

SC310 Extra Software Programming Guide

Custom Property

1. **KSPROPERTY_CUSTOM_GET_DEVICE_SERIAL_NUMBER_INFO** (0) (READ ONLY)

1. **KSPROPERTY_CUSTOM_GET_DEVICE_BUS_NUMBER_INFO** (2) (READ ONLY)

The property **KSPROPERTY_CUSTOM_GET_DEVICE_SERIAL_NUMBER_INFO** allows you to get Vendor ID (VID) and Product ID (PID) for the capture card. Vendor ID and product ID are 16-bit numbers used to identify PCI devices to a computer. The VID and PID are embedded in the capture card and communicated to the computer.

EXAMPLE#01: TO GET THE VENDER ID AND PRODUCT ID FROM THE CAPTURE CARD.

```
ULONG dwSerialNumber = 0x00000000;  
AMESDK_GET_CUSTOM_PROPERTY( hDevice, 0, &dwSerialNumber);
```

The property **KSPROPERTY_CUSTOM_GET_DEVICE_BUS_NUMBER_INFO** allows you to get current PCI bus number on the capture card. For example, the capture card on the first PCI slot, on the second PCI slot, or on the third PCI slot, etc.

EXAMPLE#02: TO GET THE BUS NUMBER ON THE CAPTURE CARD.

```
ULONG dwBusNumber = 0x00000000;  
AMESDK_GET_CUSTOM_PROPERTY( hDevice, 2, &dwBusNumber);
```

2. KSPROPERTY_CUSTOM_GET_ANALOG_VIDEO_SIGNAL_LOCK_STATUS (230) (READ ONLY)
2. KSPROPERTY_CUSTOM_GET_ANALOG_VIDEO_MACROVISION (202) (READ ONLY)

The property (230) is used to determine whether the signal is locked.

SUPPORT VALUE: 0 ~ 1 - UNLOCK ~ LOCK

EXAMPLE#01: TO GET THE CURRENT SIGNAL STATUS.

```
LONG nLock = 0x00;
```

```
AMESDK_GET_CUSTOM_PROPERTY( hDev, 230, &nLock );
```

The property (202) allows you to detect if the input's media content owns HDCP or MarcoVision protection.

Note!! To protect the content license, all behaviors in software porting should be complied with HDCP rules. Detect in any registered content of HDCP or MarcoVision, please disable the recording function in software.

SUPPORT VALUE: 0, 1 - NO ~ YES

EXAMPLE#06: GET HDCP PROTECT.

```
AMESDK_GET_CUSTOM_PROPERTY( hDev, 202, &HDCP );
```

```
if( HDCP == 1 ) { RECORD_FUNCTION = DISABLE; }
```

```
if( HDCP == 0 ) { RECORD_FUNCTION = ENABLE; }
```

- 3. **KSPROPERTY_CUSTOM_GET_ANALOG_VIDEO_SINGAL_DEBUG_INFO** (271) (READ ONLY)
- 3. **KSPROPERTY_CUSTOM_GET_PREVIEW_VIDEO_STARAM_FRAME_NUMBER_INFO** (351) (READ ONLY)
- 3. **KSPROPERTY_CUSTOM_GET_PREVIEW_AUDIO_STARAM_FRAME_NUMBER_INFO** (361) (READ ONLY)
- 3. **KSPROPERTY_CUSTOM_GET_ENCODER_VIDEO_DEFAULT_FRAME_NUMBER_INFO** (430) (READ ONLY)

The property **KSPROPERTY_CUSTOM_GET_ANALOG_VIDEO_SINGAL_DEBUG_INFO** is used to get the debug information in capture card running state. The output information is 32-bit error numbers. If the number is 0, the device is working properly. You can call it in timer function to get current signal status regularly.

SUPPORT VALUE: 0: GOOD

OTHERS: ERROR BITS

EXAMPLE#01: TO GET CURRENT SINGAL DEBUG STATUS.

```
ULONG dwSingalDebugInfo = 0x00000000;  
AMESDK_GET_CUSTOM_PROPERTY( hDevice, 271, &dwSingalDebugInfo);
```

The property **KSPROPERTY_CUSTOM_GET_PREVIEW_VIDEO_STARAM_FRAME_NUMBER_INFO** allows you to get the total number of frames in preview video. The property reads frame number information from video stream. You can call it in timer function to get current frame number regularly.

SUPPORT VALUE: FRAME NUMBER

EXAMPLE#02: TO GET VIDEO PREVIEW STREAM'S FRAME NUMBER.

```
ULONG dwPreviewVideoFrameNumber = 0;  
AMESDK_GET_CUSTOM_PROPERTY( hDev, 351, &dwPreviewVideoFrameNumber );
```

The property **KSPROPERTY_CUSTOM_GET_PREVIEW_AUDIO_STARAM_FRAME_NUMBER_INFO** allows you to get the total number of frames in preview audio. The property reads frame number information from audio stream. You can call it in timer function to get current frame number regularly.

SUPPORT VALUE: FRAME NUMBER

EXAMPLE#03: TO GET AUDIO PREVIEW STREAM'S FRAME NUMBER.

```
ULONG dwPreviewAudioFrameNumber = 0;  
AMESDK_GET_CUSTOM_PROPERTY( hDev, 361, &dwPreviewAudioFrameNumber);
```

The property **KSPROPERTY_CUSTOM_GET_ENCODER_VIDEO_DEFAULT_FRAME_NUMBER_INFO** allows you to get the total number of frames in video encoder. The property reads frame

number information from compressed video stream. You can call it in timer function to get current frame number regularly.

SUPPORT VALUE: FRAME NUMBER

EXAMPLE#04: TO GET VIDEO ENCODER STREAM STREAM'S FRAME NUMBER.

```
ULONG dwEncoderVideoFrameNumber = 0;
```

```
AMESDK_GET_CUSTOM_PROPERTY( hDev, 430, &dwEncoderVideoFrameNumber);
```

4. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_INPUT_AUTO_SCAN_ENABLED (232)

The property allows you to enable or disable the automatic scan video input signal source. If this function detects the actual video input source and format on capture card, it will automatically set the correct video input source and format.

SUPPORT VALUE: 0 ~ 1 - DISABLE ~ ENABLE

EXAMPLE#05 ENABLE THE AUTO INPUT SCAN FUNCTION

```
LONG enable = 0x01;
```

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 232, enable );
```

- 5. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_DEINTERLACE_TYPE (200)
- 5. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_VERTICAL_MIRROR (244)
- 5. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_HORIZONTAL_MIRROR (245)

The property offers one software-based de-interlace on interleaved video frame buffer. There are 4 methods to de-interlace video that can be controlled by you. After de-interlacing video, the incoming video frame buffer will become one progressive frame.

SUPPORT METHOD: 0: BOB
 1: WEAVE (OFF)
 2: LOW MOTION
 3: HIGH MOTION

EXAMPLE#01: SET DEINTERLACE METHOD TO WEAVE.

```
ULONG nDeinterlaceMethod = 1;  
AMESDK_SET_CUSTOM_PROPERTY( hDev, 200, nDeinterlaceMethod);
```

The two properties (244/245) are used to set mirror function. When mirror function is enabled, the vertical or horizontal video frame is inverted on display window. Same as deinterlacing, the property is used for display engine only.

SUPPORT VALUE: 0 ~ 1 - DISABLE ~ ENABLE

EXAMPLE#02: ENABLE THE VERTICAL MIRROR FUNCTION ON DISPLAY WINDOW

```
LONG enable = 0x01;  
AMESDK_SET_CUSTOM_PROPERTY( hDev, 244, enable);
```

EXAMPLE#03: ENABLE THE HORIZONTAL MIRROR FUNCTION ON DISPLAY WINDOW

```
LONG enable = 0x01;  
AMESDK_SET_CUSTOM_PROPERTY( hDev, 245, enable);
```

6. KSPROPERTY_CUSTOM_XET_ANALOG_AUDIO_VOLUME (251)

The property allows you to adjust hardware's audio volume. The support range is from 0 to 255. 0 is mute.

SUPPORT VALUE: 0 ~ 255 - 0% ~ 100%

EXAMPLE#01:

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 251, 0 );
```

EXAMPLE#02:

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 251, 128 );
```

7. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_IS_BLACKING_BOUNDARY_REMOVED (209)

The property allows you to check whether the video blanking data is removed. If it is enabled, the video blanking data is displayed on the screen. If it is disabled, the video blanking data is removed from the incoming data stream.

SUPPORT VALUE: 0 ~ 1 - DISABLE ~ ENABLE

EXAMPLE#01:

```
AMESDK_GET_CUSTOM_PROPERTY( hDev, 209, &BLANKING_REMOVED );
```


8. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_DEMISE_TYPE (212)

The property allows you to access hardware 3d demise property.

SUPPORT VALUE: 0 ~ 1 - OFF ~ ON

EXAMPLE#01: TO TRUN ON DEMISE FUNCTION.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 212, 0x00000001 );
```

EXAMPLE#02: TO TRUN OFF DEMISE FUNCTION.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 212, 0x00000000 );
```

9. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_FLEXIBLE_FPS_PATCH (218)

The property allows you to control the output format from one video capture filter. It allows you to adjust the video's frame rate from driver side. If it is disabled, the output frame rate is equal to input signal's frame rate.

SUPPORT VALUE: 0 ~ 1 - DISABLE ~ ENABLE

EXAMPLE#01: TO ENABLE FRAMERATE SCALER.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 218, 1 );
```

10. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_FLEXIBLE_RESOLUTION_PATCH (220)

The property allows you to adjust the video's resolution from hardware board. If it is disabled, the output resolution is equal to input signal's resolution. If it is enabled, we will enable one auto scalar to output customized format. For example, input resolution is 704x480 and capture output pin's resolution is 352x240.

SUPPORT VALUE: 0 ~ 1 - DISABLE ~ ENABLE

EXAMPLE#01: TO ENABLE RESOLUTION SCALER.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 220, 1 );
```

11. KSPROPERTY_CUSTOM_XET_ANALOG_VIDEO_QUEUE_BUFFER_SIZE (216)

The property allow you to specify the number of the rendered video frame in the queue buffer for a preview stream. By the default, the queue size of the corresponding a preview stream is set 10. Here we recommended use the size by default because this is implicated in many resource issues. For example, the unexpected signal error may occur if the total buffer sizes you want to set exceed the system capabilities.

Note: Setting queue buffer size will involve in dynamically allocated memory.

EXAMPLE#01: TO SET THE PREVIEW QUEUE SIZE TO 10 FRAMES

```
LONG nBfferSize = 10;
```

```
AMESDK_SET_CUSTOM_PROPERTY( hPreviewDevice, 216, nBfferSize );
```

12. KSPROPERTY_CUSTOM_XET_PREVIEW_VIDEO_STERAM_POST_RESOLUTION (350)

The property allows you to adjust current video resolution dynamically. The driver will re-allocate memory during changing video format on capture card running state.

SUPPORT VALUE: RESOLUTION = (WIDTH << 16) | (HEIGHT << 0)

EXAMPLE#01: TO SET PREVIEW VIDEO RESOLUTION DYNAMICALLY.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 350, &RESOLUTION );
```

13. KSPROPERTY_CUSTOM_XET_PREVIEW_VIDEO_STREAM_POST_SKIP_FRAMERATE (246)

13. KSPROPERTY_CUSTOM_XET_PREVIEW_VIDEO_STARAM_POST_AVG_FRAMERATE (247)

The property (246) allows you to adjust current video skip frame rate dynamically. The range of the property is from 1 to 255. It is identical to the skip number of frame. For example, the value 1 will generate the preview frame rate, 15.000fps.

SUPPORT VALUE: 0: DISABLE
 1, 2, 3, 4, ... SKIP

EXAMPLE#01: TO SET PREVIEW VIDEO SKIP FRAMERATE DYNAMICALLY.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 246, &FRAMERATE );
```

The property (247) allows you to adjust current video average frame rate dynamically. The range of the property is from 1 to 85. To enable it, our driver will follow the setting value to output one average fps. For example, 9 mean 9.00fps.

SUPPORT VALUE: 0: DISABLE
 1 ~ 85 FPS

EXAMPLE#02: TO SET PREVIEW VIDEO AVERAGE FRAMERATE DYNAMICALLY.

```
AMESDK_SET_CUSTOM_PROPERTY( hDev, 247, &FRAMERATE );
```

15. KSPROPERTY_CUSTOM_XET_GPIO_DIRECTION (940)

15. KSPROPERTY_CUSTOM_XET_GPIO_DATA (941)

15. KSPROPERTY_CUSTOM_XET_GPIO_SUPPORT (942) (READ ONLY)

The property allows you to access CX25820/1's GPIO interface. The property KSPROPERTY_CUSTOM_XET_GPIO_DIRECTION allows you to control its direction. Here, writing 1 to bit enables this pin as output pin. Usually, the GPIO is controlled by the first chipset in one board.

SUPPORT VALUE: 0 ~ 1 - INPUT ~ OUTPUT

The property KSPROPERTY_CUSTOM_XET_GPIO_DATA allows you to access GPIO's data.

SUPPORT VALUE: 0 ~ 1 - LOW ~ HIGH

The property KSPROPERTY_CUSTOM_XET_GPIO_SUPPORT allows you to obtain GPIO's information (pin size) on hardware board. Developer can use it to check if the device can support GPIO access.

SUPPORT VALUE: 0 IS NON-SUPPORT

EXAMPLE#01: TO DEFINE GPIO AS 16 OUTPUT PINS [0:15] AND 16 INPUT PINS [16:31].
`AMESDK_SET_CUSTOM_PROPERTY(hDev, 940, 0x0000FFFF);`

EXAMPLE#02: TO DEFINE GPIO AS 32 OUTPUT PINS [0:31] THEN PULL HIGH FOR ALL.
`AMESDK_SET_CUSTOM_PROPERTY(hDev, 940, 0xFFFFFFFF);`
`AMESDK_SET_CUSTOM_PROPERTY(hDev, 941, 0xFFFFFFFF);`

EXAMPLE#03: TO DEFINE GPIO AS 32 INPUT PINS [0:31] THEN READ DATA FROM IT.
`AMESDK_SET_CUSTOM_PROPERTY(hDev, 940, 0x00000000);`
`AMESDK_GET_CUSTOM_PROPERTY(hDev, 941, &GPIO);`

16. Application Note for AMESDK_GET_LOCK()

Customer to use AMESDK_GET_LOCK, please notes it. CX28820/1 is one 8CH integrated SOC. In order to reducing your software loading, we can group 8 channels' status into 8bits return value. You can call AMESDK_GET_LOCK to obtain 8CHs' status at the same time.

EXAMPLE#01: GET SC310N8 SIGNAL STATUS.

```
AMESDK_GET_LOCK( hDev[ 0 ], &status ); // GET CH01 ~ CH08 STATUS
ULONG status_ch01 = (status >> 0) & 0x01;
ULONG status_ch02 = (status >> 1) & 0x01;
...
ULONG status_ch08 = (status >> 7) & 0x01;
```

EXAMPLE#02: GET SC310N16 SIGNAL STATUS.

```
AMESDK_GET_LOCK( hDev[ 0 ], &status ); // GET CH01 ~ CH08 STATUS
ULONG status_ch01 = (status >> 0) & 0x01;
ULONG status_ch02 = (status >> 1) & 0x01;
...
ULONG status_ch08 = (status >> 7) & 0x01;
```

```
AMESDK_GET_LOCK( hDev[ 8 ], &status ); // GET CH09 ~ CH16 STATUS
ULONG status_ch09 = (status >> 0) & 0x01;
ULONG status_ch10 = (status >> 1) & 0x01;
...
ULONG status_ch16 = (status >> 7) & 0x01;
```


17. Access Custom Property for DirectShow Developer

Customer uses DirectShow to develop software can bypass our SDK to access CX25820/1 directly. All custom properties are implemented by IKsPropertySet interface. The interface can be queried from our capture source filter.

EXAMPLE#01: GET GPIO DATA.

```
static const GUID GUID_KPS_CX2581 = { 0xD1E5209F, 0x68FD, 0x4529, 0xBE, 0xE0, 0x5E, 0x7A, 0x1F, 0x47, 0x92, 0x17 };
ULONG gpio = 0;
m_pKsPropertySet->Set( GUID_KPS_CX2581, 941, NULL, 0, &gpio, sizeof(ULONG) );
```

18. Application Note for DirectShow Developer

The developer who uses DirectShow to access our capture source filter need check the frame size in the callback function of your SampleGrabber class. If the frame size is 0 bytes, it means the frame is one bad frame. You should drop it. More detail, please check with our engineer team directly.