

# Linux SDK EX8040S

## Application Note

**Revision 0.0.3**  
**2019/12/04**

## Legal

This is a confidential and proprietary document of Etron Technology, Inc. Any unauthorized use, reproduction, duplication, or disclosure of this document will be subject to the applicable civil and/or criminal penalties.

This document may be modified or revised subject to the discretion of Etron Technology, Inc. without further notice. Etron Technology, Inc. makes no warranty or representation to either the contents of this document or the products referenced in this document for any errors or omissions.

## Revision History

Rev	Date	Comments
0.0.1	2019/11/12	<ul style="list-style-type: none"><li>Draft release.</li></ul>
0.0.2	2019/11/18	<ul style="list-style-type: none"><li>Fixed some typo.</li></ul>
0.03	2019/12/4	<ul style="list-style-type: none"><li>Added Module Sync</li></ul>
		<ul style="list-style-type: none"><li></li></ul>

## Table of Contents

<b>1.</b>	<b>Introduction.....</b>	<b>5</b>
<b>2.</b>	<b>Device Initial.....</b>	<b>6</b>
2.1	Device Introduction .....	6
2.2	Initial.....	6
2.3	Flow chart.....	7
2.4	EtronDI_Initial .....	8
2.5	EtronDI_GetDeviceNumber.....	8
2.6	EtronDI_GetDeviceInfo .....	8
2.7	Reference File .....	8
<b>3.</b>	<b>Device Control .....</b>	<b>9</b>
3.1	color + depth device.....	9
3.2	Kolor device .....	10
3.2.1	Reference function.....	11
3.3	Reference File .....	13
<b>4</b>	<b>Preview .....</b>	<b>14</b>
4.2	Flow Chart .....	18
4.3	EtronDI_GetDeviceResolutionList .....	19
4.4	EtronDI_OpenDevice2 .....	19
4.5	EtronDI_GetColorImage.....	19
4.6	EtronDI_Rotate90 .....	19
4.7	Reference File .....	20
<b>5</b>	<b>Point Cloud.....</b>	<b>21</b>
4.1	Flow Chart .....	23
4.2	EtronDI_GetRectifyMatLogData .....	24
4.3	PlyWriter::etronFrameTo3DCylinder.....	24
4.4	PlyWriter::writePly.....	24
4.5	Reference File .....	25
<b>6</b>	<b>Multiple Module Sync .....</b>	<b>26</b>

## 1. Introduction

This document presents the Linux SDK main functions, functional flow chart and so on, the application programmer will know how to boot up the Etron Depth Map module (Device Initial), preview the color and depth images, Device Control and generate the point cloud file.

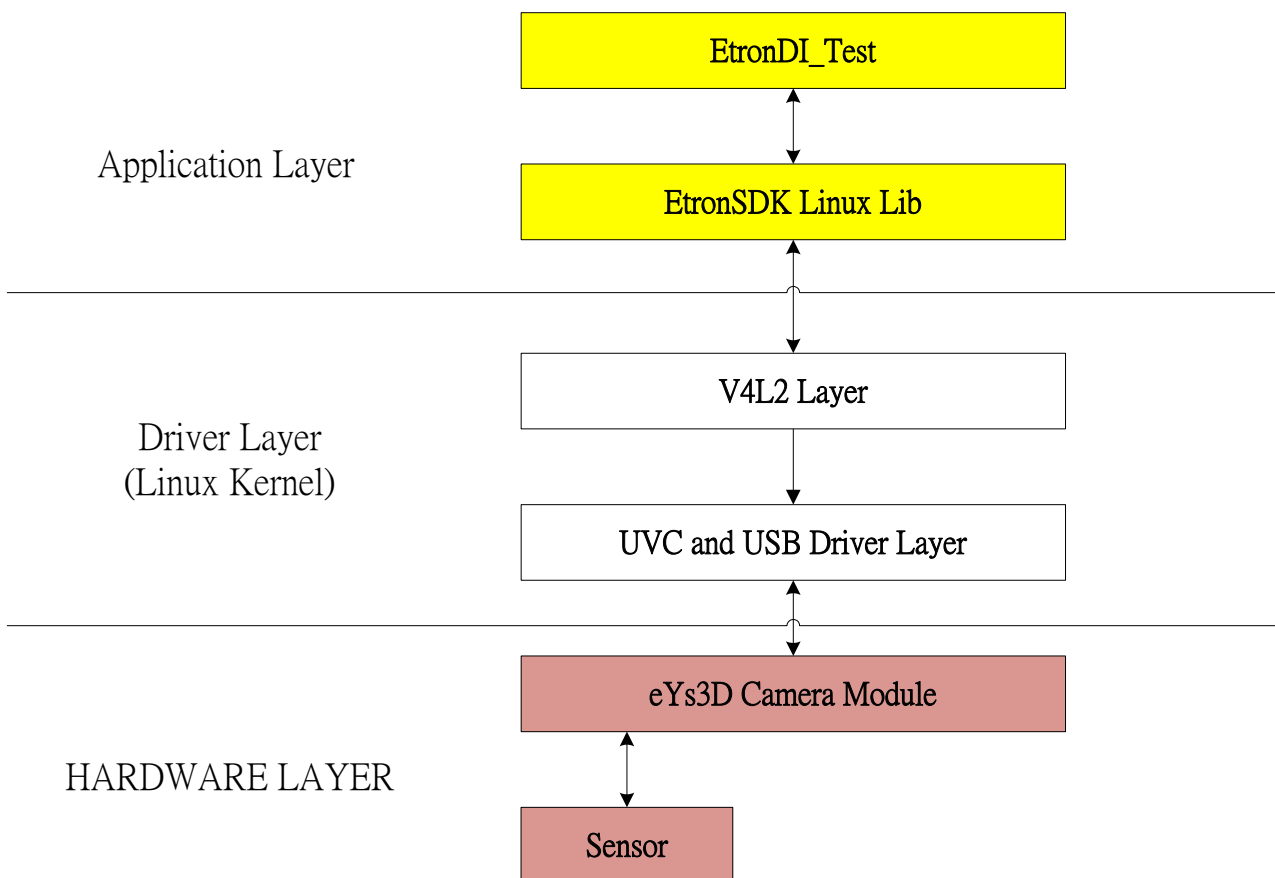
The EtronSDK\_Linux is included below items.

EtronSDK\_Linux/EtronDI\_Test: All functional application demo code

EtronSDK\_Linux/ eSPDI: EtronSDK Linux library.

EtronSDK\_Linux\doc: API Spec.

## Linux SW Architecture



## 2. Device Initial

Etron Depth Map Module plug in the Linux Platform (Eg, x86 PC, TX2, ...), the /dev/videoX will be created on V4L2, the EtronSDK\_Linux provide the application interface to access the /dev/videoX following UVC protocol.

This chapter descriptor how to initial the device.

### 2.1 Device Introduction

There are two video in EX8040S, one is C+D stream, other is K stream. The stream format table is below.

Video		Interface	USB1-EP1 Color + Depth-		USB2 - EP1 Color (2D or 3D)		Comment	Phase
L': Rectified Left, D: Depth, R: Right			Video	FPS	Video	FPS		
Mode 1	L+R (Calibration mode)	1x USB3.1 Gen1	1856x1056x2	24	N/A	N/A	for Calibration	1st
Mode 2	L+K (Calibration mode)	1x USB3.1 Gen1	1856x1056	2	3712x2112	2	for Calibration	1st
Mode 3	L'+R' (Calibration mode)	1x USB3.1 Gen1	1088x1920x2	24	N/A	N/A	for Calibration	1st
Mode 4	L'+D	1x USB3.1 Gen1	1080x1920x2	24	N/A	N/A	D11(default), Z14	1st
Mode 5	D+K (8M: K with MJPEG)	1x USB3.1 Gen1	912x1920	30	3840x1824	10	D11(default), Z14	1st
Mode 6	D+K (8M: K with MJPEG) (Multi-Module)	1x USB3.1 Gen1	912x1920	15	3840x1824	5	D11(default), Z14	1st
Mode 7	D+K' (4M: K' with MJPEG)	1x USB3.1 Gen1	912x1920	30	2560x1216	10	D11(default), Z14	2nd
Mode 8	D+K' (2M: K' with MJPEG) (Offload SW PP)	1x USB3.1 Gen1	456x960	30	1920x912	30	D11(default), Z14	2nd
Mode 9	D+K' (4M: K' with MJPEG) (Multi-Module)	1x USB3.1 Gen1	912x1920	15	2560x1216	5	D11(default), Z14	2nd
Mode 10	L+R (Calibration)	1x USB3.1 Gen1	1920x912x2	30	N/A	N/A	for Calibration	2nd
Mode 11	L+R+K (Calibration)	1x USB3.1 Gen1	1920x912x2	5	3840x1824	5	for Calibration	2nd
Mode 12	L'+D (Calibration)	1x USB3.1 Gen1	912x1920x2	30	N/A	N/A	for Calibration	2nd
Mode 13	L+R+K (Calibration)	1x USB3.1 Gen1	1920x912x2	5	2560x1216	5	for Calibration	2nd

### 2.2 Initial

First create device handler through by EtronDI\_Init API, get the Etron Device number from EtronDI\_GetDeviceNumber, get the device information (EtronDI\_GetDeviceInfo) to know device content and then select the devices, as below picture, 0 for /dev/video0 and 1 for /dev/video1

There are two devices in EX8040, B1335 for K stream of /dev/video0 and D777 for C+D stream of /dev/video1. We can create two Device Info classes to store device index. The m\_pDevInfoEx.index = 0 for K stream, m\_pDevInfo.index = 1 for C+D stream.

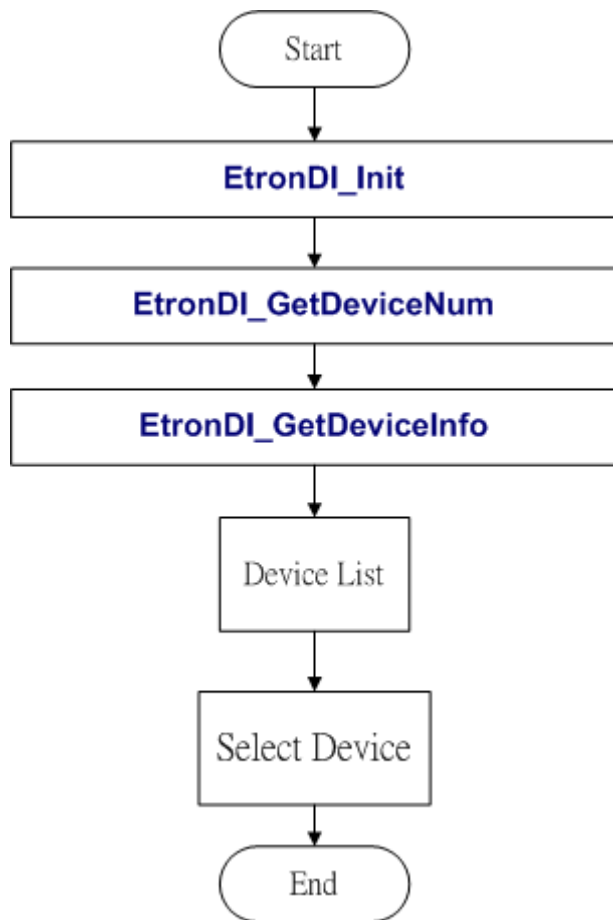
```

CVideoDevice: /dev/video0
CVideoDevice: /dev/video1
CVideoDevice: /dev/video2
nRet = 0
nDevCount = 2
Device Name = /dev/video0
PID = 0x0149
VID = 0x1e4e
Chip ID = 0x29
FW Version = EX8040S-C01-B1335-BL00U-004-BETA05

Device Name = /dev/video1
PID = 0x0131
VID = 0x1e4e
Chip ID = 0x15
FW Version = EX8040S-B01-D777-BL00U-004-BETA06

```

## 2.3 Flow chart



## **2.4 EtronDI\_Initial**

This function create device handler, the device /dev/videX will be opened, and the device information recorded in m\_pEtronDI (device handler). The device information is included PID, VID, ChipID and so on, the m\_pEtronDI (device handler) is a requirement of almost EtronSDK\_Linux API.

## **2.5 EtronDI\_GetDeviceNumber**

We can use this function to get the number of Etron Depth Map device.

## **2.6 EtronDI\_GetDeviceInfo**

This function will provide the device information such as PID, VID, Chip name and device name.

## **2.7 Reference File**

- EtronDI\_Test/Mainwindows.cpp
- EtronDI\_Test/Mainwindows.h
- eSPDI/eSPDI.h



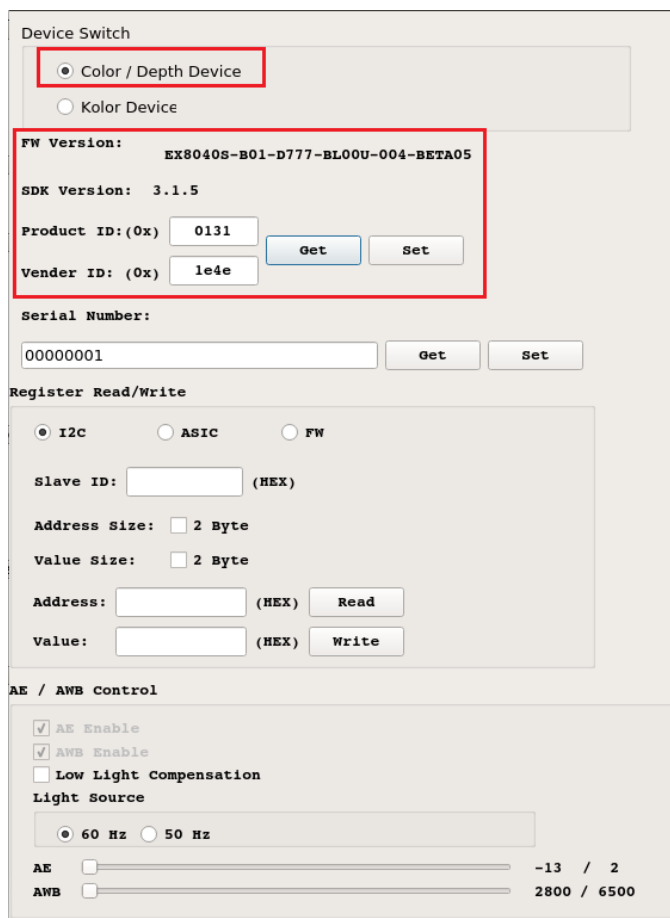
## 3. Device Control

As videodevicedlg.cpp example code and according to chap 2.2, we can select `m_pDevInfo.index = 1` for color + depth device, the `m_pDevInfoEx.index = 0` for K device.

### 3.1 color + depth device

If we select `m_pDevInfo.index = 1`, the Color/Depth device will control below functions.

- (1) Get Firmware Version
- (2) Get PID / VID
- (3) Get Serial Number
- (4) Register Read / Write
- (5) AE / AWB Control



The screenshot shows a GUI for device control with the following sections:

- Device Switch:** Contains two radio buttons. "Color / Depth Device" is selected and highlighted with a red box. "Kolor Device" is unselected.
- FW Version:** Displays "EX8040S-B01-D777-BL00U-004-BETA05".
- SDK Version:** Displays "3.1.5".
- Product ID: (0x):** A text box containing "0131".
- Vender ID: (0x):** A text box containing "1e4e".
- Get / Set buttons:** Two buttons are located to the right of the Product ID and Vender ID fields.
- Serial Number:** A text box containing "00000001".
- Get / Set buttons:** Two buttons are located to the right of the Serial Number field.
- Register Read/Write:** Contains three radio buttons: "I2C" (selected), "ASIC", and "FW". Below them are fields for "Slave ID: (HEX)", "Address Size: 2 Byte", "Value Size: 2 Byte", "Address: (HEX)", and "Value: (HEX)". "Read" and "Write" buttons are located to the right of the Address and Value fields respectively.
- AE / AWB Control:** Contains checkboxes for "AE Enable" (checked), "AWB Enable" (checked), and "Low Light Compensation" (unchecked). Below these is a "Light Source" section with two radio buttons: "60 Hz" (selected) and "50 Hz". At the bottom, there are two sliders: "AE" with a range of "-13 / 2" and "AWB" with a range of "2800 / 6500".

## 3.2 Kolor device

If we select `m_pDevInfoEx.index = 0`, the Kolor device will control below functions.

- (6) Get Firmware Version
- (7) Get PID / VID
- (8) Get Serial Number
- (9) Register Read / Write
- (10) AE / AWB Control

Device Switch

☐ Color / Depth Device
☒ Kolor Device

FW Version: EX8040S-C01-B1335-BL00U-004-BETA05
SDK Version: 3.1.5
Product ID: (0x) 0149
Vendor ID: (0x) 1e4e
Get Set

Serial Number:
00000001
Get Set

Register Read/Write

☒ I2C
☐ ASIC
☐ FW

Slave ID: (HEX)
Address Size: ☐ 2 Byte
Value Size: ☐ 2 Byte
Address: (HEX) Read
Value: (HEX) Write

AE / AWB Control

☒ AE Enable
☒ AWB Enable
☐ Low Light Compensation

Light Source

☒ 60 Hz
☐ 50 Hz

AE -13 / 2
AWB 2800 / 6500

### **3.2.1 Reference function**

#### **3.2.1.1 EtronDI\_GetFwVersion**

Get the firmware version of device.

#### **3.2.1.2 EtronDI\_GetPidVid**

Get PID(product ID) and VID(vendor ID) of device

#### **3.2.1.3 EtronDI\_SetPidVid**

Set PID(product ID) and VID(vendor ID) of device

#### **3.2.1.3 EtronDI\_GetSerialNumber**

Get Serial Number of device

#### **3.2.1.4 EtronDI\_SetSerialNumber**

Get Serial Number of device

#### **3.2.1.5 EtronDI\_GetCTPropVal**

Get camera terminal(CT) property value by v4l2\_control to get control value of camera terminal.

#### **3.2.1.6 EtronDI\_SetCTPropVal**

Setup camera terminal(CT) property value by v4l2\_control to get control value of camera terminal.

#### **3.2.1.7 EtronDI\_GetPUPropVal**

Get processing unit property value by v4l2\_control to get processing unit(PU) property value.

#### **3.2.1.8 EtronDI\_SetPUPropVal**

---

Set processing unit property value by v4l2\_control to get processing unit(PU) property value.

### 3.2.1.9 Parameters

CT\_PROPERTY\_ID\_AUTO\_EXPOSURE\_MODE\_CTRL  
CT\_PROPERTY\_ID\_AUTO\_EXPOSURE\_PRIORITY\_CTRL  
CT\_PROPERTY\_ID\_EXPOSURE\_TIME\_ABSOLUTE\_CTRL  
CT\_PROPERTY\_ID\_EXPOSURE\_TIME\_RELATIVE\_CTRL  
CT\_PROPERTY\_ID\_FOCUS\_ABSOLUTE\_CTRL  
CT\_PROPERTY\_ID\_FOCUS\_RELATIVE\_CTRL  
CT\_PROPERTY\_ID\_FOCUS\_AUTO\_CTRL  
CT\_PROPERTY\_ID\_IRIS\_ABSOLUTE\_CTRL  
CT\_PROPERTY\_ID\_IRIS\_RELATIVE\_CTRL  
CT\_PROPERTY\_ID\_ZOOM\_ABSOLUTE\_CTRL  
CT\_PROPERTY\_ID\_ZOOM\_RELATIVE\_CTRL  
CT\_PROPERTY\_ID\_PAN\_ABSOLUTE\_CTRL  
CT\_PROPERTY\_ID\_PAN\_RELATIVE\_CTRL  
CT\_PROPERTY\_ID\_TILT\_ABSOLUTE\_CTRL  
CT\_PROPERTY\_ID\_TILT\_RELATIVE\_CTRL  
CT\_PROPERTY\_ID\_PRIVACY\_CTRL  
PU\_PROPERTY\_ID\_BACKLIGHT\_COMPENSATION\_CTRL  
PU\_PROPERTY\_ID\_BRIGHTNESS\_CTRL  
PU\_PROPERTY\_ID\_CONTRAST\_CTRL  
PU\_PROPERTY\_ID\_GAIN\_CTRL  
PU\_PROPERTY\_ID\_POWER\_LINE\_FREQUENCY\_CTRL  
PU\_PROPERTY\_ID\_HUE\_CTRL  
PU\_PROPERTY\_ID\_HUE\_AUTO\_CTRL

PU\_PROPERTY\_ID\_SATURATION\_CTRL

PU\_PROPERTY\_ID\_SHARPNESS\_CTRL

PU\_PROPERTY\_ID\_GAMMA\_CTRL

PU\_PROPERTY\_ID\_WHITE\_BALANCE\_CTRL

PU\_PROPERTY\_ID\_WHITE\_BALANCE\_AUTO\_CTRL

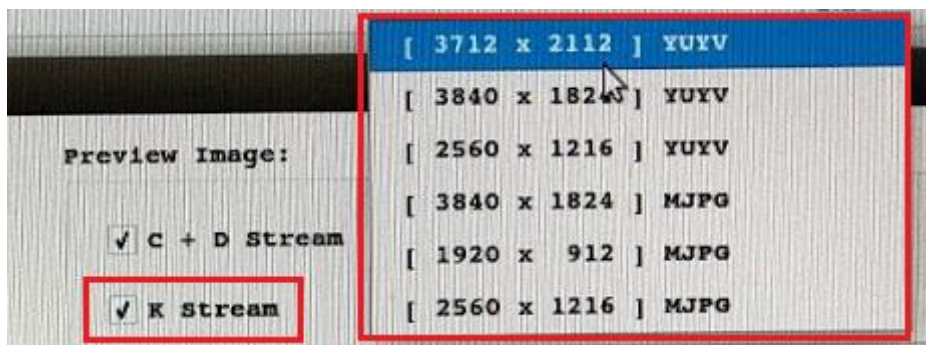
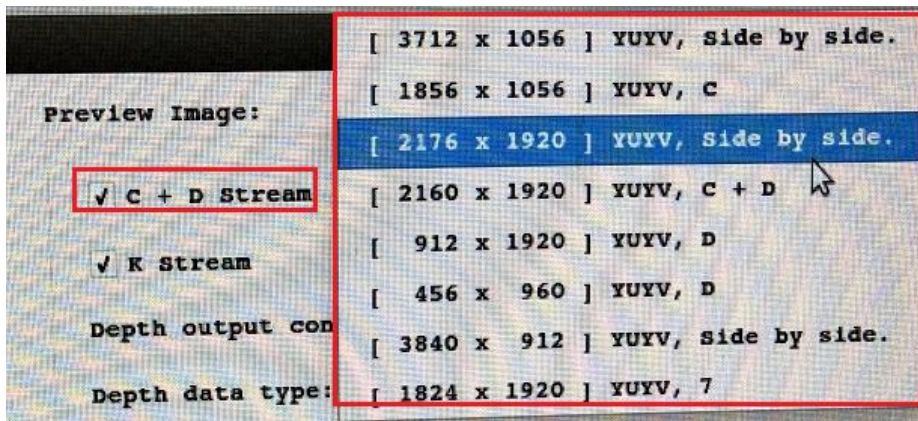
### **3.3 Reference File**

- EtronDI\_Test/videodevicedlg.cpp
- EtronDI\_Test/ videodevicedlg.h
- eSPDI/eSPDI.h

## 4 Preview

The Etron Depth module provided the depth and color video stream. This chapter descriptor previewed the color video and depth video through by EtronSDK\_Linux.

Fisrt we need initial the device (please refer the Device Initial chapter), setup the Depth Data Type, and doing EtronDI\_GetDeviceResolutionList to get the all resolution from C+D Device and Kolor device, and then we will know which resolution and MJPEG supported.



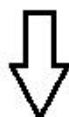
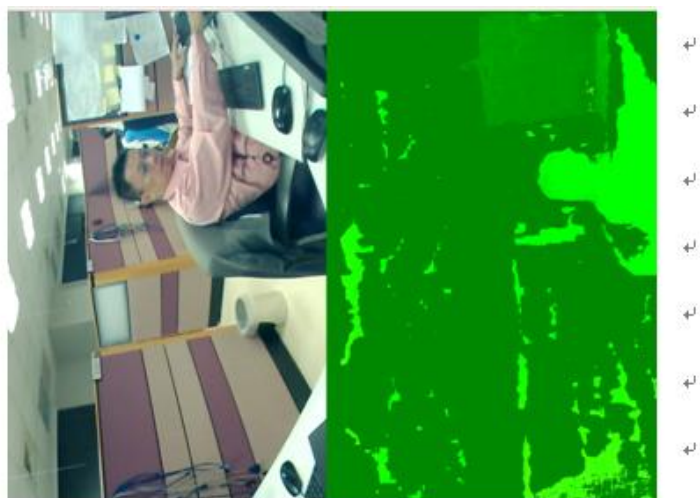
And do EtronDI\_OpenDevice2 to open C+D and Kolor device.

We need create the two dialog on for color video image and one for depth image. The C+D image is from the EtronDI\_GetColorImage and the Kolor image is from the EtronDI\_GetColorImage.

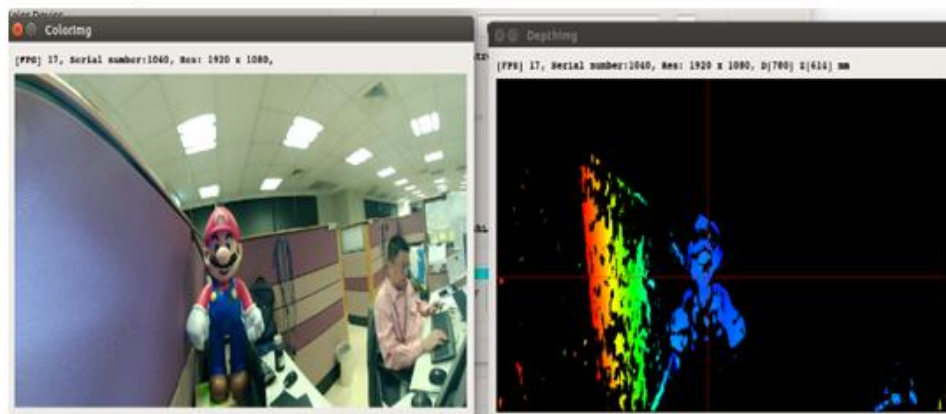
The C + D image should be rotated 90, K image do not need rotate 90.

For example 2160 x 1920 YUYV C+ D mode, we need separate the Color Image and Depth Image, and then rotate both color and Depth image, the resolution are 1920 x 1080.



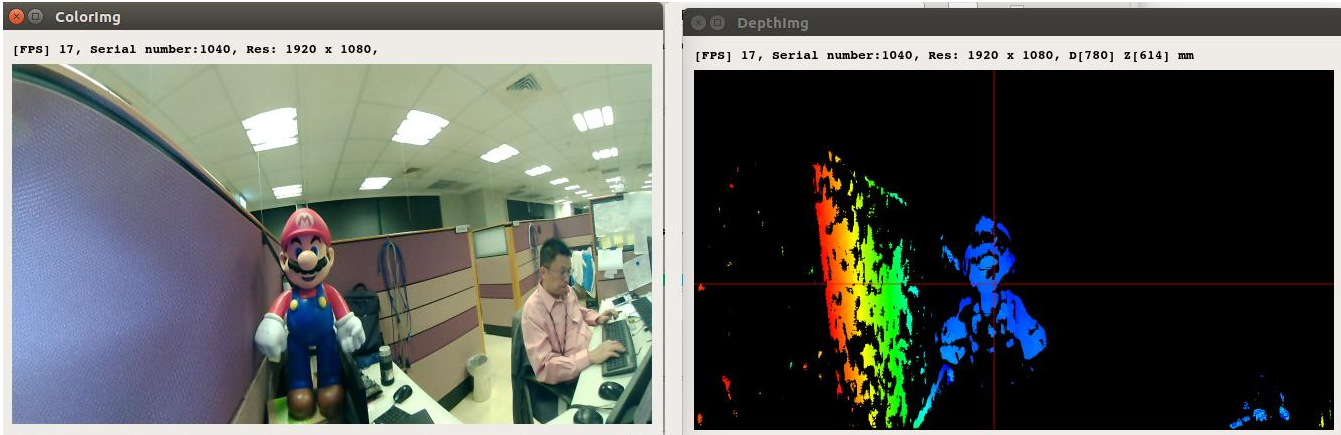


Rotate 90 °  
and YUYV2RGB

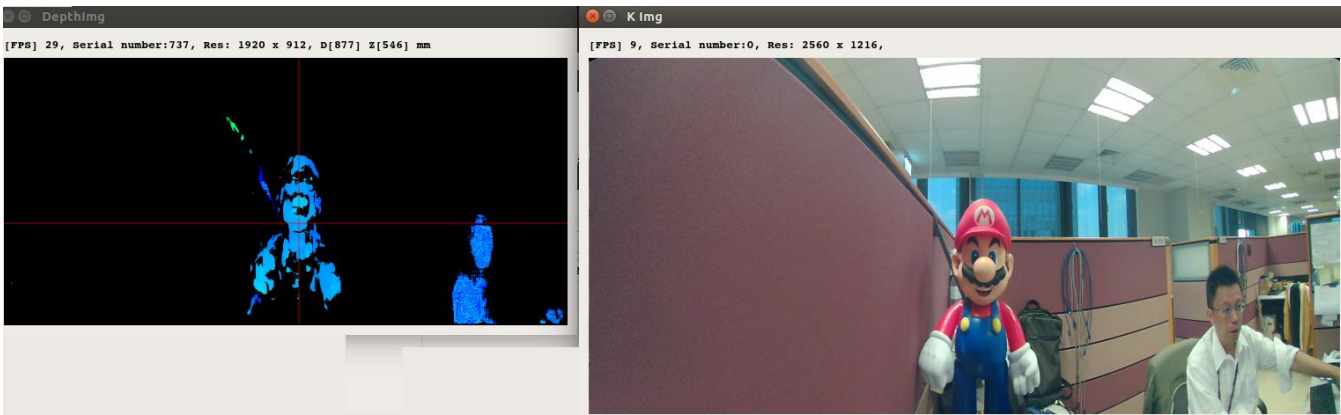




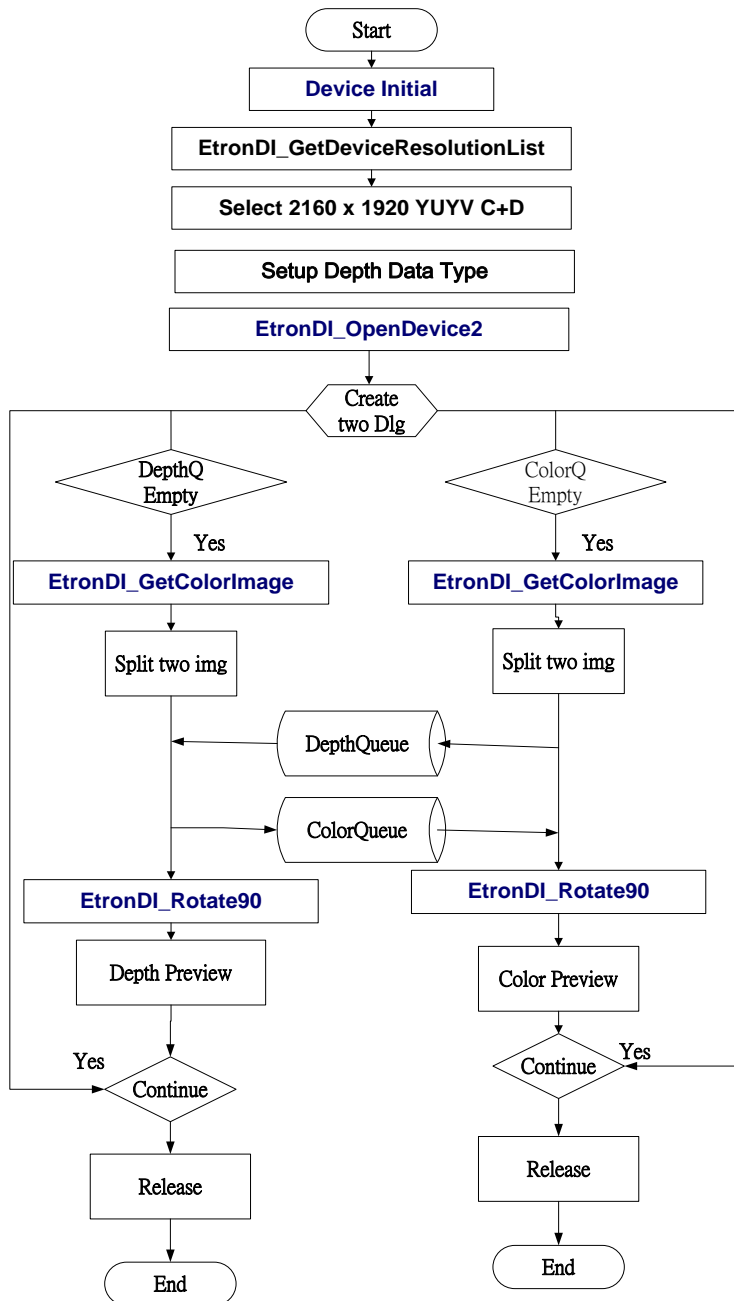
2160 x 1080 YUYV C+D



1920 x 912 YUYV Depth and 2560 x 1216 Kolor MJPEG



## 4.2 Flow Chart



### **4.3 EtronDI\_GetDeviceResolutionList**

Get the device resolution List

### **4.4 EtronDI\_OpenDevice2**

This function provided opening both depth and color device.

The implement layer to open Etron camera device by V4L2(<https://en.wikipedia.org/wiki/Video4Linux>),

It open color and depth at one time call, do functions as below,

1. Initialize the USB device by V4L2 protocol
  - Query device v4l2 capability, e.g. video capability, streaming capability
  - Setup the resolution mode to UVC driver and check result
  - Initialize memory buffer mapping from kernel to user mode
2. Enumerate frame interval to set frame rate
3. Start video capture processes

### **4.5 EtronDI\_GetColorImage**

Get Color image only by this function.

### **4.6 EtronDI\_Rotate90**

Rotate the image 90 degree.

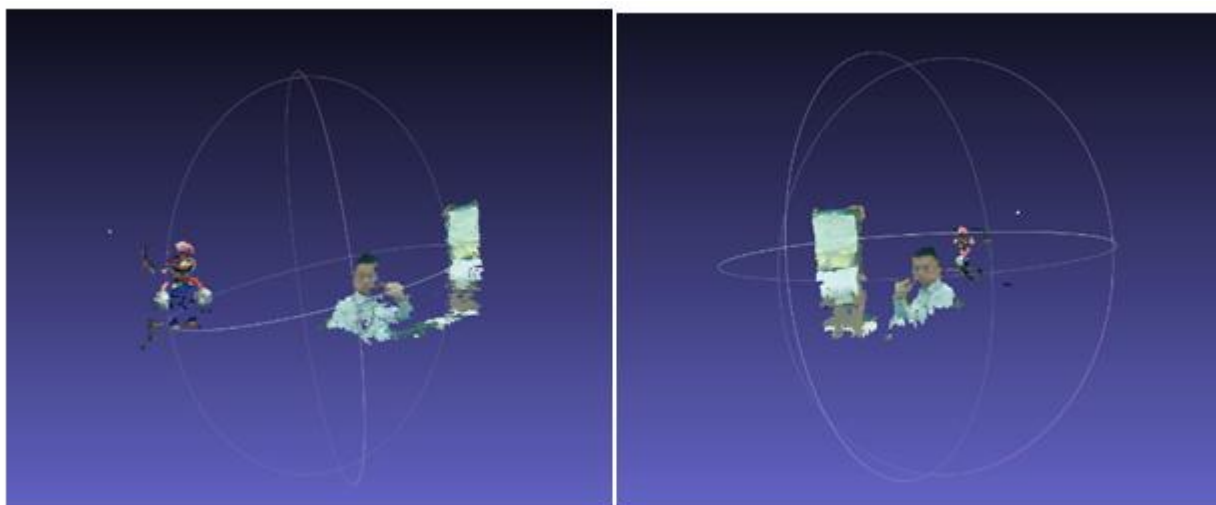
#### **4.7 Reference File**

- EtronDI\_Test/Mainwindows.cpp
- EtronDI\_Test/Mainwindows.h
- eSPDI/eSPDI.h
- EtronDI\_Test/videodevicedlg.cpp
- EtronDI\_Test/videodevicedlg.h
- EtronDI\_Test/colordlg.cpp
- EtronDI\_Test/colordlg.h
- EtronDI\_Test/depthdlg.cpp
- EtronDI\_Test/depthdlg.h

## 5 Point Cloud

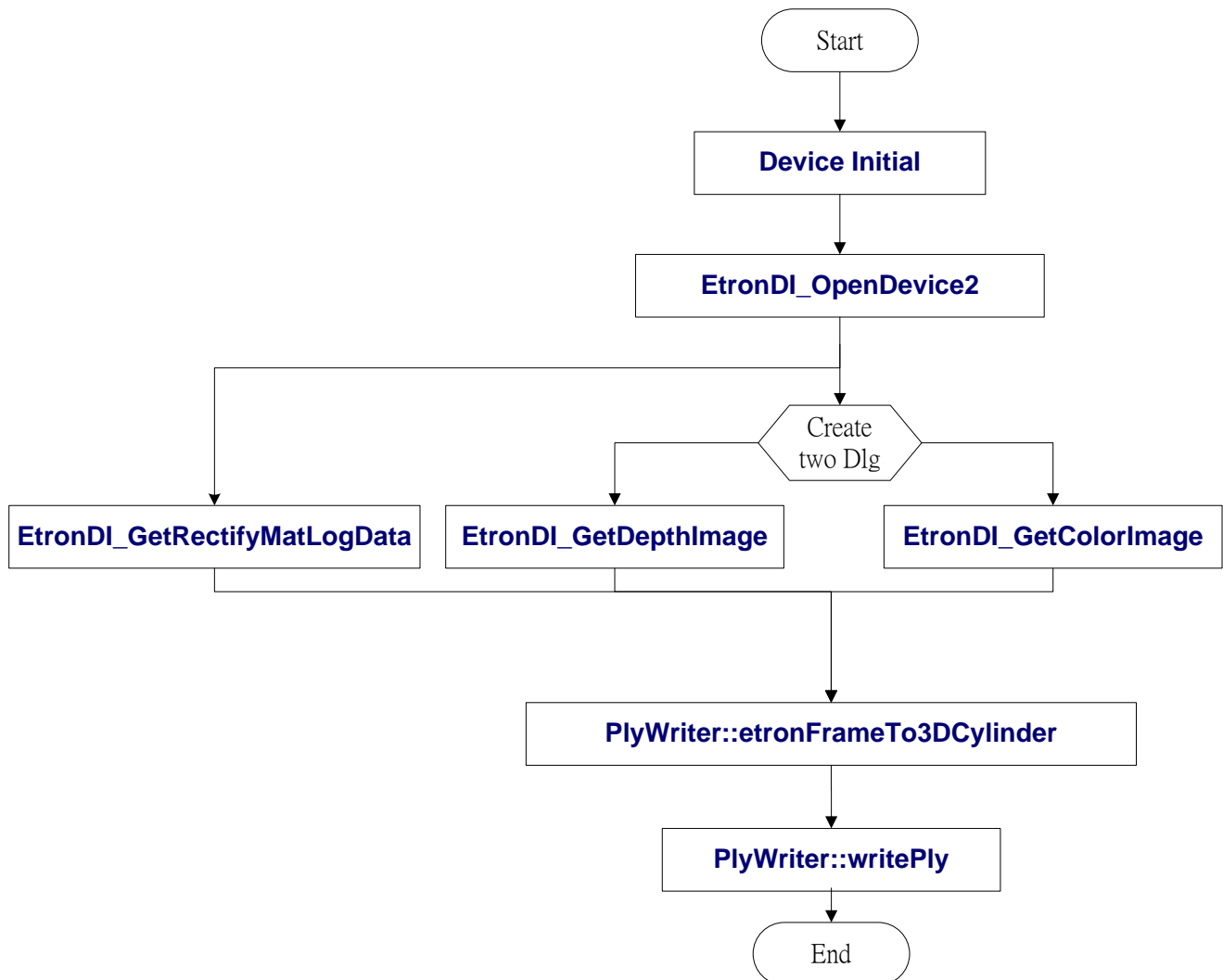
EtronSDK\_Linux support 3D image point cloud, we can generate the ply (**Polygon File Format**) image, we can read ply file through by MeshLab tools as below pictures.

The ply file need Rectify data, Depth Map data and ColorImage, they are from EtronDI\_GetRectifyMatLogData, EtronDI\_GetDepthImage and EtronDI\_GetColorImage, put Rectify data, Depth Map data and ColorImage to PlyWriter::etronFrameTo3D to get 3D image, and put the 3D image to PlyWriter::writePly to get ply image.





## 4.1 Flow Chart



#### **4.2 EtronDI\_GetRectifyMatLogData**

This function can get the Rectify Math Data from Etron Depth Map module.

#### **4.3 PlyWriter::etronFrameTo3DCylinder**

Generated the 3D image from etronFrameTo3DCylinder function, we shell input depth map data, color RGB image and Rectify Math Data.

#### **4.4 PlyWriter::writePly**

Input the 3D image in this function to generate ply file.



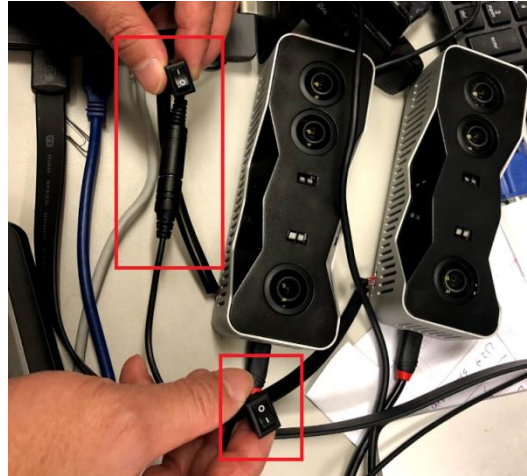
#### **4.5 Reference File**

- EtronDI\_Test/Mainwindows.cpp
- EtronDI\_Test/Mainwindows.h
- eSPDI/eSPDI.h
- EtronDI\_Test/videodevicedlg.cpp
- EtronDI\_Test/videodevicedlg.h
- EtronDI\_Test/colordlg.cpp
- EtronDI\_Test/colordlg.h
- EtronDI\_Test/depthdlg.cpp
- EtronDI\_Test/depthdlg.h
- EtronDI\_Test/plywriter.cpp
- EtronDI\_Test/plywriter.h

## 6 Multiple Module Sync

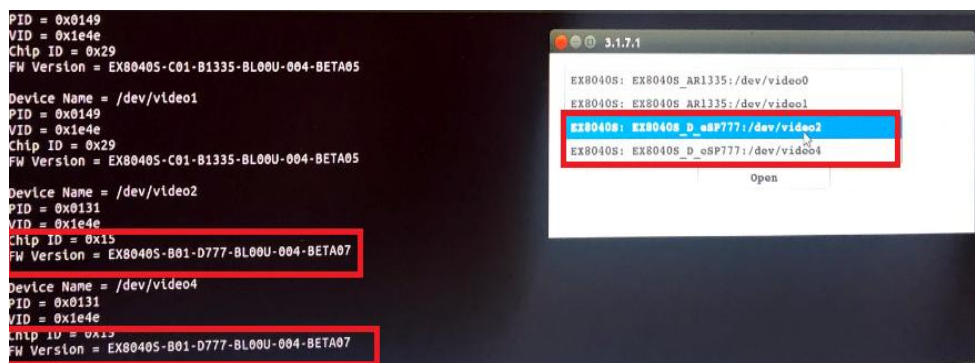
Two module sync testing steps following,

- (1) Connect two module on one or two PC
- (2) Sync Line config

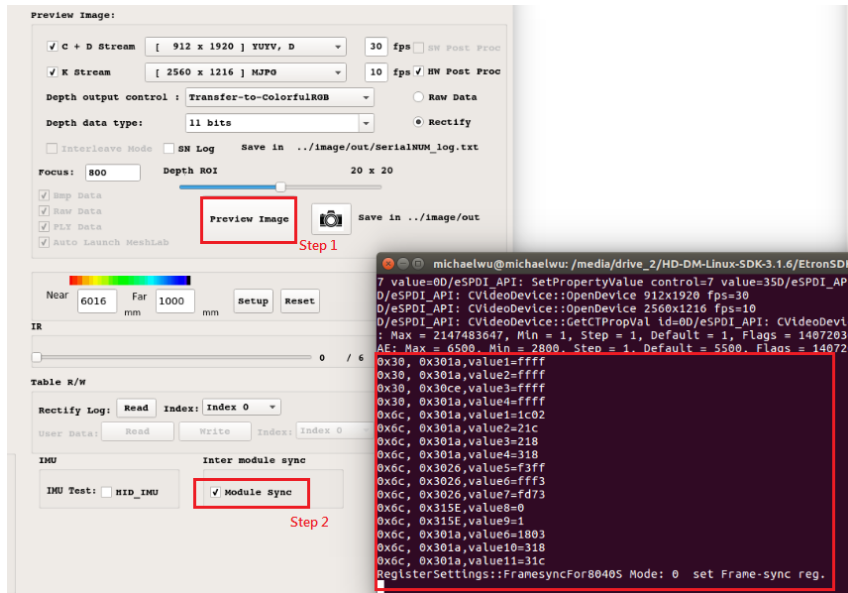


- (3) Open device

If two module connect one PC, we need do run\_qt/run\_x86\_EtronDI\_Test\_EX8040.sh two times, and open eSP777 device for each module.



- (4) Do preview for each module
- (5) Checked Module Sync Checked Box



- (6) The Module Sync function please reference to RegisterSetting.cpp Fraamsync() function