Simply Brighter

(In Canada)
111 Railside Road
Suite 201
Toronto, ON M3A 1B2
CANADA
Tel: 1-416-840 4991
Fax: 1-416-840 6541

(In US)
1241 Quarry Lane
Suite 105
Pleasanton, CA 94566
USA
Tel: 1-925-218 1885
Email: sales@mightex.com

Mightex Buffer USB CCD Camera USB Protocol

Version 1.0.6

Dec. 24, 2018

Relevant Products

Part Numbers
CCN-B013-U, CCE-B013-U,CCN-C013-U, CCE-C013-U, CGN-B013-U, CGE-B013-U,CGN-C013-U, CGE-C013-U, CXN-B013-U, CXE-B013-U, CXN-C013-U, CXE-C013-U, CCN-B020-U, CCE-B020-U,CCN-C020-U, CCE-C030-U

Revision History

Revision	Date	Author	Description
1.0.0	Aug. 28, 2008	JT Zheng	Initial Revision
1.0.1	Oct. 21, 2008	JT Zheng	C-Mount Modals only
1.0.2	Dec. 21, 2008	JT Zheng	Add CGX modals
1.0.3	Jan. 16, 2010	JT Zheng	Add CXX modals
1.0.4	Apr. 16, 2010	JT Zheng	Gain description for Color modal
1.0.5	Apr. 26, 2010	JT Zheng	Adding CCX-B020-U/CCX-C020-U
1.0.6	Dec. 24, 2018	JT Zheng	New Mightex Logo

Mightex USB 2.0 Buffer CCD camera is designed for low cost machine vision and other applications, With USB 2.0 high speed interface and powerful PC camera engine, the camera delivers CMOS image frames at high frame rate. The GUI demostration application and SDK are provided for user's application developments, For users who want to embedded it to Non-Windows based system, the USB2.0 based command protocol is described below.

Mightex USB 2.0 Buffer USB camera is using Cypress CY68013A as its on board USB device controller, please refer to the Cypress documents for the details of the chip's USB protocol supporting.

Descriptors

Device Descriptor:

bcdUSB:	0x0200
bDeviceClass:	0x00
bDeviceSubClass:	0x00
bDeviceProtocol:	0x00
bMaxPacketSize0:	0x40 (64)
idVendor:	0x04B4 (For evaluation samples)
idProduct:	0x0528
bcdDevice:	0x0000
iManufacturer:	0x01
	0x0409: "Mightex"
iProduct:	0x02
	0x0409: "USB- BUF-CCD-1"
iSerialNumber:	0x00
bNumConfigurations:	0x01

Configuration Descriptor:

wTotalLength:	0x0027
bNumInterfaces:	0x01
bConfigurationValue:	0x01
iConfiguration:	0x00
bmAttributes:	0x80 (Bus Powered)
MaxPower:	0xE1 (450 mA)

Interface Descriptor:

bInterfaceNumber:	0x00
bAlternateSetting:	0x00
bNumEndpoints:	0x03
bInterfaceClass:	0xFF
bInterfaceSubClass:	0x00
bInterfaceProtocol:	0x00
iInterface:	0x00

Endpoint Descriptor:

bEndpointAddress:	0x01 **OUT**
Transfer Type:	Bulk
wMaxPacketSize:	0x0200 (512)
bInterval:	0x00
bEndpointAddress:	0x81 **IN**
Transfer Type:	Bulk
wMaxPacketSize:	0x0200 (512)
bInterval:	0x00
bEndpointAddress:	0x82 **IN**
Transfer Type:	Bulk
wMaxPacketSize:	0x0200 (512)
bInterval:	0x00

IMPORTANT:

1). End point 0 is NOT listed above, as it's used for standard USB enumeration and configuration, for details of EP0 protocols, please refer to the latest USB specification and Cypress CY68013A manual.

2). End point 1 (for IN and OUT) and 2 (for IN only) are used for camera specific commands, this documents is focused on the data communications occurred on these End points.

The protocol is extremely simplified for quick understanding and development.

End points 1 (IN & OUT)

End Point 1(OUT) and 0x81(IN) are used for command control, it's always the Host initiates a command to Camera on EP1(OUT), and according to the command, Camera may prepare Response in its buffer and waiting for Host to fetch. The command and response has the following generic format:

EP(0x01)	→	CommandID	Length	Data
EP(0x81)	←	Result	Length	Data

For **Command**:
CommandID : This is a ONE byte field, represent the command itself, it tells device what to do.
Length: The length of data in byte.
Data: Depends on the command ID.

For **Response**:
Result: 0x01 – Means OK, 0x00 – means Error
Length: The length of response data in byte. (When Result is OK)
Data: Depends on the command ID.

Note that in the following command description, the "→" means data flow on EP1 OUT, "←" means data flow on EP1 IN.

****. Query Firmware Version (CommandID = 0x01)***

→ 0x01	1	0x01 or 0x02	
← 0x01	3	Major	Minor Revision

Host can use this command to query version of the firmware from device, device will response the current firmware version. There're two firmwares on device, firmware for USB interface (CY68013A) chip and firmware for DSP. While the Data is 0x01, the return value is version information for USB interface chip, 0x02 will return the version information for DSP.

****. Query Device Information (CommandID = 0x21)***

→0x21	1	0x00
←0x01	sizeof(tDeviceInfo)	DeviceInfo

For this command, the data field (0x00) is not used. Upon receiving this command, device will prepare device information and put in the EP1(IN) buffer, for host to fetch. The tDeviceInfo has the following definition:

```
#define STRING_LENGTH   14
typedef struct
{
    BYTE    ConfigRevision;
    BYTE    ModuleNo[STRING_LENGTH];
    BYTE    SerialNo[STRING_LENGTH];
    BYTE    UserSerialNo[STRING_LENGTH];
} tDeviceInfo;
```

it's actually a 43 bytes data structure which contains the ModuleNo[] and SerialNo[], these two fields are important if user wants to support multiple devices in the host software, the serialNo[] might be the only information for user to distinct the devices of the same type. For ModuleNo[], it contains the Module type (B013, C013 or C030…etc.) information, which is useful while there're different types of cameras connected.

Note: For above two commands, usually host uses them after the camera is powered up, host might have to wait for a while (e.g. 3 seconds) after the camera is turned on before sending these two queries, as the device needs time to boot up and the USB enumeration will also take some time.

****. Camera Work Mode Setting (commandID = 0x30)***

→ 0x30	2	MODE (0 or 1) CameraBit(8 or 12)
		#define NORMAL_MODE 0x00
		#define TRIGGER_MODE 0x01

This command is used for setting the current mode of the device, there's no response for this command, for the details of NORMAL and TRIGGER mode, please refer to Camera's user manual and SDK manual. After sending this command to camera, Host might wait for a while (e.g. 100ms) before sending next command, As camera will take some time to switch the mode.

****. Camera CCD Frequency Setting (commandID = 0x32)***

→ 0x32	1	Frequency ID

This command is used for setting the working frequency of the CCD sensor, by default (power up), the working frequency is Full frequency (e.g. for CCX modules, it's 28MHz), user might set it to half, one fourth, one eithth and one sixteenth of the full frequency (e.g. for CCX modules, they are 14M, 7M, 3.5M and 1.75M.) After sending this command to camera, Host might wait for a while (e.g. 100ms) before sending next command, As camera will take some time to switch the mode.

For **CCX** modules, the Frequency ID is as following:

For CCX-C013-U and CCX-B013-U, the ID is:

0 – 28MHz
1 – 14MHz
2 – 7MHz
3 – 3.5MHz
4 – 1.75MHz

For CCX-C020-U and CCX-B020-U, the ID is:

0 – 32MHz
1 – 16MHz
2 – 8MHz
3 – 4MHz
4 – 2MHz

For **CXX** modules, the Frequency ID is as following:

0 – 28MHz
1 – 18MHz
2 – 14MHz
3 – 7MHz
4 – 3.5MHz

For **CGX** modules, the Frequency ID is as following:

0 – 32MHz
1 – 16MHz
2 – 8MHz
3 – 4MHz
4 – 2MHz

Important: For applications frame rate is NOT a concern, user might set to a lower CCD Frequency, which might provide better SNR.

. Get current buffered image frames (commandID = 0x33)

→0x33	1	0x00
←0x01	6	FrameCount RowSize(MSB,LSB) ColumnSize(MSB,LSB) Bin

Device has on board frame buffer for buffering image frames in both NORMAL and TRIGGER modes, currently, the buffer can hold up to at least 8 frames. (for maximum resolution @12bit mode) it's settable via "Set Resolution" command) This gives host the flexibility to do other operations, not worry the loss of the continuous frames (If the on camera frame buffer are FULL and host still doesn't fetch them, the camera will still grab frames from CCD sensor but simply ignore them).

It's always recommended for host to query the buffered frames before fetching image data (from EP2), as this is important information for host to arrange the proper buffer for the image data.
The RowSize/ColumnSize/Bin is the current settings on camera, note that it might be useful while Host set a new resolution to camera, after device gets the "Set resolution" command, the camera might take some time to do the house keeping, which includes the clean up of the frames (with "Older" resolution setting) in device buffer. So it's always recommended to check the RowSize/ColumnSize/Bin…make sure it's the same as the last resolution set to the camera, otherwise, even the FrameCount is Non-Zero…those frame should be ignored.
The RowSize, ColumnSize and Bin should be the same as the settings in command 0x60 (Set Resolution).

. Get buffered image data (commandID = 0x34)

→0x34	1	Frame Count (Get image data)

This is a very important command, Host sends the command on EP1(OUT), and there's no response on EP1(IN), However, the frame data will be ready on EP2(IN) for host to fetch. So Host should prepare proper buffer and fetch image data from EP2(IN) after this command is sent. Another very important point is: The "Frame Count" here is the number of frames host wants to fetch this time, usually, user should use command (0x33) to get the count of current available buffered frames, and use the returned frame count for this command(0x34), or if user have limited memory for image data, user can set the frame count less than the number returned from command(0x33), but NEVER bigger than that number.

For the details of Frame Data on EP2, please refer to the following "End Point2(IN)".

. Ignore buffered image data (commandID = 0x34)

→0x35	1	Frame Count (Ignore image data)

This is similar to above command 0x34, however, instead of fetching image data from EP2, this command simply ask the camera to ignore the frames (Frame Count defines the number). Host send the command on EP1(OUT), and there's no response on EP1(IN). A very important point is: The "Frame Count" here is the number of frames host wants to ignore this time, usually, user should use command (0x33) to get the count of current available buffered frames, and use the returned frame count or less for this command(0x35), to be sure NEVER bigger than frame count returned by command (0x33).

User might want to use this command when there're several frames in on-camera buffer, but as Some of them are "old", user might ignore them and only get the last one with command (0x34).

. Simulate Trigger assertion (commandID = 0x36)

→0x36	1	0

When the camera is set in TRIGGER mode, camera will grab one frame upon each external trigger assertion, however, host might send this command to camera which has the same effect as the external trigger assertion.

. GPIO Chip Register Write (commandID = 0x40)

→ 0x40	2	Register Value

. GPIO Chip Register Read (commandID = 0x41)

→0x41	1	Register

←0x01 1 Value

The device has a on board Philips PCA9536 chip which provide 4 GPIO pins, for user wants to use it, user can use the above two commands for writing and reading register values. For details of the register and their definitions, please refer to the specification of PCB9536.

. Reset Control (commandID = 0x50)

→ 0x50 1 ResetType

ResetType can be one of:

```
#define RESET_CAMERA    0x01  // Reset USB I/F Chip, ISP and CCD Sensor
#define RESET_DSP       0x02  // Reset ISP and CCD Sensor
#define RESET_CCD       0x03  // Reset CCD Sensor Only.
```

Note: Those command are for factory use (e.g. firmware upgrade) mainly, it's **NOT** recommended to user these ***Reset*** commands in user's application.

. Set Camera Resolution (commandID = 0x60)

→ 0x60 7 RowSize(MSB,LSB) ColumnSize(MSB,LSB) BinMode BufferCount BufferOption

Host can set ROI (Region Of Interesting), 1:2/1:3/1:4 Bin/Skip mode, Camera Buffer Count and Camera Buffer Option via this command.

Note:

1). RowSize and ColumnSize should be set porperly according to the camera type, For existing CCD Cameras, we have the following available settings:

***. CCX/CXX Modules**

1.3M CCN/CCE/CXN/CXE Mono Modules: (CCN-B013-U, CCE-B013-U, CXN-B013-U and CXE-B013-U)

BinMode:

0 : No Bin/Skip, it's 1394x1040 as full resolution, ROI is only possible on column size:
0x81 : 1:2 Bin mode, it's pre-defined as: 1392 x 520
0x82 : 1:3 Bin mode, it's pre-defined as: 1392 x 344
0x83 : 1:4 Bin mode, it's pre-defined as: 1392 x 256
0x03 : 1:4 Skip mode, it's pre-defined as: 1392 x 256

1.3M CCN/CCE/CXN/CXE Color Modules: (CCN-C013-U, CCE-C013-U, CXN-C013-U and CXE-C013-U)

BinMode:

0 : No Bin/Skip, it's 1392x1040 as full resolution, ROI is only possible on column size:
0x03 : 1:4 Skip mode, it's pre-defined as: 1392 x 256
Note: for C013 Camera, Bin modes (1:2Bin, 1:3Bin and 1:4Bin) are not supported.

2M CCN/CCE Mono Modules: (CCN-B020-U, CCE-B020-U)

BinMode:

0 : No Bin/Skip, it's 1616x1232 as full resolution, ROI is only possible on column size:
0x81 : 1:2 Bin mode, it's pre-defined as: 1616 x616
0x82 : 1:3 Bin mode, it's pre-defined as: 1616 x410
0x83 : 1:4 Bin mode, it's pre-defined as: 1616 x 308
0x03 : 1:4Skip mode, it's pre-defined as: 1616 x 308

2M CCN/CCE Color Modules: (CCN-C020-U, CCE-C020-U)

BinMode:

0 : No Bin/Skip, it's 1616x1232 as full resolution, ROI is only possible on column size:
0x03 : 1:4 Skip mode, it's pre-defined as: 1616 x 308
Note: for C020 Camera, Bin modes (1:2Bin, 1:3Bin and 1:4Bin) are not supported.

The ROI is only possible while BinMode is "0", please also note that the Row Size is actually fixed (to 1392 for X013 cameras and 1616 for X020 cameras) in all cases, actually only the ColumnSize can be set to a customized number when BinMode is 0, but it must be multiples of 8. (e.g. 64, 128…etc.)

It's important to set Row Size, Column Size and Bin Mode correctly according to above available options to the Camera.

2). Buffer Count should not exceed Max buffer count (maximum is 8 for CCX cameras), it's important for setting the buffer count, if the buffer count is too small, the frame rate might be affected, as device might have to ignore frames while the buffer is full (and host is not fetching frames on time). While setting buffer count too big, it's maximum optimized for frame rate, but frame delay might be an issue. The setting of buffer count is very dependent on the Host bandwidth.
3). Buffer Option should always be 0 currently.

Example1: Set camera to 1392x1040 (default), No Bin/Skip mode, 4 buffer count, we have:
→ 0x60 7 5 70 4 10 0 4 0
Here, (5 70) means 0x570, which is 1392 decimal, (4 10) means 0x410, which is 1040.

Example2: Set camera to 1392x480 (ROI), No Bin/Skip mode, 4 buffer count, we have:
→ 0x60 7 5 70 1 E0 0 4 0
Here, (5 70) means 0x570, which is 1392 decimal, (1 E0) means 0x1E0, which is 480

Example3: Set camera to 1392x256, 1:4 Bin, 4 buffer count, we have:
→ 0x60 7 5 70 1 0 83 4 0
Here, (5 70) means 0x570, which is 1392 decimal, (1 0) means 0x100, which is 256. Bin Mode is 0x83, which means 1:4 Bin mode.

*. CGX Modules

CGN/CGE Mono Modules: (CGN-B013-U and CGE-B013-U)
CGN/CGE Color Modules: (CCN-C013-U and CCE-C013-U)

BinMode:
0 : No Bin/Skip, it's 1280x960 as full resolution, ROI is only possible on column size:
0x81 : 1:2 Bin mode, it's pre-defined as: 1280 x 480
0x82 : 1:3 Bin mode, it's pre-defined as: 1280 x 320
0x83 : 1:4 Bin mode, it's pre-defined as: 1280 x 240
0x03 : 1:4 Bin mode2, it's pre-defined as: 1280 x 240
For Bin mode, the pixel is the sum of the bin area (e.g. for 1:2 bin, the pixel value is the sum of the 2x2 area, which is actually 4 pixels, it's Row Bin and Column Bin), the Bin mode2 is lightly different, it's sum of the first column pixel of the rows of the bin area, thus it's sum of 2 pixels for a 2x2 area (in 1:2 bin mode), so it's Row Bin and Column Skip.
Note that for CGN and CGE C013 modules (color camera), we have:
1). Color information is NOT present in the frame data any more while it's set to Bin mode.

So, the ROI is only possible while BinMode is "0", please also note that the Row Size is actually fixed (to 1280) in all cases, actually only the ColumnSize can be set to a customized number when BinMode is 0, but it must be multiples of 8. (e.g. 64, 128…etc.)
It's important to set Row Size, Column Size and BinMode correctly according to above available options to the Camera.

2). Buffer Count should not exceed Max buffer count (maximum is 24 for CGX cameras), it's important for setting the buffer count, if the buffer count is too small, the frame rate might be affected, as device might have to ignore frames while the buffer is full (and host is not fetching frames on time). While setting buffer count too big, it's maximum optimized for frame rate, but frame delay might be an issue. The setting of buffer count is very dependent on the Host bandwidth.
3). Buffer Option should always be 0 currently.

Example1: Set camera to 1280x960 (default), No Bin/Skip mode, 4 buffer count, we have:
→ 0x60 7 5 0 3 C0 0 4 0
Here, (5 0) means 0x500, which is 1280 decimal, (3 C0) means 0x3C0, which is 960.

Example2: Set camera to 1280x480 (ROI), No Bin/Skip mode, 4 buffer count, we have:
→ 0x60 7 5 0 1 E0 0 4 0

Here, (5 0) means 0x500, which is 1280 decimal, (1 E0) means 0x1E0, which is 480

Example3: Set camera to 1280x240, 1:4 Bin, 4 buffer count, we have:
→ 0x60 7 5 0 0 F0 83 4 0
Here, (5 0) means 0x500, which is 1280 decimal, (0 F0) means 0xF0, which is 240. Bin Mode is 0x83, which means 1:4 Bin mode.

****. Set Camera (X,Y) Start Points (commandID = 0x61)***
→ 0x61 4 XStartMSB XStartLSB YStartMSB YStartLSB

Host can set Start position of the ROI at Bin Mode 0, On the other hand, X Start must be 0 as the row size in all cases is fixed (e.g. for B013 camera, it's 1392). The Y Start must be in a valid range according to the current ROI setting, and it must be a value of multiple of 8 too.

****. Set Camera Gains (commandID = 0x62)***
→ 0x62 3 Rgain Ggain Bgain

Host can set Gains for Red, Green and Blue pixels on sensor. (Currently, For Mono and Color cameras, only the Green Gain is used by camera, but it's recommended to set the other two other gains the same value as Ggain, for Color camera, the Green Gain works as "Global" gain for R, G and B, thus if user want to have different Gains for R, G and B, **user have to have "digital" gain for R and B after getting the Bayer form image data from the camera,** this might be necessary if user want to have White balance feature). The Gain value should be in 6 – 41dB (inclusive). Device will check the validity of the input gain value and set proper gain if the data in command is out of the range. (E.g. if gain value is more than 41, device will set the gain to 41).
Note:
1). For setting proper exposure for an image, it's recommended to adjust exposure time prior to the gain, as setting high gain will increase the noise (Gain is similar to the ISO settings on consumer camera), for applications which the SNR is important, it's recommended to set Gain not more than 16dB.
2). For **CCX** modules, although the minimum Gain is 6dB, user might have to set it to 14dB when the camera is NOT in BIN mode. With the current hardware/firmware design, the CCD output (Sony ICX205 and ICX274) is only up to ~0.45V as its saturation voltage, even with 6dB gain (2x), it's ~0.9V signal, while the CCD processor is with a 2V reference ADC, only set the Gain to 14dB will let the ADC generate full range data. Here, we leave this feasibility to users, In some cases, user might still want to set Gain to 6dB to get optimized SNR (rather than the ADC range). In most of the applications, the **Minimum Gain** recommended for CCX modals is as following:
No Bin mode (Bin = 0, or Bin = 0x03<Skip mode>), Gain = 14 (dB)
1:2 Bin mode (Bin = 0x81), Gain = 8 (dB)
1:3 Bin mode (Bin = 0x82), Gain = 6 (dB)
1:4 Bin mode (Bin = 0x83), Gain = 6 (dB)

3). For **CGX** modules, although the minimum Gain is 6dB, user might have to set it to 15dB when the camera is NOT in BIN mode. With the current hardware/firmware design, the CCD output (Sony ICX445) is only up to 0.38V as its saturation voltage, even with 6dB gain (2x), it's ~0.76V signal, while the CCD processor is with a 2V reference ADC, only set the Gain to 15dB will let the ADC generate full range data. Here, we leave this feasibility to users, as in some cases, user might still want to set Gain to 6dB to get optimized SNR (rather than the ADC range). In most of the applications, the **Minimum Gain** recommended for CGX-B013-U/CGX-C013-U is as following:
No Bin mode (Bin = 0) , Gain = 15 (dB) for B013 module and 17(dB) for C013 module.
1:2 Bin mode (Bin = 0x81), Gain = 8 (dB)
1:3 Bin mode (Bin = 0x82), Gain = 6 (dB)
1:4 Bin mode (Bin = 0x83), Gain = 6 (dB)

1:4 Bin mode2 (Bin = 0x03), Gain = 8 (dB)

3). For **CXX** modules, although the minimum Gain is 6dB, user might have to set it to 9dB when the camera is NOT in BIN mode. With the current hardware/firmware design, the CCD output (Sony ICX285) is only up to 0.8V as its saturation voltage, even with 6dB gain (2x), it's ~1.6V signal, while the CCD processor is with a 2V reference ADC, only set the Gain to 9dB will let the ADC generate full range data. Here, we leave this feasibility to users, as in some cases, user might still want to set Gain to 6dB to get optimized SNR (rather than the ADC range). In most of the applications, the **Minimum Gain** recommended for CXX-B013-U is as following:
No Bin mode (Bin = 0), or Bin=0x03<skip mode>. Gain = 9 (dB).
1:2 Bin mode (Bin = 0x81), Gain = 6 (dB)
1:3 Bin mode (Bin = 0x82), Gain = 6 (dB)
1:4 Bin mode (Bin = 0x83), Gain = 6 (dB)

****. Set Camera Exposure Time (commandID = 0x63)***

→ 0x63 4 MSB Byte2 Byte3 LSB (of Exposure Time)

Host can set Exposure Time of the device with this command. The four bytes Data are the MSB to LSB of the new exposure time (32bit integer), note that the unit of the exposure time is 0.05ms, for example, if setting exposure time to 5ms, the value will be 100 (100x0.05ms = 10ms), so the command is:

→0x63 2 0x00 0x00 0x00 0x64

Note: the Exposure time range is 0.05ms – 200,000ms (1 – 4000,000 for the set value)

****. Set Camera Frame Time (commandID = 0x64)***

→ 0x64 2 MSB LSB (of Frame Time)

Host can set Frame Time of the device with this command. The two bytes Data are the MSB and LSB of the new frame time (16 bit word), note that the unit of the frame time is 0.1ms, for example, if setting frame time to 100ms (which means the frame rate is 10fps), the value will be 1000 (1000x0.1ms = 100ms), so the command is:

→0x64 2 0x03 0xE8

Note: Camera will do "best effort" to set frame time precisely, however, the actual frame time is affected by other factors such as sensor's maximum frame rate and current exposure time…etc.

End points 2 (IN)

End point2 is used for fetching image data only, before reading data from it, please make sure Command 0x34 is sent successfully (please refer to the command 0x34 description, and before command 0x34, command 0x33 is usually to be issued to get the available frame count).
After command 0x34 is sent and proper buffer is arranged, host can read data from EP2(IN), device will return the data in the following format for each frame:

```
Typedef struct
{
   tUINT16 Row;
   tUINT16 Column;
   tUINT16 Bin;
   tUINT16 XStart;
   tUINT16 YStart;
   tUINT16 RedGain;
   tUINT16 GreenGain;
   tUINT16 BlueGain;
   tUINT16 TimeStamp;
   tUINT16 TriggerEventOccurred;
   tUINT16 TriggerEventCount;
    tUINT16 UserMark;
    tUINT16 FrameTime;
    tUINT16 CCDFrequency;
    tUINT32 ExposureTime;
    tUINT16 Reserved[240];
} tFrameProperty; // Note: Sizeof (tFrameProperty) is 512 byte.

Typedef struct
{
// For 8bit mode, e.g. PixelData[1040][1392] for CCX-B013-U module, PixelData[960][1280] for CGX
// modules.
tUINT8 PixelData[RowNumber][ColumnNumber];
/*
* For 12bit mode, we have the following:
* tUINT8 PixelData[RowNumber][ColumnNumber][2]; //  12 bit camera
* and PixelData[][][0] contains the 8bit MSB of 12bit pixel data, while the  4 LSB of PixelData[][][1] has
* the 4bit LSB of the 12bit pixel data.
*/
tUINT8 Paddings[]; // Depends on different resolution.
tFramePropery  ImageProperty;  // Note: Sizeof (tFrameProperty) is 512 byte.
} tImageFrame;
```

The Row and Column is depending on the resolution (and bin mode setting). Please refer to the resolution command for Row and Column value for each resolution index setting.
The Paddings[] are padding bytes if the PixelData[Row][Column] is not 512byte block alignment, for example, in 1392x1040 (default) resolution, PixelData[1040][1392] is 1447680 bytes, it's NOT 512byte block alignment, in this case, Paddings[256] is used… while in some of the resolution settings, no padding is needed.(e.g. 12bit mode for 1392x1040)

Note:
1). The above structure is only for one frame. For multiple frame fetching (the frame count in command 0x34 is more the ONE), the totally data fetching should be: sizeof(tImageFrame) x FrameCount

There’s no gap between frames.
2). For PixelData value, it’s CCD Sensor dependant, for Monochrome sensor, they’re ADC value for each pixel. For Color sensor, note that there’s a bayer filter on sensor and the format for rows are:
1st Row → G R G R G R….
2nd Row → B G B G B G….
….
For details, user might refer to sensor’s specification.
3). For the image property fields (the fields in tFrameProperty), please refer to the SDK manual for the description of them, briefly, these are parameters for the camera used to catch THIS frame.
4). The data stream from EP2 is in Byte, the above “tUINT16” and “tUINT32” (items in image property) is in little endian format.