

SDK Function

UnPackImage

```
void UnPackImage(  
    void *hDev,  
    void *pInBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf  
);
```

RotateImage

```
void RotateImage(  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

ConvertBayerPackedToRgb / ConvertBayerToRgb

```
void ConvertBayerPackedToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

```
void ConvertBayerToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

ConvertBayerPackedBgr / ConvertBayerToBgr

```
void ConvertBayerPackedToBgr(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf  
);
```

```
void ConvertBayerToBgr(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

ConvertMonoPackedToRgb / ConvertMonoToRgb

```
void ConvertMonoPackedToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf  
);
```

```
void ConvertMonoToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

ConvertMonoPackedToBgr / ConvertMonoToBgr

```
void ConvertMonoPackedToBgr(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf  
);
```

```
void ConvertMonoToBgr(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

ConvertBayerRgbtoBgr

```
void ConvertRGB8toBGR8(  
    const uint8_t *Src,  
    uint8_t *pDst,  
    uint32_t cbSrc  
);
```

UnPackImage

Function:

```
void UnPackImage(  
    DEV_HANDLE hDev,  
    void *pInBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf  
)
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|--------------|--|
| [in] | hDev | - handle of an open device object |
| [in] | pInBuf | - input data buffer |
| [out] | pUnPackBuf | - output data buffer |
| [in] | pcbUnPackBuf | - size of the output buffer/actual output size |

Returns:

Remarks:

```
typedef struct _TLI_CF_UNPACK_IMAGE  
{  
    uint32_t pixel_type;  
    void* packed_image;  
    size_t packed_image_len;  
} TLI_CF_UNPACK_IMAGE;
```

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::GrabberThread()
```

```
{
```

```
:
```

```
PVOID pBase;
```

```
uint32_t cb = uint32_t(sizeof(pBase));
```

```
DSGetBufferInfo( m_hDS, hBuffer, BUFFER_INFO_BASE, NULL, &pBase, &cb );
```

```
:
```

```
if( m_pixelType == GVSP_PIX_MONO12_PACKED )
```

```
{
```

```
    BYTE *m_pUnpackedImage;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight*3/2;
```

```
    TLI_CF_UNPACK_IMAGE cf;
```

```
    cf.pixel_type = m_pixelType;
```

```
    cf.packed_image = PBYTE(pBase);
```

```
    cf.packed_image_len = &cb;
```

```
    UnPackImage( NULL, &cf, m_pUnpackedImage, &cb );
```

```
}
```

```
:
```

```
}
```

RotateImage

Function:

```
void RotateImage(  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
)
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|-----------|--|
| [in] | pInBuf | - input data buffer |
| [out] | pOutBuf | - output data buffer |
| [in] | pcbOutBuf | - size of the output buffer/actual output size |

Returns:

Remarks:

```
typedef struct _TLI_CF_ROTATE_IMAGE  
{  
    uint32_t width;  
    uint32_t height;  
    int16_t offset;  
    uint32_t bpp;  
    int16_t rotate;  
    void* input_data;  
} TLI_CF_ROTATE_IMAGE;
```

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::DrawImage( UINT nWidth, UINT nHeight, GEV_PIXEL_TYPE pixelType, PVOID  
pImage, HDC dc )
```

```
{
```

```
    PVOID pFinalImage = pImage;
```

```
    DWORD bpp = m_pBmpInfo->bmiHeader.biBitCount / 8;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight;
```

```
    switch(m_rotate)
```

```
    {
```

```
        case 0:
```

```
        {
```

```
            TLI_CF_ROTATE_IMAGE cf;
```

```
            cf.width = nWidth;
```

```
            cf.height = nHeight;
```

```
            cf.input_data = pImage;
```

```
            cf.rotate = 0;
```

```
            cf.bpp = bpp;
```

```
            RotateImage( &cf, pFinalImage, &cb );
```

```
        }
```

```
        break;
```

```
        case 180:
```

```
        {
```

```
            TLI_CF_ROTATE_IMAGE cf;
```

```
            cf.width = nWidth;
```

```
            cf.height = nHeight;
```

```
            cf.input_data = pImage;
```

```
            cf.rotate = 180;
```

```
            cf.bpp = bpp;
```

```
            RotateImage( &cf, pFinalImage, &cb );
```

```
        }
```

```
        :
```

```
    }
```

ConvertBayerPackedToRgb

Function:

```
void ConvertBayerPackedToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|--------------|--|
| [in] | hDev | - handle of an open device object |
| [in] | pInBuf | - input data buffer |
| [out] | pOutBuf | - output data buffer |
| [in] | pcbOutBuf | - size of the output buffer/actual output size |
| [out] | pUnPackBuf | - output unpacked data buffer |
| [in] | pcbUnPackBuf | - size of the unpacked buffer/actual output size |

Returns:

Remarks:

```
typedef struct _TLI_CF_CONVERT_BAYER_TO_RGB  
{  
    uint32_t pixel_type;  
    uint32_t width;  
    uint32_t height;  
    int16_t offset;  
    void* input_data;  
} TLI_CF_CONVERT_BAYER_TO_RGB;
```

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::GrabberThread()
```

```
{
```

```
:
```

```
PVOID pBase;
```

```
uint32_t cb = uint32_t(sizeof(pBase));
```

```
DSGetBufferInfo( m_hDS, hBuffer, BUFFER_INFO_BASE, NULL, &pBase, &cb );
```

```
:
```

```
if( m_pixelType == GVSP_PIX_BAYRG12_PACKED )
```

```
{
```

```
    BYTE *m_pUnpackedImage;
```

```
    BYTE *m_pOutputImage;
```

```
    uint32_t ucb = m_nImageWidth*m_nImageHeight*3/2;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight*6;
```

```
    TLI_CFC_CONVERT_BAYER_TO_RGB cf;
```

```
    cf.pixel_type = m_pixelType;
```

```
    cf.width = m_nImageWidth;
```

```
    cf.height = m_nImageHeight;
```

```
    cf.input_data = PBYTE(pBase);
```

```
    ConvertBayerPackedToRgb(
```

```
        NULL,
```

```
        &cf,
```

```
        m_pOutputImage,
```

```
        &cb,
```

```
        PWORD(m_pUnpackedImage),
```

```
        &ucb
```

```
    );
```

```
}
```

```
:
```

```
}
```

ConvertBayerToRgb

Function:

```
void ConvertBayerPackedToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|-----------|--|
| [in] | hDev | - handle of an open device object |
| [in] | pInBuf | - input data buffer |
| [out] | pOutBuf | - output data buffer |
| [in] | pcbOutBuf | - size of the output buffer/actual output size |

Returns:

Remarks:

```
typedef struct _TLI_CF_CONVERT_BAYER_TO_RGB  
{  
    uint32_t pixel_type;  
    uint32_t width;  
    uint32_t height;  
    int16_t offset;  
    void* input_data;  
} TLI_CF_CONVERT_BAYER_TO_RGB;
```

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::GrabberThread()
```

```
{
```

```
    :
```

```
    PVOID pBase;
```

```
    uint32_t cb = uint32_t(sizeof(pBase));
```

```
    DSGetBufferInfo( m_hDS, hBuffer, BUFFER_INFO_BASE, NULL, &pBase, &cb );
```

```
    :
```

```
    if( m_pixelType == GVSP_PIX_BAYRG8 )
```

```
    {
```

```
        BYTE *m_pOutputImage;
```

```
        uint32_t cb = m_nImageWidth*m_nImageHeight*3;
```

```
        TLI_CFC_CONVERT_BAYER_TO_RGB cf;
```

```
        cf.pixel_type = m_pixelType;
```

```
        cf.width = m_nImageWidth;
```

```
        cf.height = m_nImageHeight;
```

```
        cf.input_data = PBYTE(pBase);
```

```
        ConvertBayerToRgb(
```

```
            NULL,
```

```
            &cf,
```

```
            m_pOutputImage,
```

```
            &cb,
```

```
        );
```

```
    }
```

```
    :
```

```
}
```

ConvertMonoPackedToRgb

Function:

```
void ConvertMonoPackedToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf,  
    void *pUnPackBuf,  
    size_t *pcbUnPackBuf  
);
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|--------------|--|
| [in] | hDev | - handle of an open device object |
| [in] | pInBuf | - input data buffer |
| [out] | pOutBuf | - output data buffer |
| [in] | pcbOutBuf | - size of the output buffer/actual output size |
| [out] | pUnPackBuf | - output unpacked data buffer |
| [in] | pcbUnPackBuf | - size of the unpacked buffer/actual output size |

Returns:

Remarks:

```
typedef struct _TLI_CF_CONVERT_MONO_TO_RGB  
{  
    uint32_t pixel_type;  
    uint32_t width;  
    uint32_t height;  
    int16_t offset;  
    void* input_data;  
} TLI_CF_CONVERT_MONO_TO_RGB;
```

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::GrabberThread()
```

```
{
```

```
:
```

```
PVOID pBase;
```

```
uint32_t cb = uint32_t(sizeof(pBase));
```

```
DSGetBufferInfo( m_hDS, hBuffer, BUFFER_INFO_BASE, NULL, &pBase, &cb );
```

```
:
```

```
if( m_pixelType == GVSP_PIX_MONO12_PACKED )
```

```
{
```

```
    BYTE *m_pUnpackedImage;
```

```
    BYTE *m_pOutputImage;
```

```
    uint32_t ucb = m_nImageWidth*m_nImageHeight*3/2;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight*6;
```

```
    TLI_CF_CONVERT_MONO_TO_RGB cf;
```

```
    cf.pixel_type = m_pixelType;
```

```
    cf.width = m_nImageWidth;
```

```
    cf.height = m_nImageHeight;
```

```
    cf.input_data = PBYTE(pBase);
```

```
    ConvertMonoToRgb(
```

```
        NULL,
```

```
        &cf,
```

```
        m_pOutputImage,
```

```
        &cb,
```

```
        PWORD(m_pUnpackedImage,
```

```
        &ucd
```

```
    );
```

```
}
```

```
:
```

```
}
```

ConvertMonoToRgb

Function:

```
void ConvertMonoToRgb(  
    void *hDev,  
    void *pInBuf,  
    void *pOutBuf,  
    size_t *pcbOutBuf  
);
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|-----------|--|
| [in] | hDev | - handle of an open device object |
| [in] | pInBuf | - input data buffer |
| [out] | pOutBuf | - output data buffer |
| [in] | pcbOutBuf | - size of the output buffer/actual output size |

Returns:

Remarks:

```
typedef struct _TLI_CF_CONVERT_MONO_TO_RGB  
{  
    uint32_t pixel_type;  
    uint32_t width;  
    uint32_t height;  
    int16_t offset;  
    void* input_data;  
} TLI_CF_CONVERT_MONO_TO_RGB;
```

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::GrabberThread()
```

```
{
```

```
:
```

```
PVOID pBase;
```

```
uint32_t cb = uint32_t(sizeof(pBase));
```

```
DSGetBufferInfo( m_hDS, hBuffer, BUFFER_INFO_BASE, NULL, &pBase, &cb );
```

```
:
```

```
if( m_pixelType == GVSP_PIX_MONO8 )
```

```
{
```

```
    BYTE *m_pOutputImage;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight*3;
```

```
    TLI_CF_CONVERT_MONO_TO_RGB cf;
```

```
    cf.pixel_type = m_pixelType;
```

```
    cf.width = m_nImageWidth;
```

```
    cf.height = m_nImageHeight;
```

```
    cf.input_data = PBYTE(pBase);
```

```
    ConvertMonoToRgb(
```

```
        NULL,
```

```
        &cf,
```

```
        m_pOutputImage,
```

```
        &cb,
```

```
    );
```

```
}
```

```
:
```

```
}
```

ConvertRGBtoBGR8

Function:

```
void ConvertRGB8toBGR8(  
    const uint8_t *pSrc,  
    uint8_t *pDst,  
    uint32_t cbSrc  
);
```

Purpose:

Invokes custom manufacture-defined TLI function.

Parameters:

| | | |
|-------|-------|--|
| [in] | Src | - input data buffer |
| [out] | pDst | - output data buffer |
| [in] | cbSrc | - size of the output buffer/actual output size |

Returns:

Remarks:

Example:

```
#include Client.h
```

```
#include TLICustomFeatures.h
```

```
void CVideoWindow::GrabberThread()
```

```
{
```

```
:
```

```
PVOID pBase;
```

```
uint32_t cb = uint32_t(sizeof(pBase));
```

```
DSGetBufferInfo( m_hDS, hBuffer, BUFFER_INFO_BASE, NULL, &pBase, &cb );
```

```
:
```

```
if( m_pixelType == GVSP_PIX_BAYRG8 )
```

```
{
```

```
    BYTE *m_pOutputImage;
```

```
    BYTE *m_pConvertImage;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight*3;
```

```
    TLI_CFC_CONVERT_BAYER_TO_RGB cf;
```

```
    cf.pixel_type = m_pixelType;
```

```
    cf.width = m_nImageWidth;
```

```
    cf.height = m_nImageHeight;
```

```
    cf.input_data = PBYTE(pBase);
```

```
    ConvertBayerToRgb(
```

```
        NULL,
```

```
        &cf,
```

```
        m_pOutputImage,
```

```
        &cb,
```

```
    );
```

```
    ConvertRGB8toBGR8(
```

```
        m_pOutputImage,
```

```
        m_pConvertImage,
```

```
        &cb
```

```
    );
```

```
}
```

```
if( m_pixelType == GVSP_PIX_BAYRG12 )
```

```
{
```

```
    BYTE *m_pOutputImage;
```

```
    BYTE *m_pConvertImage;
```

```
    uint32_t cb = m_nImageWidth*m_nImageHeight*6;
```

```
    TLI_CFC_CONVERT_BAYER_TO_RGB cf;
```

```
    cf.pixel_type = m_pixelType;
```

```
    cf.width = m_nImageWidth;
```

```
    cf.height = m_nImageHeight;
```

```
    cf.input_data = PBYTE(pBase);
```

```
ConvertBayerToRgb(  
    NULL,  
    &cf,  
    m_pOutputImage,  
    &cb,  
    );  
  
ConvertRGB8toBGR8(  
    m_pOutputImage,  
    m_pConvertImage,  
    &cb  
    );  
}  
:  
}
```