MI DIVP API

Version 2.04

REVISION HISTORY

Revision No.	Description	Date
2.03	• Initial release	04/12/2018
2.04	• Added new API: • MI_DIVP_StretchBuf	12/16/2019

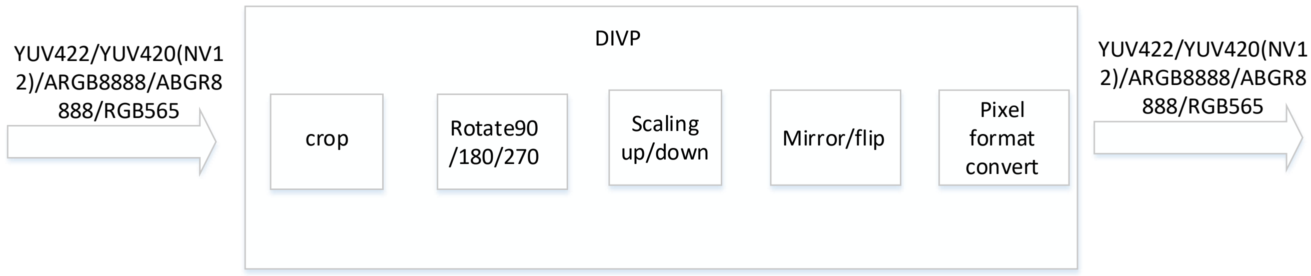
TABLE OF CONTENTS

1. SUMMARY

1.1. Module Description

DIVP supports pre-processing of an input image, such as cropping, image pixel format conversion, rotation, mirroring, etc., and separate scaling processing for each channel, and finally outputs a variety of images with different resolutions.
The specific image processing functions supported by DIVP include image cropping, image pixel format conversion, image rotation, image mirror operation, and image stretching and scaling.

1.2. Flow Chart


YUV422/YUV420(NV12)/ARGB8888/ABGR8888/RGB565
DIVP
crop
Rotate90/180/270
Scaling up/down
Mirror/flip
Pixel format convert
YUV422/YUV420(NV12)/ARGB8888/ABGR8888/RGB565

Note:
Rotation is NOT supported by chip series including 325, 327, 621, 623, 201, 202, 335, and 337 series.

1.3. Keyword

- Crop
  Image cropping

- Rotate
  Image rotation

- Scaling up/down
  Image stretching and scaling

- Mirror/Flip
  Image mirror operation

- Pixel Format Conversion
  Image pixel format conversion

2. API LIST

The MI DIVP module provides the following APIs:

Name of API	**Function**
MI_DIVP_CreateChn	Create a DIVP channel.
MI_DIVP_DestroyChn	Destroy a DIVP channel.
MI_DIVP_SetChnAttr	Set the DIVP channel attribute.
MI_DIVP_GetChnAttr	Get the DIVP channel attribute.
MI_DIVP_StartChn	Start a channel.
MI_DIVP_StopChn	Stop a channel.
MI_DIVP_SetOutputPortAttr	Set the DIVP output port attribute.
MI_DIVP_GetOutputPortAttr	Get the DIVP output port attribute.
MI_DIVP_RefreshChn	Refresh a channel in pause state.
MI_DIVP_StretchBuf	Stretch or crop image in specified memory to target memory

2.1. MI_DIVP_CreateChn

➢ Function

Create a new DIVP channel.

➢ Syntax

MI_S32 MI_DIVP_CreateChn (
MI_DIVP_CHN DivpChn,
MI_DIVP_ChnAttr_t* pstAttr);

➢ Parameter

Parameter Name	Description	Input/Output
DivpChn	Return the ID of the created DIVP channel. DIVP supports maximum 6 channels, parameter range: [0, 5].	Input
pstAttr	DIVP channel attribute pointer, used to set the attribute of the created channel.	Input

➢ Return Value

- MI_SUCCESS: DIVP channel created successfully
- MI_DIVP_ERR_FAILED: Attempt to create DIVP channel failed
- MI_DIVP_ERR_NO_RESOUCE: System resource not enough; attempt to create DIVP channel failed

➢ Requirement

- Header: mi_divp.h, mi_divp_datatype.h
- Library: libmi_divp.so

➢ Note

The DIVP module supports maximum 6 channels. When the total number of channels created exceeds 6 or the system resource is insufficient, the channel creation task will fail.

➢ Example

```
MI_DIVP_CHN u32ChnId = 0;
MI_DIVP_ChnAttr_t stDivpChnAttr;
MI_DIVP_OutputPortAttr_t stDivpOutputPortAttr;

memset(&stDivpChnAttr,0,sizeof(MI_DIVP_ChnAttr_t));
memset(&stDivpOutputPortAttr,0,sizeof(MI_DIVP_OutputPortAttr_t));

stDivpChnAttr.bHorMirror =  false;
stDivpChnAttr.bVerMirror = false;
stDivpChnAttr.eDiType = E_MI_DIVP_DI_TYPE_OFF;
stDivpChnAttr.eRotateType = E_MI_SYS_ROTATE_90;
stDivpChnAttr.eTnrLevel = E_MI_DIVP_TNR_LEVEL_OFF;
stDivpChnAttr.stCropRect.u16X = 0;
stDivpChnAttr.stCropRect.u16Y = 0;
stDivpChnAttr.stCropRect.u16Width = 1280;
stDivpChnAttr.stCropRect.u16Height = 720;
stDivpChnAttr.u32MaxWidth = 1920;
stDivpChnAttr.u32MaxHeight = 1080;
MI_DIVP_CreateChn(u32ChnId,&stDivpChnAttr);

MI_DIVP_GetChnAttr(u32ChnId,&stDivpChnAttr);
stDivpChnAttr.stCropRect.u16X = 0;
stDivpChnAttr.stCropRect.u16Y = 0;
stDivpChnAttr.stCropRect.u16Width = 1920;
stDivpChnAttr.stCropRect.u16Height = 1080;
MI_DIVP_SetChnAttr(u32ChnId,&stDivpChnAttr);

stDivpOutputPortAttr.eCompMode = E_MI_SYS_COMPRESS_MODE_NONE;
stDivpOutputPortAttr.ePixelFormat = E_MI_SYS_PIXEL_FRAME_YUV422_YUYV;
stDivpOutputPortAttr.u32Width = 1920;
stDivpOutputPortAttr.u32Height = 1080;
MI_DIVP_SetOutputPortAttr(u32ChnId,&stDivpOutputPortAttr);

MI_DIVP_StartChn(u32ChnId);

//exit flow
MI_DIVP_StopChn(u32ChnId);
MI_DIVP_DestroyChn(u32ChnId);
```

- Related API

  MI_DIVP_DestroyChn

2.2. MI_DIVP_DestroyChn

- Function

  Destroy a DIVP channel.

- Syntax

  MI_S32 MI_DIVP_DestroyChn(MI_DIVP_CHN DivpChn);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	ID of the destroyed DVIP channel	Input

- Return Value
  - MI_SUCCESS: DVIP channel destroyed successfully
  - MI_DIVP_ERR_FAILED: Attempt to destroy DVIP channel failed

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  N/A.

- Related API

  MI_DIVP_CreateChn

2.3. MI_DIVP_SetChnAttr

- Function

  Set the DIVP channel attribute.

- Syntax

  MI_S32 MI_DIVP_SetChnAttr(
  MI_DIVP_CHN DivpChn,
  MI_DIVP_ChnAttr_t* pstAttr);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	DIVP channel ID.	Input
pstAttr	Pointer to the set DVIP channel attribute structure.	Input

- Return Value
  - MI_SUCCESS: DIVP channel attribute set successfully
  - MI_DIVP_ERR_FAILED: Attempt to set DIVP channel attribute failed
  - MI_ERR_INVALID_PARAMETER: Invalid parameter used. Please refer to MI_DIVP_ChnAttr_t

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  The maximum width and the maximum height of input image in channel attribute structure member are static attributes which cannot be modified once the channel is created.

- Example

  N/A.

- Related API

  MI_DIVP_GetChnAttr

2.4. MI_DIVP_GetChnAttr

- Function

  Get the DIVP channel attribute.

- Syntax

  MI_S32 MI_DIVP_GetChnAttr(
  MI_DIVP_CHN DivpChn,
  MI_DIVP_ChnAttr_t* pstAttr);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	DIVP channel ID.	Input
pstAttr	Pointer to the returned DVIP channel attribute structure.	Output

- Return Value
  - MI_SUCCESS: DIVP channel attribute gotten successfully
  - MI_DIVP_ERR_FAILED: Attempt to get DIVP channel attribute failed

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  Initial value will be returned with the first successful attempt to get the DIVP channel attribute.

- Example

  N/A.

- Related API

  MI_DIVP_SetChnAttr

2.5. MI_DIVP_StartChn

- Function

  Start a channel.

- Syntax

  MI_S32 MI_DIVP_StartChn(MI_DIVP_CHN DivpChn);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	DIVP channel ID.	Input

- Return Value
  - MI_SUCCESS: DIVP channel started successfully
  - MI_DIVP_ERR_FAILED: Attempt to start DIVP channel failed

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  Channels to be started must be created beforehand and not destroyed.

- Example

  N/A.

- Related API

  MI_DIVP_StopChn

2.6. MI_DIVP_StopChn

- Function

  Stop a channel.

- Syntax

  MI_S32 MI_DIVP_StopChn(MI_DIVP_CHN DivpChn);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	DIVP channel ID.	Input

- Return Value
  - MI_SUCCESS: DIVP channel stopped successfully
  - MI_DIVP_ERR_FAILED: Attempt to stop DIVP channel failed

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  Repeated execution of this stop function will return MI_SUCCESS.

- Example

  N/A.

- Related API

  MI_DIVP_StartChn

2.7. MI_DIVP_SetOutputPortAttr

- Function

  Set the DIVP channel output port attribute.

- Syntax

  MI_S32 MI_DIVP_SetOutputPortAttr (
  MI_DIVP_CHN DivpChn,
  MI_DIVP_OutputPortAttr_t * pstOutputPortAttr);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	DIVP channel ID.	Input
pstOutputPortAttr	Output port attribute pointer.	Input

- Return Value
  - MI_SUCCESS: DIVP channel output port attribute set successfully
  - MI_DIVP_ERR_FAILED: Attempt to set DIVP channel output port attribute failed

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  N/A.

- Example

  N/A.

- Related API

  MI_DIVP_GetOutputPortAttr

2.8. MI_DIVP_GetOutputPortAttr

- Function

  Get the DIVP output port attribute.

- Syntax

  MI_S32 MI_DIVP_GetOutputPortAttr(
  MI_DIVP_CHN DivpChn,
  MI_DIVP_OutputPortAttr_t * pstOutputPortAttr);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	DIVP channel ID.	Input
pstOutputPortAttr	Output port attribute pointer.	Output

- Return Value
  - MI_SUCCESS: DIVP channel output port attribute gotten successfully
  - MI_DIVP_ERR_FAILED: Attempt to get DIVP channel output port attribute failed

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  N/A.

- Example

  N/A.

- Related API

  MI_DIVP_SetOutputPortAttr

2.9. MI_DIVP_RefreshChn

- Function

  Refresh a DIVP channel in pause state.

- Syntax

  MI_S32 MI_DIVP_RefreshChn (
  MI_DIVP_CHN DivpChn);

- Parameter

Parameter Name	Description	Input/Output
DivpChn	ID of refreshed channel.	Input

- Return Value
  - MI_SUCCESS: DIVP channel refreshed successfully.
  - MI_DIVP_ERR_NO_CONTENT: No content in the channel being refreshed.

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  This interface is used for refreshing a channel under pause state.

- Example

  In the scenario where DIVP is bound to VDEC, this interface is called. DIVP can retain the last frame of the picture after VDEC stops streaming, and repeat the processing and then send it to the back end. DIVP will not process the new picture until VDEC restarts streaming.

- Related API

  N/A.

2.10. MI_DIVP_StretchBuf

- Function

  Stretch or crop image in specified memory to target memory.

- Syntax

  MI_S32 MI_DIVP_StretchBuf(MI_DIVP_DirectBuf_t *pstSrcBuf, MI_SYS_WindowRect_t *pstSrcCrop, MI_DIVP_DirectBuf_t *pstDstBuf);

- Parameter

Parameter Name	Description	Input/Output
pstSrcBuf	Information pointer of specified buff used to store source image. Cannot be NULL.	Input
pstSrcCrop	Pointer to image crop attribute, If you don't need to crop the image, pass NULL.	Input
pstDstBuf	Information pointer of target buff used to store processed image. Cannot be NULL.	Input

- Return Value
  - MI_SUCCESS: Image stretched successfully.
  - MI_DIVP_ERR_FAILED: Image stretching failed.

- Requirement
  - Header: mi_divp.h, mi_divp_datatype.h
  - Library: libmi_divp.so

- Note

  Supports only YUV420SP or ARGB8888 image pixel format

- Example

```
#define SRC_WIDTH 1280
#define SRC_HEIGHT 720
#define SRC_BUFF_STRIDE (ALIGN_UP(SRC_WIDTH,16))
#define SRC_BUFF_SIZE (SRC_BUFF_STRIDE*SRC_HEIGHT*3/2)

#define DST_WIDTH 640
#define DST_HEIGHT 480
```

```
#define DST_BUFF_STRIDE (ALIGN_UP(DST_WIDTH,16))
#define DST_BUFF_SIZE (DST_BUFF_STRIDE*DST_HEIGHT*3/2)

#define CROP_X 200
#define CROP_Y 100
#define CROP_W 68
#define CROP_H 48

int main(void)
{
    MI_PHY phySrcBufAddr = 0;
    MI_PHY phyDstBufAddr = 0;
    MI_DIVP_DirectBuf_t stSrcBuf;
    MI_DIVP_DirectBuf_t stDstBuf;
    MI_SYS_WindowRect_t stSrcCrop;

    MI_SYS_MMA_Alloc(NULL, SRC_BUFF_SIZE, &phySrcBufAddr);
    MI_SYS_MMA_Alloc(NULL, DST_BUFF_SIZE, &phyDstBufAddr);

    stSrcBuf.ePixelFormat = E_MI_SYS_PIXEL_FRAME_YUV_SEMIPLANAR_420;
    stSrcBuf.u32Width = SRC_WIDTH;
    stSrcBuf.u32Height = SRC_HEIGHT;
    stSrcBuf.u32Stride[0] = SRC_BUFF_STRIDE;
    stSrcBuf.u32Stride[1] = SRC_BUFF_STRIDE;
    stSrcBuf.phyAddr[0] = phySrcBufAddr;
    stSrcBuf.phyAddr[1] = stSrcBuf.phyAddr[0] + SRC_BUFF_STRIDE*SRC_HEIGHT;
    stDstBuf.ePixelFormat = E_MI_SYS_PIXEL_FRAME_YUV_SEMIPLANAR_420;
    stDstBuf.u32Width = DST_WIDTH;
    stDstBuf.u32Height = DST_HEIGHT;
    stDstBuf.u32Stride[0] = DST_BUFF_STRIDE;
    stDstBuf.u32Stride[1] = DST_BUFF_STRIDE;
    stDstBuf.phyAddr[0] = phyDstBufAddr;
    stDstBuf.phyAddr[1] = stDstBuf.phyAddr[0] + DST_BUFF_STRIDE*DST_HEIGHT;

    stSrcCrop.u16X = CROP_X;
    stSrcCrop.u16Y = CROP_Y;
    stSrcCrop.u16Width = CROP_W;
    stSrcCrop.u16Height = CROP_H;

    if(FillSrcBuf("./1280x720_yuv420.yuv", &stSrcBuf))
        return NULL;
    if(MI_SUCCESS == MI_DIVP_StretchBuf(&stSrcBuf, &stSrcCrop, &stDstBuf))
    {
         if(DumpDstBuf(&stDstBuf))
             return NULL;
    }

    MI_SYS_MMA_Free(phySrcBufAddr);
    MI_SYS_MMA_Free(phyDstBufAddr);
}

//fill src buff with yuv420/argb8888 image data
static int FillSrcBuf(const char* FilePath,MI_DIVP_DirectBuf_t *pstDirectSrcBuf)
{
   int ret = 0;
   FILE *fp;
   void *pVirSrcBufAddr = NULL;
   int LineIdx = 0;
```

```
    int ReadSize = 0;

    fp = fopen(FilePath,"r");
    if(!fp)
    {
        divp_ut_dbg("open file[%s] failed\n",FilePath);
        ret = -1;
        goto EXIT;
    }

    MI_SYS_Mmap(pstDirectSrcBuf->phyAddr[0], SRC_BUFF_SIZE, &pVirSrcBufAddr, FALSE);
    if(!pVirSrcBufAddr)
    {
        divp_ut_dbg("mmap dst buff failed\n");
        ret = -1;
        goto EXIT;
    }

    for(LineIdx = 0; LineIdx < SRC_HEIGHT*3/2; LineIdx++)
    {
        ReadSize += fread(pVirSrcBufAddr+LineIdx*SRC_BUFF_STRIDE, 1, SRC_WIDTH, fp);
    }
    if(ReadSize < SRC_WIDTH*SRC_HEIGHT*3/2)
    {
        fseek(fp, 0, SEEK_SET);
        ReadSize = 0;
        for(LineIdx = 0; LineIdx < SRC_HEIGHT*3/2; LineIdx++)
        {
            ReadSize += fread(pVirSrcBufAddr+LineIdx*SRC_BUFF_STRIDE, 1, SRC_WIDTH,
fp);
        }
        if(ReadSize < SRC_WIDTH*SRC_HEIGHT*3/2)
        {
            divp_ut_dbg("read file failed, read size:%d\n",ReadSize);
            ret = -1;
            goto EXIT;
        }
    }

EXIT:
    if(fp)
        fclose(fp);
    if(pVirSrcBufAddr)
        MI_SYS_Munmap(pVirSrcBufAddr, SRC_BUFF_SIZE);

    return ret;
}

//image processing result is stored in dst buff
static int DumpDstBuf(MI_DIVP_DirectBuf_t *pstDirectDstBuf)
{
    int ret = 0;
    FILE *fp;
    void *pVirDstBufAddr = NULL;
    int LineIdx = 0;
    int WriteSize = 0;
    char outputfile[128];
    struct timeval timestamp;
    gettimeofday(&timestamp, 0);
    sprintf(outputfile,
"output_%dx%d_%d_%08d.yuv",pstDirectDstBuf->u32Width,pstDirectDstBuf->u32Height,(i
nt)timestamp.tv_sec,(int)timestamp.tv_usec);

    fp = fopen(outputfile,"w+");
    if(!fp)
    {
        divp_ut_dbg("open file[%s] failed\n",outputfile);
        ret = -1;
        goto EXIT;
    }

    MI_SYS_Mmap(pstDirectDstBuf->phyAddr[0], DST_BUFF_SIZE, &pVirDstBufAddr, FALSE);
    if(!pVirDstBufAddr)
    {
        divp_ut_dbg("mmap dst buff failed\n");
        ret = -1;
        goto EXIT;
    }

    for(LineIdx = 0; LineIdx < DST_HEIGHT*3/2; LineIdx++)
    {
        WriteSize += fwrite(pVirDstBufAddr+LineIdx*DST_BUFF_STRIDE, 1, DST_WIDTH, fp);
    }
    if(WriteSize < DST_WIDTH*DST_HEIGHT*3/2)
    {
        divp_ut_dbg("write file failed, write size:%d\n",WriteSize);
    }
    fflush(fp);
    sync();
    divp_ut_dbg("save stretch dst buff to[%s]\n",outputfile);

EXIT:
    if(fp)
        fclose(fp);
    if(pVirDstBufAddr)
        MI_SYS_Munmap(pVirDstBufAddr, DST_BUFF_SIZE);

    return 0;
}
```

➢ Related API

N/A.

3. DIVP DATA TYPE

The table below lists the data structure definitions of the related DIVP data types:

MI_DIVP_DiType_e	Defines DIVP deinterlace type.
MI_DIVP_TnrLevel_e	Defines DIVP TNR level.
MI_DIVP_OutputPortAttr_t	Defines DIVP output port attribute parameter.
MI_DIVP_ChnAttr_t	Defines DIVP channel attribute parameter.
MI_DIVP_CHN	DIVP channel ID.
MI_DIVP_DirectBuf_t	Memory information for stretching image

NOTE: This section covers only the most important data structures. For data types not listed here, please refer to mi_divp_datatype.h.

3.1. MI_DIVP_DiType_e

➢ Description

Defines DIVP deinterlace type.

➢ Definition

```
typedef enum
{
    E_MI_DIVP_DITYPE_OFF, //off
    E_MI_DIVP_DITYPE_2D, ///2.5D DI
    E_MI_DIVP_DITYPE_3D, ///3D DI
    E_MI_DIVP_DITYPE_NUM,
} MI_DIVP_DiType_e;
```

➢ Member

Member	Description
E_MI_DIVP_DITYPE_OFF	Disable deinterlace function on DIVP channel.
E_MI_DIVP_DITYPE_2D	Enable 2.5D deinterlace function on DIVP channel.
E_MI_DIVP_DITYPE_3D	Enable 3D deinterlace function on DIVP channel.
E_MI_DIVP_DITYPE_NUM	Number of deinterlace types on DIVP channel.

➢ Note

- TNR must be turned on when DI is enabled. MSR930 supports 3D DI only, and the TNR level must be set as E_MI_DIVP_TNRLEVEL_MIDDLE.
- 3D DI conflicts with rotation; the two functions cannot be used at the same time.
- Note that the following chips do not support the DI function:
  328Q/329D/326D
  325/325DE/327DE
  621/623/201/202
  336D/336Q/339G
  335/337DE

➢ Related Data Type and Interface

N/A.

3.2. MI_DIVP_TnrLevel_e

➢ Description

Defines DIVP TNR level.

➢ Definition

```
typedef enum
{
    E_MI_DIVP_TNRLEVEL_OFF,
    E_MI_DIVP_TNRLEVEL_LOW,
    E_MI_DIVP_TNRLEVEL_MIDDLE,
    E_MI_DIVP_TNRLEVEL_HIGH,
    E_MI_DIVP_TNRLEVEL_NUM,
} MI_DIVP_TnrLevel_e;
```

➢ Member

Member	Description
E_MI_DIVP_TNRLEVEL_OFF	Disable TNR on DIVP channel.
E_MI_DIVP_TNRLEVEL_LOW	Apply low level TNR on DIVP channel.
E_MI_DIVP_TNRLEVEL_MIDDLE	Apply middle level TNR on DIVP channel.
E_MI_DIVP_TNRLEVEL_HIGH	Apply high level TNR on DIVP channel.
E_MI_DIVP_TNRLEVEL_NUM	Number of TNR levels on DIVP channel.

➢ Note

- MSR930 supports TNR, but the TNR level cannot be modified.
- Note that the following chips do not support the TNR function:
  328Q/329D/326D
  325/325DE/327DE
  621/623/201/202
  336D/336Q/339G
  335/337DE

➢ Related Data Type and Interface

N/A.

3.3. MI_DIVP_OutputPortAttr_t

➢ Description

Defines DIVP output port attribute parameter.

➢ Definition

```
typedef struct MI_DIVP_OutputPortAttr_s
{
        MI_U32 u32Width;//output width
        MI_U32 u32Height;//output height
        MI_SYS_PixelFormat_e ePixelFormat;
        MI_SYS_CompressMode_e eCompMode;//compress mode
}MI_DIVP_OutputPortAttr_t;
```

➢ Member

Member	Description
u32Width	DIVP channel output screen width.
u32Height	DIVP channel output screen height.
ePixelFormat	DIVP channel output screen pixel format.
eCompMode	DIVP channel output image compression mode. DIVP channel can only output images not in compressed format.

➢ Note

The following table lists the different output attributes of each chip:

Output Attr / Chip Series	Output Pixel Format	Output Stride Alignment	Output Width Alignment	Output Height Alignment	Output Min Size	Output Max Size
MSR930	YUV422/YUV420(NV12)/ ARGB8888/ ABGR8888/ARGB1555/ MST420	32	2	2	128x64	4096x4096
328Q/329D/326D	YUV422	32	2	2	64x4	3840x3840
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
325/325DE/327DE	YUV422	32	2	2	64x4	2688x2688
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
621/623/201/202	YUV422	32	2	2	64x4	1920x1920
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
336D/336Q/339G	YUV422	32	2	2	Rotate: 16x2 No rotate: 32x4	3840x3840
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
335/337DE	YUV422	32	2	2	64x4	2688x2688
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				

- Related Data Type and Interface

  N/A.

3.4. MI_DIVP_ChnAttr_t

- Description

  Defines DIVP channel attribute parameter.

- Definition

```
typedef struct MI_DIVP_ChnAttr_s
{
        MI_U32 u32MaxWidth;//support max input width
        MI_U32 u32MaxHeight;//support max input height
        MI_DIVP_TnrLevel_e eTnrLevel;//TNR level
        MI_DIVP_DiType_e eDiType;//DI type
        MI_SYS_Rotate_e eRotateType;//rotate angle
        MI_SYS_WindowRect_t stCropRect;//crop information
        MI_BOOL bHorMirror;//horizontal mirror
         MI_BOOL bVerMirror;//vertical mirror
}MI_DIVP_ChnAttr_t;
```

- Member

Member	Description
u32MaxWidth	Maximum input width on DIVP channel.
u32MaxHeight	Maximum input height on DIVP channel.
eTnrLevel	TNR level on DIVP channel.
eDiType	Deinterlace type on DIVP channel.
eRotateType	Angle of screen rotation on DIVP channel.
stCropRect	Crop information on DIVP channel.
bHorMirror	Horizontal mirroring on DIVP channel.
bVerMirror	Vertical mirroring on DIVP channel.

➢ Note

Input Attr / Chip Series	Input Pixel Format	Input Stride Alignment	Input Width Alignment	Input Height Alignment	Input Min Size	Input Max Size
MSR930	YUV422/YUV420(NV12)/ ARGB8888/ ABGR8888/ARGB1555/ Tile Mode	32	YUV422:16 NV12:32	2	128x64	4096x4096
328Q/329D/326D	YUV422	32	2	2	64x4	3840x3840
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
325/325DE/327DE	YUV422	32	2	2	64x4	2688x2688
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
621/623/201/202	YUV422	32	2	2	64x4	1920x1920
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
336D/336Q/339G	YUV422	32	2	2	Rotate: 128x128 No rotate: 32x4	3840x3840
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				
335/337DE	YUV422	32	2	2	64x4	2688x2688
	YUV420(NV12)	16				
	ARGB8888/ABGR8888	64				
	RGB565	32				

➢ Related Data Type and Interface

N/A.

3.5. MI_DIVP_DirectBuf_t

➢ Description

Memory information for stretching image.

- Definition

```
typedef struct MI_DIVP_DirectBuf_s
{
    MI_SYS_PixelFormat_e ePixelFormat; //YUV420SP or ARGB888 only
    MI_U32 u32Width;
    MI_U32 u32Height;
    MI_U32 u32Stride[3];
    MI_PHY phyAddr[3];
}MI_DIVP_DirectBuf_t;
```

- Member

Member	Description
ePixelFormat	Pixel format of image
u32Width	Width of image
u32Height	Height of image
u32Stride	Number of bytes per line of an image
phyAddr	Start physical address of buffer

- Note

The pixel format of image will only be YUV420SP or ARGB8888.
The stride cannot be less than 64.

- Related Data Type and Interface

MI_DIVP_StretchBuf

4. DIVP ERROR CODES

The following table lists the DIVP API Error Codes.

Table 1: DIVP API Return Value

Error Code	Macro Definition	Description
0x0	MI_SUCCESS	Successful.
0xa00c2002	MI_DIVP_ERR_INVALID_CHNID	Invalid channel ID.
0xa00c2003	MI_DIVP_ERR_INVALID_PARAM	Invalid parameter.
0xa00c2006	MI_DIVP_ERR_NULL_PTR	Null pointer.
0xa00c201c	MI_DIVP_ERR_FAILED	DIVP operation failed
0xa00c2005	MI_DIVP_ERR_NO_RESOUCE	No resource available.
0xa00c201c	MI_DIVP_ERR_NO_CONTENT	No content in channel.