,  eSVM\_COLOR\_G,  eSVM\_COLOR\_B,  eSVM\_COLOR\_MAX,  }; |

[Description]

RGB color 영역. Brightness control을 위한 계수 및 RGB 색 통계 값을 위해 사용된다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_COLOR\_R | Red |
| eSVM\_COLOR\_G | Green |
| eSVM\_COLOR\_B | Blue |

### SVM\_OUTPUT\_PART\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_OUTPUT\_PART\_E  {  eSVM\_OUTPUT\_PART\_FB = 0,  eSVM\_OUTPUT\_PART\_LR,  eSVM\_OUTPUT\_PART\_MAX,  }; |

[Description]

출력 부분에 대한 영역 구분 및 LUT에 구분에 사용된다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_OUTPUT\_PART\_FB | Front and Back |
| eSVM\_OUTPUT\_PART\_LR | Left and Right |

### SVM\_SUBPART\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_SUBPART\_E  {  eSVM\_SUBPART\_FL = 0,  eSVM\_SUBPART\_FR,  eSVM\_SUBPART\_BR,  eSVM\_SUBPART\_BL,  eSVM\_SUBPART\_MAX,  }; |

[Description]

Front, Left, Right, Back의 겹치는 영역. Brightness control위한 계수 및 RGB 색 통계 값을 위해

사용된다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_SUBPART\_FL | Front and Left |
| eSVM\_SUBPART\_FR | Front and Right |
| eSVM\_SUBPART\_BR | Back and Right |
| eSVM\_SUBPART\_BL | Back and Left |

### SVM\_FB\_PART\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_FB\_PART\_E  {  eSVM\_FB\_PART\_F = 0,  eSVM\_FB\_PART\_B,  eSVM\_FB\_PART\_MAX,  }; |

[Description]

Front/Back에서 Front, Back 구분 영역. Brightness control의 계수를 설정하기 위해 사용된다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_FB\_PART\_F | Front |
| eSVM\_FB\_PART\_B | Back |

### SVM\_LR\_PART\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_LR\_PART\_E  {  eSVM\_LR\_PART\_L = 0,  eSVM\_LR\_PART\_R,  eSVM\_LR\_PART\_MAX,  }; |

[Description]

Left/Right에서 Left, Right 구분 영역. Brightness control의 계수를 설정하기 위해 사용된다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_LR\_PART\_L | Left |
| eSVM\_LR\_PART\_R | Right |

### SVM\_PAGEFLIPPING\_NUMBER\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_PAGEFLIPPING\_NUMBER\_E  {  eSVM\_PAGEFLIPPING\_NUMBER\_0 = 0,  eSVM\_PAGEFLIPPING\_NUMBER\_1,  eSVM\_PAGEFLIPPING\_NUMBER\_MAX,  }; |

[Description]

LUT의 page flipping에 사용할 page 번호.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_PAGEFLIPPING\_NUMBER\_0 | Page flipping의 첫 번째 번호 |
| eSVM\_PAGEFLIPPING\_NUMBER\_1 | Page flipping의 두 번째 번호 |

### SVM\_SECTION\_NUMBER\_E

[Syntax]

|  |
| --- |
| enum tagSVM\_SECTION\_NUMBER\_E  {  eSVM\_SECTION\_NUMBER\_0 = 0,  eSVM\_SECTION\_NUMBER\_1,  eSVM\_SECTION\_NUMBER\_MAX,  }; |

[Description]

Section 번호

[Member]

|  |  |
| --- | --- |
| Member | Description |
| eSVM\_SECTION\_NUMBER\_0 | Section의 첫 번째 번호 |
| eSVM\_SECTION\_NUMBER\_0 | Section의 두 번째 번호 |

### SVM\_ROI\_AREA\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_ROI\_AREA\_T  {  uint16 startYLine;  uint16 endYLine;  } SVM\_ROI\_AREA\_T, \*pSVM\_ROI\_AREA\_T; |

[Description]

ROI영역은 세로 라인의 시작 값과 마지막 값을 사용한다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| startYLine | 세로의 시작 라인 |
| endYLine | 세로의 마지막 라인 |

### SVM\_BOUNDARY\_RANGE\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_BOUNDARY\_RANGE\_T  {  uint16 startYLine;  uint8 scale;  } SVM\_BOUNDARY\_RANGE\_T, \*pSVM\_BOUNDARY\_RANGE; |

[Description]

가로 scale down을 위한 한 부분의 세로 라인 시작 위치와 scale 값이다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| startYLine | 세로의 시작 라인 |
| scale | Scale down 값 (SVM\_BOUNDARYSCALE\_E 참조) |

### SVM\_BOUNDARY\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_BOUNDARY\_T  {  SVM\_BOUNDARY\_RANGE\_T area[eSVM\_BOUNDARYAEAR\_MAX];  } SVM\_BOUNDARY\_T, \*pSVM\_BOUNDARY; |

[Description]

가로 scale down을 위한 각 부분을 나타낸다.

[Member]

|  |  |
| --- | --- |
| Member | Description |
| area | 4부분의 영역 |

### SVM\_COLORCOEF\_R2Y\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_COLORCOEFR2Y\_T  {  uint8 r4y;  uint8 g4y;  uint8 b4y;  uint8 r4cb;  uint8 g4cb;  uint8 b4cb;  uint8 r4cr;  uint8 g4cr;  uint8 b4cr;  } SVM\_COLORCOEF\_R2Y\_T, \*pSVM\_COLORCOEF\_R2Y\_T; |

[Description]

RGB to YCbCr에 사용될 계수. SVM\_COLORCOEF\_T 참조

[Member]

|  |  |
| --- | --- |
| Member | Description |
| r4y | Red to Y를 위한 계수 |
| g4y | Green to Y를 위한 계수 |
| b4y | Blue to Y를 위한 계수 |
| r4cb | Red to Cb를 위한 계수 |
| g4cb | Green to Cb를 위한 계수 |
| b4cb | Blue to Cb를 위한 계수 |
| r4cr | Red to Cr를 위한 계수 |
| g4cr | Green to Cr를 위한 계수 |
| b4cr | Blue to Cr를 위한 계수 |

### SVM\_COLORCOEF\_Y2R\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_COLORCOEF\_Y2R\_T  {  uint16 cr4r;  uint8 cr4g;  uint8 cb4g;  uint16 cb4b;  } SVM\_COLORCOEF\_Y2R\_T, \*pSVM\_COLORCOEF\_Y2R; |

[Description]

YCbCr to RGB에 사용될 계수. SVM\_COLORCOEF\_T 참조

[Member]

|  |  |
| --- | --- |
| Member | Description |
| cr4r | Cr to Red를 위한 계수 |
| cr4g | Cr to Green를 위한 계수 |
| cb4g | Cb to Green를 위한 계수 |
| cb4b | Cb to Blue를 위한 계수 |

### SVM\_COLORCOEF\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_COLORCOEF\_T  {  bool isWith128;  SVM\_COLORCOEF\_R2Y\_T r2y;  SVM\_COLORCOEF\_Y2R\_T y2r;  } SVM\_COLORCOEF\_T, \*pSVM\_COLORCOEF\_T; |

[Description]

RGB와 YCbCr과의 변환을 위해 사용될 계수

|  |
| --- |
| Y = coefr4y \* R + coefg4y \* G + coefb4y \* B  Cb = - coefr4cb \* R - coefg4cb \* G + coefb4cb \* B + 128  Cr = coefr4cr \* R - coefg4cr \* G - coefb4cr \* B + 128  R = Y + coefcr4r \* (Cr – 128)  G = Y – coefcb4g \* (Cb – 128) – coefcr4g \* (Cr – 128)  B = Y + coefcb4b \* (Cb – 128) |

[Member]

|  |  |
| --- | --- |
| Member | Description |
| isWith128 | 변환식에 128을 쓸 것인지 여부 |
| r2y | RGB to YCbCr의 Coefficient |
| y2r | YCbCr to RGB의 Coefficient |

### SVM\_RECT\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_RECT\_T  {  uint16 x;  uint16 y;  uint16 width;  uint16 height;  } SVM\_RECT\_T, \*pSVM\_RECT\_T; |

[Description]

사각 영역

[Member]

|  |  |
| --- | --- |
| Member | Description |
| x | 사각영역 x의 시작 위치 |
| y | 사각영역 y의 시작 위치 |
| width | 사각영역 가로 길이 |
| height | 사각영역 세로 길이 |

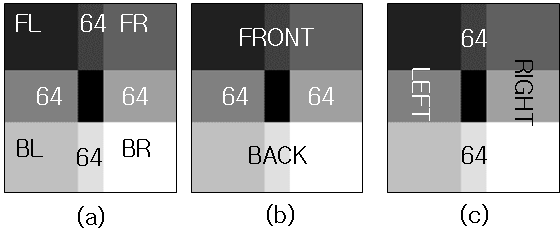
### SVM\_COLORCOEF\_T

[Syntax]

|  |
| --- |
| typedef struct tagSVM\_BCCOEF\_T  {  uint8 coef1[eSVM\_COLOR\_MAX][eSVM\_SUBPART\_MAX];  uint8 coef2[eSVM\_COLOR\_MAX][eSVM\_FB\_PART\_MAX];  uint8 coef3[eSVM\_COLOR\_MAX][eSVM\_SUBPART\_MAX];  uint8 coef4[eSVM\_COLOR\_MAX][eSVM\_LR\_PART\_MAX];  } SVM\_BCCOEF\_T, \*pSVM\_BCCOEF\_T; |

[Description]

Brightness control의 LUT를 사용하기 위한 계수



[Member]

|  |  |
| --- | --- |
| Member | Description |
| coef1 | 그림 (a) 영역의 값 |
| coef2 | 그림 (b)의 front와 back 영역의 값 |
| coef3 | 그림 (a) 영역의 값 |
| coef4 | 그림 (c)의 left와 right 영역의 값 |

### SVM\_RGB\_STATISTICS\_T

[Syntax]

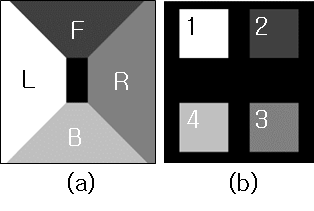
|  |
| --- |
| typedef struct tagSVM\_RGB\_STATISTICS\_T  {  uint32 value[eSVM\_OUTPUT\_PART\_MAX][eSVM\_SUBPART\_MAX][eSVM\_COLOR\_MAX];  } SVM\_RGB\_STATISTICS\_T, \*pSVM\_RGB\_STATISTICS\_T; |

[Description]

RGB 통계 값의 구조체.

아래 그림의 (a)와 같이 4개의 이미지가 merge된다고 할 때 (b)의 1번 사각 영역은 F와 L이 섞이는 영역이고 2는 F와 R이 섞이는 영역이 된다. 이 영역의 통계값 중 FB 이미지의 R 통계값은

FL\_R0에 LR 이미지의 R 통계값은 FL\_R1에 저장된다.



[Member]

|  |  |
| --- | --- |
| Member | Description |
| value | 각 영역에 대한 색 통계 값 |

## API

### SVMAPI\_initialize

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_initialize(uint16 IN inputWidth, uint16 IN inputHeight, uint16 IN outputWidth,  uint16 IN outputHeight, uint8 IN samplingCount, float IN inputFrameCount); |

[Description]

SVM API를 사용하기 위해 제일 처음으로 수행되어야 하는 초기화 함수이다. 입출력 사이즈, LUT에 대한 sampling 간격의 인자를 받는과 입력 채널들에 대하여 메모리에 저장할 frame 수를 정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| inputWidth | 입력 채널들의 width |
| inputHeight | 입력 채널들의 height |
| outputWidth | 출력 width |
| outputHeight | 출력 height |
| samplingCount | LUT에 대한 sampling 간격 |
| inputFrameCount | 각각의 입력 채널의 메모리 사이즈를 frame 수로 결정한다. |

[Return]

Error code

### SVMAPI\_deInitialize

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_deInitialize(void); |

[Description]

초기화를 해제한다.

[Parameter]

[Return]

### SVMAPI\_createViewMode

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_createViewMode(uint8 IN setFBLUT, uint8 IN setLRLUT, uint8 IN setBCLUT,  uint8 IN setCarImage, uint8 IN outputMode, uint8 OUT \*viewMode); |

[Description]

View mode를 생성한다. View mode는 하나의 Scene에 대한 처리를 수행하는 handler의 번호로 각 Scene에 대한 FB/LR LUT, BC LUT, Car 등의 출력 형태를 미리 정의한다. 여러 View mode를 생성하여 서로간의 변경을 간편하게 해준다. 세부적인 View mode에서의 설정은 아래에서 나오는 여러 함수를 통해서 설정 할 수 있다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| setFBLUT | Front & Back의 look up table 설정  SVM\_USE\_LUT : LUT를 사용하도록 하여 메모리 영역을 할당한다.  SVM\_UNUSE\_LUT : LUT를 사용하지 않도록 하여 메모리 영역을  할당하지 않는다.  이전에 생성한 View mode 번호 : 해당 View mode의 LUT을 공유해서  사용한다. 그러므로 LUT가 변경되면 공유한 모든 View mode의  LUT가 변경이 된다. |
| setLRLUT | Left & Right의 look up table 설정 (setFBLUT 참조) |
| setBCLUT | Brightness control의 look up table 설정(setFBLUT 참조) |
| setCarImage | Car image 설정(setFBLUT 참조)  SVM\_USE\_CAR, SVM\_UNUSE\_CAR, View mode 번호 |
| outputMode | 출력 타입 설정 (SVM\_OUTPUTMODE\_E 참조) |
| \*viewMode | 생성된 View mode handler의 번호 |

[Return]

Error code

### SVMAPI\_deleteAllViewMode

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_deleteAllViewMode(void); |

[Description]

생성한 View mode handler를 전부 삭제한다.

[Parameter]

[Return]

### SVMAPI\_setViewMode

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setViewMode(uint8 IN viewMode, bool IN isFBMorphing, bool IN isLRMorphing); |

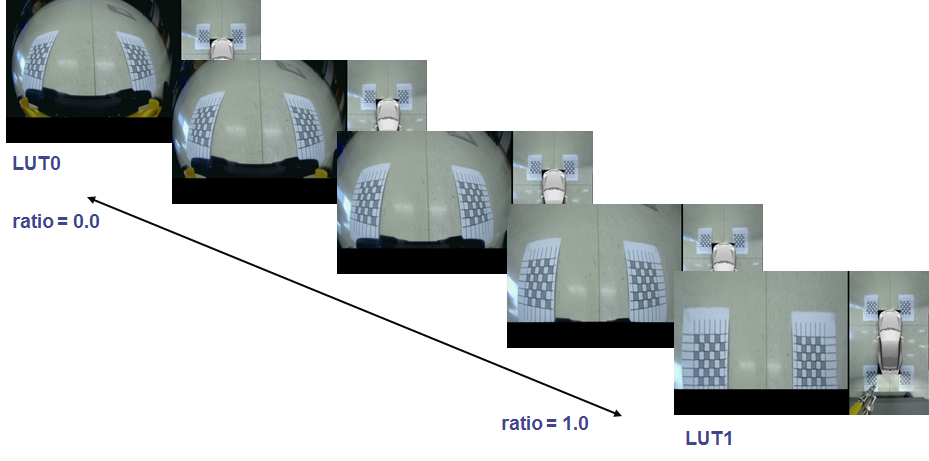
[Description]

생성한 View mode를 설정한다. 이 함수를 통해 View mode에 따른 출력 형태가 변경된다.

현 View mode에서 다른 View mode로 변경 시 Morphing 기능을 사용할 수 있다.

Morphing 기능을 통해 한번에 LUT가 변화하는 것보다 사용자가 직관적으로 이해할 수 있도록 해 준다. Morphing은 여러 카메라가 섞이는 부분을 제외한 한 카메라만 출력되는 영역에 대해서만

가능하다. 그러므로 Morphing을 위해 두 개의 LUT 생성시 여러 카메라가 섞이는 부분은 같은 LUT 값을 가져야 하며 한 카메라가 출력되는 부분만 LUT 값이 달라야 한다.



[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| isFBMorphing | Front&Back LUT에서 Morphing 수행 여부 |
| isLRMorphing | Left&Right LUT에서 Morphing 수행 여부 |

[Return]

Error code

### SVMAPI\_setChannelInputAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setChannelInputAddr(uint8 IN channel, uint32 IN addr); |

[Description]

각 채널 입력 데이터를 저장하는 메모리의 주소를 변경한다.

메모리 주소는 SVMAPI\_initialize()를 통해 설정이 자동으로 설정된다. 그러므로 강제적으로 설정이 필요할 때만 사용 해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| channel | Channel 번호 (SVM\_CHANNEL\_E 참조) |
| addr | 메모리 주소 |

[Return]

Error code

### SVMAPI\_getChannelInputAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_getChannelInputAddr(uint32 OUT \*channel0Addr, uint32 OUT \*channel1Addr,  uint32 OUT \*channel2Addr, uint32 OUT \*channel3Addr); |

[Description]

설정된 각 채널의 메모리 주소를 얻는다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| \*channel0Addr | Front의 메모리 주소 |
| \*channel1Addr | Left의 메모리 주소 |
| \*channel2Addr | Right의 메모리 주소 |
| \*channel3Addr | Back의 메모리 주소 |

[Return]

Error code

### SVMAPI\_setChannelFilpping

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setChannelFilpping(uint8 IN channel, bool IN isHorizontal, bool IN isVertical); |

[Description]

각 채널의 flipping을 설정한다. 입력 채널이 flipping되어 들어 왔을 때 이 함수를 통해 바르게 입력되도록 변경 할 수 있다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| channel | Channel 번호 (SVM\_CHANNEL\_E 참조) |
| isHorizontal | 가로에 대한 flipping |
| isVertical | 세로에 대한 flipping |

[Return]

Error code

### SVMAPI\_setCompression

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setCompression(uint8 IN viewMode, uint8 IN ratio); |

[Description]

입력 채널의 압축률을 설정한다.

모든 입력 채널에 대하여 메모리 write와 read시 압축을 사용하여 Band width을 줄여준다.

압축은 손실 압축이다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| ratio | 압축률 (SVM\_COMPRESSION\_RATIO\_E 참조) |

[Return]

Error code

### SVMAPI\_setCompressionEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setCompressionEnable(uint8 IN viewMode, bool IN isEnable); |

[Description]

입력 채널의 압축을 활성 또는 비활성 한다.

모든 입력 채널에 대하여 메모리 write와 read시 압축을 사용하여 Band width을 줄여준다.

압축은 손실 압축이다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setROI

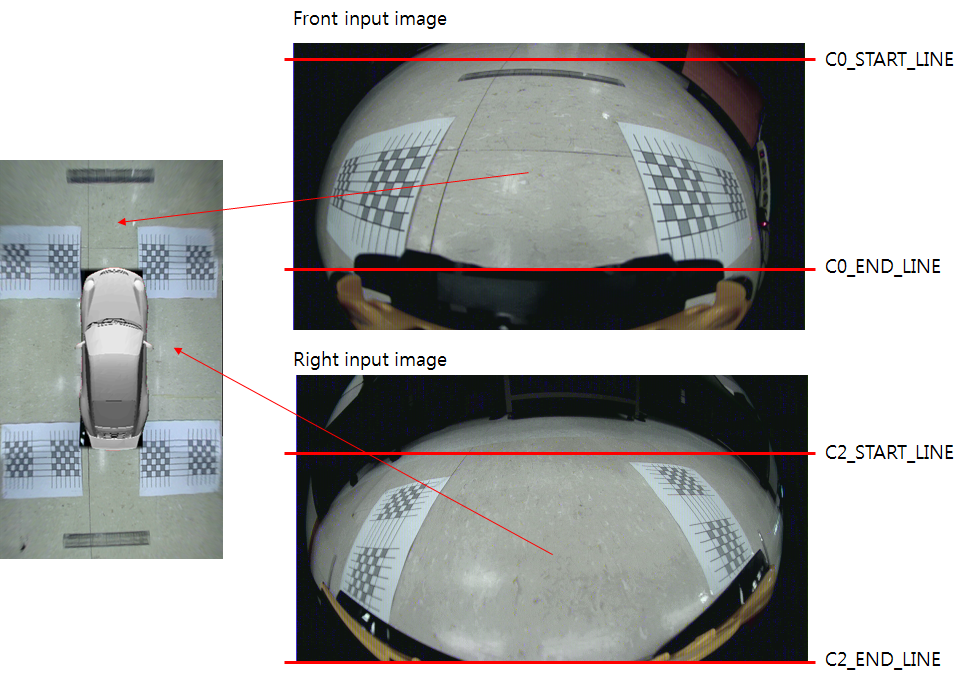
[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setROI(uint8 IN viewMode, uint8 IN channel, SVM\_ROI\_AREA\_T IN roi); |

[Description]

각 입력 채널의 ROI 영역을 설정한다.

각 채널의 ROI 영역을 설정하여 Band width을 줄여준다.



[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| channel | Channel 번호 (SVM\_CHANNEL\_E 참조) |
| roi | ROI 영역 (SVM\_ROI\_AREA\_T 참조) |

[Return]

Error code

### SVMAPI\_setROIEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setROIEnable(uint8 IN viewMode, uint8 IN channel, bool IN isEnable); |

[Description]

각 입력 채널의 ROI 설정을 활성 또는 비활성 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| channel | Channel 번호 (SVM\_CHANNEL\_E 참조) |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

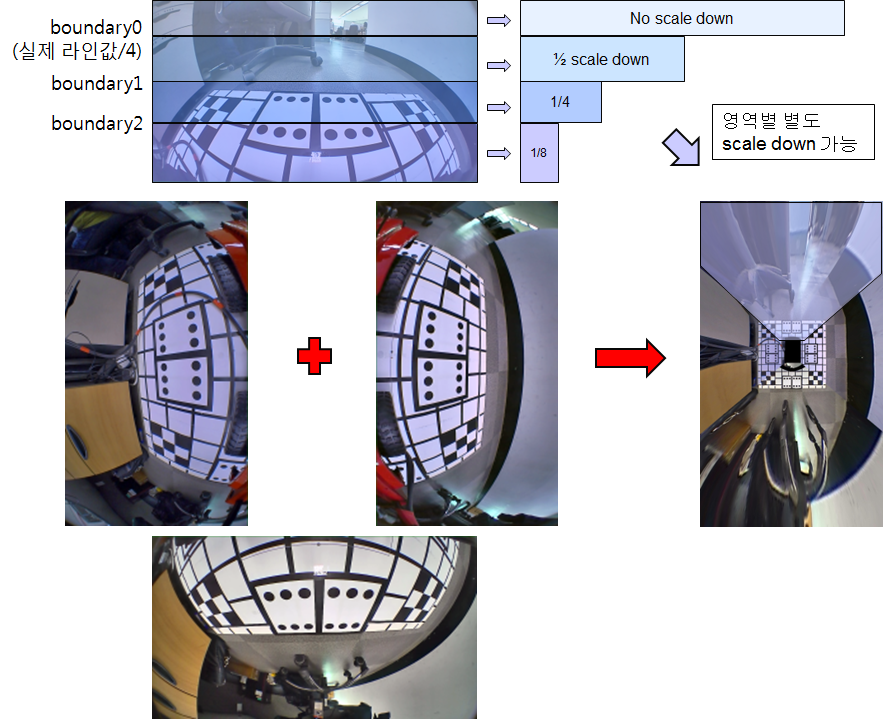
### SVMAPI\_setHorizontalScale

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setHorizontalScale(uint8 IN viewMode, uint8 IN channel,  SVM\_BOUNDARY\_T IN chBoundary); |

[Description]

각 입력 채널에 대하여 세로로 4부분으로 나누어 각 부분에 대해서 가로로 scale down을 적용할 값을 설정한다. 이 설정으로 band width을 줄여준다.



[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| channel | Channel 번호 (SVM\_CHANNEL\_E 참조) |
| chBoundary | Scale down을 할 4영역의 설정 값 (SVM\_BOUNDARY\_T 참조) |

[Return]

Error code

### SVMAPI\_setHoldTime

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setHoldTime(uint16 IN noScale, uint16 IN oneOverTwoScale,  uint16 IN oneOverFourScale, uint16 IN oneOverEightScale); |

[Description]

가로 scale down에 대한 hold time을 설정한다.

입력 채널의 write request를 일정 간격만큼 지연시켜 요청하도록 하여 read 동작의 우선권을 보장한다. 이러한 request의 한 라인당 요청 회수는 scale down 정도에 따라 다르게 되므로 scale down 정도에 따라 그 간격을 별도로 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| noScale | Scale down을 하지 않았을 때의 hold time |
| oneOverTwoScale | 1/2 Scale down을 하지 않았을 때의 hold time |
| oneOverFourScale | 1/4 Scale down을 하지 않았을 때의 hold time |
| oneOverEightScale | 1/8 Scale down을 하지 않았을 때의 hold time |

[Return]

### SVMAPI\_setVerticalHalfScale

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setVerticalHalfScale(uint8 IN viewMode, uint8 IN channel, bool IN isEnable); |

[Description]

각 입력 채널에 대하여 세로로 1/2 scale down을 설정한다.

이 설정으로 band width을 줄여준다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| channel | Channel 번호 (SVM\_CHANNEL\_E 참조) |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setInputBurstLength

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setInputBurstLength(uint8 IN burstLength); |

[Description]

입력 이미지의 write에 대한 burst length를 설정한다.

입력 이미지를 system memory에 write하는 방식은 한 번의 request로 한 라인을 모두 write하는 것이 아니라 burst ctrl 값에서 설정된 단위로 나누어서 여러 번 request하도록 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| burstLength | eSVM\_BURST\_LENGTH\_16 ~ eSVM\_BURST\_LENGTH\_128  (SVM\_BURST\_LENGTH\_E 참조) |

[Return]

Error code

### SVMAPI\_setInputTestModeEnable

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setInputTestModeEnable(bool IN isEnable); |

[Description]

입력 채널의 데이터를 카메라가 아닌 메모리에서 읽어서 사용하도록 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | 활성 또는 비활성 |

[Return]

### SVMAPI\_setInputWrappingInitEnable

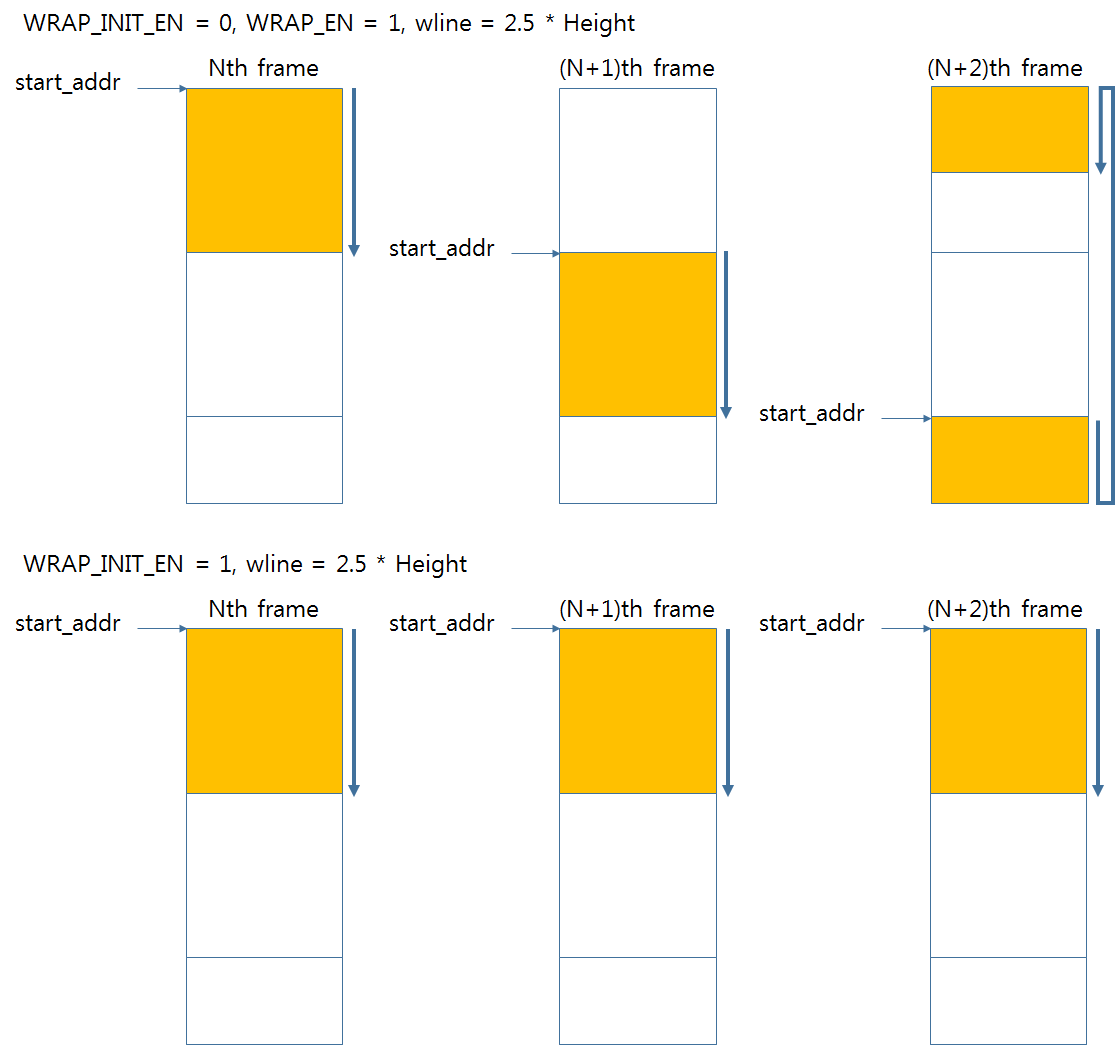
[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setInputWrappingInitEnable(bool IN isEnable); |

[Description]

SVMAPI\_initialize()의 inputFrameCount를 여러 frame값으로 설정하였어도 입력 채널의 쓸 때 page

flipping을 사용하지 않도록 한다.



[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | 활성 또는 비활성 |

[Return]

### SVMAPI\_setInputEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setInputEnable(bool IN isFrontEnable, bool IN isLeftEnable, bool IN isRightEnable,  bool IN isBackEnable); |

[Description]

각 채널의 입력을 사용할 것인지 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isFrontEnable | Front의 활성 또는 비활성 |
| isLeftEnable | Left의 활성 또는 비활성 |
| isRightEnable | Right의 활성 또는 비활성 |
| isBackEnable | Back의 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setSync

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setSync(uint8 IN pipLineDelay, uint16 IN outLineDelay); |

[Description]

출력 vsync의 생성은 입력 vsync negedge를 기준으로 OUT\_LINE\_DELAY 만큼의 line delay를 가지고 생성되고 내부 동작 시작에 필요한 pre\_vsync 생성을 위해 PIP\_LINE\_DELAY 신호가 사용된다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| pipLineDelay | 내부 동작 시작에 필요한 pre\_vsync 동작을 위한 line delay |
| outLineDelay | 출력 vsync를 위한 line delay |

[Return]

### SVMAPI\_setBGColor

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setBGColor(uint8 IN y, uint8 IN cb, uint8 IN cr); |

[Description]

Back ground color를 설정한다.

[Parameter]

[Return]

### SVMAPI\_setMaskColor

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setMaskColor(uint8 IN y, uint8 IN cbcr); |

[Description]

front와 back 이미지에 의해 출력을 생성하는 warping core에서는 left와 right에 해당하는 부분의 invalid LUT에 대해서는 MASK\_COLOR로 대체, 출력하도록 하여 이후 blend block에서 MASK\_COLOR와 같은 색인 경우 BG\_COLOR로 대체하여 출력되도록 한다. 그리고 출력 활성 영역인 section 영역 외의 영역에 대해서도 BG\_COLOR가 출력된다. MASK\_COLOR는 Cb = Cr 로 설정하여야 한다.

[Parameter]

[Return]

### SVMAPI\_setColorCoef

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setColorCoef(pSVM\_COLORCOEF\_T IN coefColor); |

[Description]

RGB와 YCbCr 간의 변환에 사용될 계수 값을 설정한다. 입력 카메라와의 변환 공식을 맞추기 위해서 사용된다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| coefColor | SVM\_COLORCOEF\_T 참조 |

[Return]

### SVMAPI\_setCar

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setCar(uint8 IN viewMode, SVM\_RECT\_T IN rect, uint8 IN maskColorY,  uint8 IN maskColorCb, uint8 IN maskColorCr); |

[Description]

자동차 이미지의 위치 및 make color를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| rect | 사각 영역 (SVM\_RECT\_T 참조) |
| maskColorY | Y mask color |
| maskColorCb | Cb mask color |
| maskColorCr | Cr mask color |

[Return]

Error code

### SVMAPI\_setCarReadBurstLength

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setCarReadBurstLength(uint8 IN burstLength); |

[Description]

자동차 이미지를 메모리에서 읽을 때의 burst length을 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| burstLength | 0 ~ 127 |

[Return]

Error code

### SVMAPI\_updateCar

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_updateCar(uint8 IN viewMode, uint16\* IN carImage, uint32 IN size); |

[Description]

자동차 이미지를 업데이트한다.

자동차 이미지를 사용하기 위해서는 SVMAPI\_setViewMode()로 view mode를 활성화 하기 전에

업데이트를 해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*carImage | 자동차 이미지의 데이터의 포인터 |
| size | 자동차 이미지의 사이즈 |

[Return]

Error code

### SVMAPI\_setCarAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setCarAddr(uint8 IN viewMode, uint32 IN addr); |

[Description]

자동차 이미지의 메모리 주소를 설정한다. 자동차 이미지의 메모리 주소는 SVMAPI\_createViewMode() 수행 시 자동으로 만들어 지므로 강제적으로 변경이 필요할 때만 사용해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| addr | 메모리 주소 |

[Return]

Error code

### SVMAPI\_getCarAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_getCarAddr(uint8 IN viewMode, uint32 OUT \*addr); |

[Description]

자동차 이미지의 메모리 주소를 얻는다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*addr | 메모리 주소 |

[Return]

Error code

### SVMAPI\_setCarEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setCarEnable(uint8 IN viewMode, bool IN isEnable); |

[Description]

자동차 이미지의 활성화를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setEdgeEnhancementMode

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setEdgeEnhancementMode(uint8 IN mode); |

[Description]

Edge enhancement mode를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| mdoe | SVM\_EDGE\_MODE\_E 참조 |

[Return]

Error code

### SVMAPI\_setEdgeEnhancementFixedGain

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setEdgeEnhancementFixedGain(float IN gain); |

[Description]

Edge enhancement의 eSVM\_EDGE\_MODE\_FIXED\_GAIN mode에서 사용될 gain 값을 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| gain | 고정된 gain 값 |

[Return]

Error code

### SVMAPI\_setEdgeEnhancementEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setEdgeEnhancementEnable(bool IN isEnable); |

[Description]

Edge enhancement의 활성화를 설정한다.

출력 영상의 경우 외곽으로 갈수록 많이 펴짐에 따라 흐리게 생성되므로 edge enhancement 기능을 통해 보안 할 수 있다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setDynamicblendingCoef

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setDynamicblendingCoef(uint8 coef1, uint8 coef3, uint8 coef6, uint8 coef8); |

[Description]

Dynamic blending을 위한 계수를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| coef1 | 1번째 영역의 계수 |
| coef3 | 3번째 영역의 계수 |
| coef6 | 6번째 영역의 계수 |
| coef8 | 8번째 영역의 계수 |

[Return]

Error code

### SVMAPI\_setDynamicblendingEnable

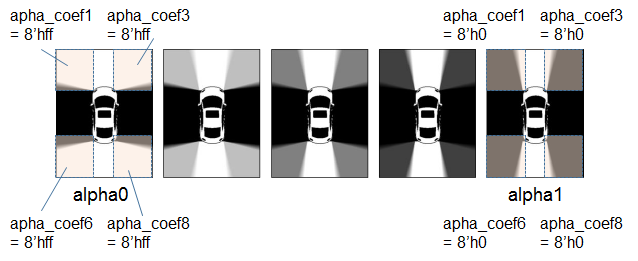
[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setDynamicblendingEnable(bool IN isEnable); |

[Description]

Dynamic blending의 활성화를 설정한다.

각 카메라 간에 blending 되는 영역에서의 사각 문제를 해결하기 위해 dynamic blending 기능을 지원하며 이를 위해 카메라 간에 섞이는 4군데의 영역에 대해 두 개의 alpha 값 중에 interpolation하는 정도를 조정할 수 있다. 이를 위한 control registers이며 각 의미는 아래와 같다. 값이 0xff이면 alpha0, 0x0이면 alpha1을 사용하며 중간 값의 경우 interpolation된 값을 사용하게 된다.



[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setMorphingSpeed

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setMorphingSpeed(uint8 IN speed); |

[Description]

Morphing 수행 시 속도를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| speed | Mophing speed (0~ 7) |

[Return]

Error code

### SVMAPI\_updateBCCoef

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_updateBCCoef(pSVM\_BCCOEF\_T IN coefBC); |

[Description]

Brightness control을 위한 계수들을 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| coefBC | SVM\_BCCOEF\_T 참조 |

[Return]

Error code

### SVMAPI\_updateBCRatio

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_updateBCRatio(uint8 IN ratio); |

[Description]

Brightness control을 위한 C Value를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| ratio | C value |

[Return]

### SVMAPI\_getPageNum4Lut

[Syntax]

|  |
| --- |
| uint8 APIENTRY SVMAPI\_getPageNum4Lut(uint8 IN viewMode, uint8 IN outputPartNum); |

[Description]

SVM 동작 중 Front/Back과 Left/Right LUT를 원활하게 변환하기 위해 각각의 LUT에 대하여 page flipping(page0 or page1)을 지원한다.

이 함수는 viewMode의 Front/Back 또는 Left/Right LUT의 page 번호를 얻을 수 있다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| outputPartNum | Front/Back or Left/Right (SVM\_OUTPUT\_PART\_E 참조) |

[Return]

Page number 또는 error시 eSVM\_PAGEFLIPPING\_NUMBER\_MAX (SVM\_PAGEFLIPPING\_NUMBER\_E 참조)

### SVMAPI\_updateFBLUT

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_updateFBLUT(uint8 IN viewMode, uint32\* IN lutBin, uint32 IN size); |

[Description]

Front/Back LUT를 업데이트 한다. 업데이트시 Page flipping을 수행한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*lutBin | LUT의 배얼 |
| size | LUT의 byte 사이즈 |

[Return]

Error code

### SVMAPI\_updateLRLUT

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_updateLRLUT(uint8 IN viewMode, uint32\* IN lutBin, uint32 IN size); |

[Description]

Left/Right LUT를 업데이트 한다. 업데이트시 Page flipping을 수행한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*lutBin | LUT의 배얼 |
| size | LUT의 byte 사이즈 |

[Return]

Error code

### SVMAPI\_updateBCLUT

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_updateBCLUT(uint8 IN viewMode, uint32\* IN lutBin, uint32 IN size); |

[Description]

Brightness control LUT를 업데이트 한다. BC LUT의 경우 고정된 LUT를 사용하므로 Page flipping은 지원하지 않는다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*lutBin | LUT의 배얼 |
| Size | LUT의 byte 사이즈 |

[Return]

Error code

### SVMAPI\_setFBLUTAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setFBLUTAddr(uint8 IN viewMode, uint32 IN pageflipping0Addr,  uint32 IN pageflipping1Addr); |

[Description]

Front/Back LUT의 메모리 주소를 설정한다. 메모리 주소는 page flipping을 위해 두 개의 메모리 주소가 필요하다. LUT 메모리 주소는 SVMAPI\_createViewMode() 수행 시 자동으로 만들어 지므로 강제적으로 변경이 필요할 때만 사용해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| pageflipping0Addr | Page0의 메모리 주소 |
| Pageflipping1Addr | Page1의 메모리 주소 |

[Return]

Error code

### SVMAPI\_setLRLUTAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setLRLUTAddr(uint8 IN viewMode, uint32 IN pageflipping0Addr,  uint32 IN pageflipping1Addr); |

[Description]

Left/Right LUT의 메모리 주소를 설정한다. 메모리 주소는 page flipping을 위해 두 개의 메모리 주소가 필요하다. LUT 메모리 주소는 SVMAPI\_createViewMode() 수행 시 자동으로 만들어 지므로 강제적으로 변경이 필요할 때만 사용해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| pageflipping0Addr | Page0의 메모리 주소 |
| Pageflipping1Addr | Page1의 메모리 주소 |

[Return]

Error code

### SVMAPI\_setBCLUTAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setBCLUTAddr(uint8 IN viewMode, uint32 IN addr); |

[Description]

Brightness control LUT의 메모리 주소를 설정한다. LUT 메모리 주소는 SVMAPI\_createViewMode() 수행 시 자동으로 만들어 지므로 강제적으로 변경이 필요할 때만 사용해야 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| addr | 메모리 주소 |

[Return]

Error code

### SVMAPI\_getFBLUTAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_getFBLUTAddr(uint8 IN viewMode, uint32 OUT \*pageflipping0Addr,  uint32 OUT \*pageflipping1Addr); |

[Description]

Front/Back LUT의 메모리 주소를 얻는다. 메모리 주소는 page flipping을 위해 두 개의 메모리 주소가 있다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*pageflipping0Addr | Page0의 메모리 주소 |
| \*Pageflipping1Addr | Page1의 메모리 주소 |

[Return]

Error code

### SVMAPI\_getLRLUTAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_getLRLUTAddr(uint8 IN viewMode, uint32 OUT \*pageflipping0Addr,  uint32 OUT \*pageflipping1Addr); |

[Description]

Left/Right LUT의 메모리 주소를 얻는다. 메모리 주소는 page flipping을 위해 두 개의 메모리 주소가 있다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*pageflipping0Addr | Page0의 메모리 주소 |
| \*Pageflipping1Addr | Page1의 메모리 주소 |

[Return]

Error code

### SVMAPI\_getBCLUTAddr

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_getBCLUTAddr(uint8 IN viewMode, uint32 OUT \*addr); |

[Description]

Brightness control LUT의 메모리 주소를 얻는다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| \*addr | 메모리 주소 |

[Return]

Error code

### SVMAPI\_getRGBStatistics

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_getRGBStatistics(pSVM\_RGB\_STATISTICS\_T OUT stats); |

[Description]

Auto brightness control을 dnlgo Front/Back이미지와 Left/Right 이미지간의 서로 섞이게 되는 영역의 RGB색의 통계 값을 구한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| \*stats | RGB색 통계 값을 저장할 구조체의 포인터(SVM\_RGB\_STATISTICS\_T 참조) |

[Return]

### SVMAPI\_setSVMEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setSVMEnable(bool IN isEnable); |

[Description]

SVM을 동작시킨다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setBlank

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setBlank(uint16 IN hBlank, uint16 IN vBlank); |

[Description]

출력의 frame rate 생성을 위한 horizontal blank, vertical blank를 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| hBlank | Horizontal blank |
| vBlank | Vertical blank |

[Return]

### SVMAPI\_setSection

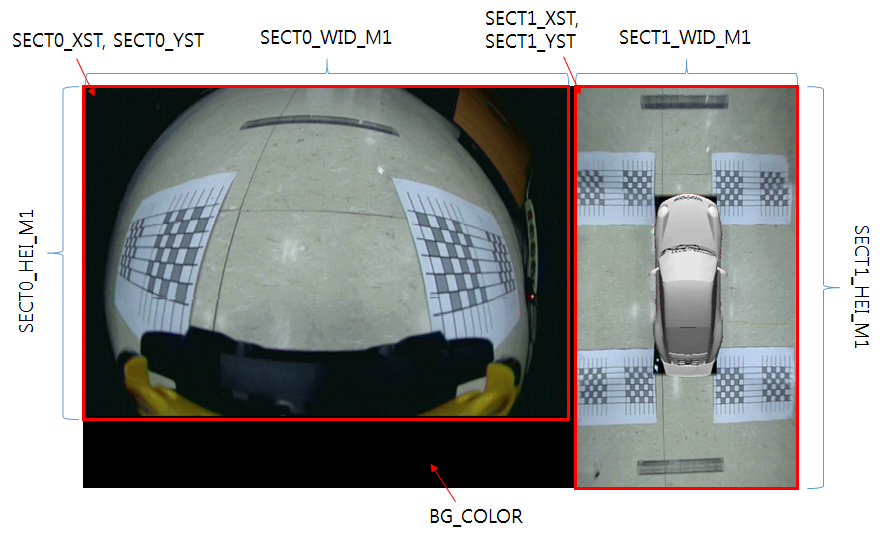
[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setSection(uint8 IN viewMode, uint8 IN sectionNumber,  SVM\_RECT\_T IN sectionRect); |

[Description]

Section 영역을 설정한다. Section은 사각 영역을 설정하여 그 부분에 대해서만 출력하도록 한다.

2개의 section을 지원한다.



[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| sectionNumber | Section 번호 (SVM\_SECTION\_NUMBER\_E 참조) |
| sectionRect | Section 사각 영역 (SVM\_RECT\_T 참조) |

[Return]

Error code

### SVMAPI\_setSectionEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setSectionEnable(uint8 IN viewMode, uint8 IN sectionNumber, bool IN isEnable); |

[Description]

각 Section 영역의 동작을 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| viewMode | View mode handler의 번호 |
| sectionNumber | Section 번호 (SVM\_SECTION\_NUMBER\_E 참조) |
| isEnable | 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setOutputBurstLength

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setOutputBurstLength(uint8 IN burstLength); |

[Description]

출력 이미지의 read에 대한 burst length를 설정한다.

출력 이미지를 system memory에 read하는 방식은 한 번의 request로 한 라인을 모두 read하는 것이 아니라 burst ctrl 값에서 설정된 단위로 나누어서 여러 번 request하도록 한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| burstLength | eSVM\_BURST\_LENGTH\_8 ~ eSVM\_BURST\_LENGTH\_128  (SVM\_BURST\_LENGTH\_E 참조) |

[Return]

Error code

### SVMAPI\_setInvalidLUTEnable

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setInvalidLUTEnable(uint8 IN isEnable); |

[Description]

Brightness control LUT의 Alpha값을 이용하여 불필요한 LUT 영역을 Invalid LUT 영역으로 변경하여 사용하도록 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isEnable | 활성 또는 비활성 |

[Return]

### SVMAPI\_setOutputEnable

[Syntax]

|  |
| --- |
| int32 APIENTRY SVMAPI\_setOutputEnable(bool IN isFBEnable, bool IN isLREnable); |

[Description]

Front/Back 또는 Left/Right의 출력을 각각 사용할 것인지 설정한다.

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| isFBEnable | Front/Back의 활성 또는 비활성 |
| isLREnable | Left/Right의 활성 또는 비활성 |

[Return]

Error code

### SVMAPI\_setCBOutputVsync

[Syntax]

|  |
| --- |
| void APIENTRY SVMAPI\_setCBOutputVsync(SVM\_CALLBACK IN callback); |

[Description]

출력 VSYMC 인터럽트의 callback 함수를 설정한다.

typedef void (\*SVM\_CALLBACK) (void);

[Parameter]

|  |  |
| --- | --- |
| Member | Description |
| callback | Callback 함수 포인터 또는 NULL |

[Return]