Allied Vision

APPLICATION NOTE

RS-232 Port on Allied Vision GigE Cameras

2016-Mar-18

Introduction

Allied Vision GigE cameras can send and receive serial commands using its RS-232 port. Tx and Rx data lines are available via the Hirose port on the back of the cameras. Enabling and controlling serial communication is performed by writing to serial IO registers on the camera through the PvAPI GigE SDK (see GigESDK/ examples/siotest source code), or through the serial dialog window in the SampleViewer.

Requirements

- Allied Vision GigE camera with an open ended Hirose connector allowing access to RS-232 pins
- 9-pin DSUB connector for connection to host PC
- Computer with a working serial port, or USB to serial cable
- Terminal application which allows serial communication (HyperTerminal, etc)

Hirose to DSUB wiring

Location of RS-232 RxD, RS-232 TxD, and GND pins varies between models. Naming of correct GND pin is not consistent between models. Wire as follows:

	RxD - Pin	**TxD - Pin**	**GND - Pin**
Manta	8	9	1 – Camera Ground
Prosilica GB	11	10	14 – Signal Ground
Prosilica GC	9	8	10 – Signal Ground
Prosilica GE	4	5	10/11/12 – Isolated Ground
Prosilica GT	8	9	1 – Camera Ground
Prosilica GX	8	9	1 – Camera Ground

See the camera Technical Manual for reference. See figure 1 for 9 Pin DSUB wiring.

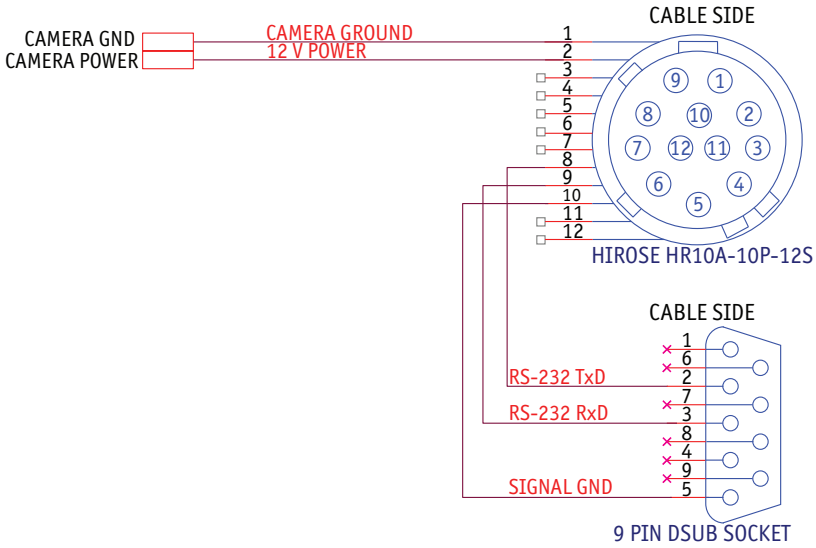

Figure 1: *RS-232 wiring diagram. Hirose plug is GC series wiring only*

Connect the Allied Vision GigE camera to the host PC, ensuring it is recognized by the GigE network and that the RS-232 lines are connected to your computer serial port.

Terminal setup

Open your terminal application, settings:

Baud Rate	9600 (Manta and Prosilica GT: 115200)
Data Bits	8
Stop Bits	1
Parity	None
Flow control	None
COM	Whichever is used by host (typically COM1)

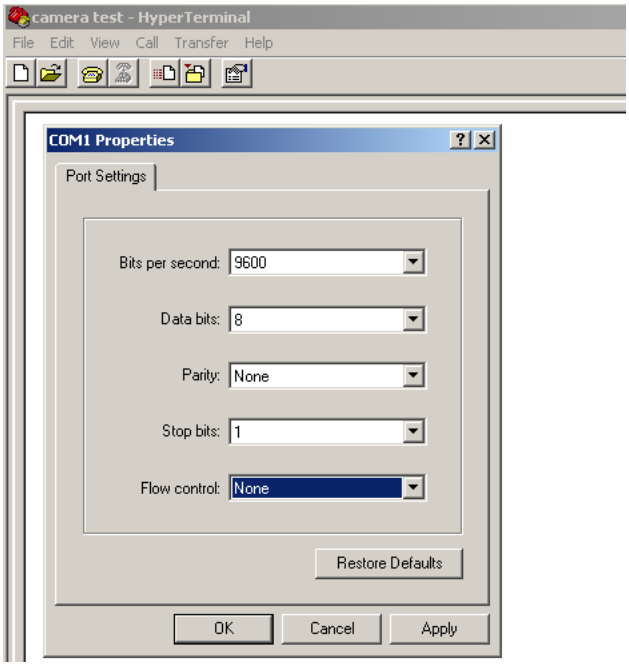

Figure 2: *Terminal settings. Note: Manta and GT series uses Baud 115200*

SampleViewer test

1. Open the SampleViewer, and click on the **IIOI** icon. Set Baud rate, parity, character length, stop bits to same as terminal settings. Click **Connect**.
2. Type in the SampleViewer RS-232 dialog window and click send. The text will appear in terminal application. This verifies camera TxD communication.
3. Type in the terminal application and hit enter. The text will appear in SampleViewer. This verifies camera RXD communication.

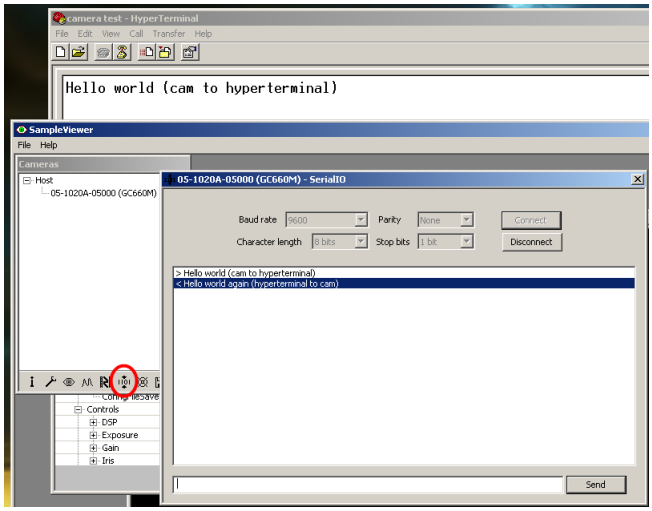

Figure 3: *Testing RS-232 in SampleViewer*

Siotest example code test

- Compile the siotest example code, and run the resulting .exe in a windows command prompt.
- You will see a test pattern of integers appear in the terminal window. This verifies camera TXD communication.
- Type in the terminal window, you will see your typed input appear in the windows command prompt. This verifies camera RXD communication.

Camera register map

The following camera registers are used for RS-232 communication. See the siotest example code for more on reading/writing to camera registers.

Address	Name	Bits	Description
16000h	SerialIoInquiry	[0]	[R] Serial IO transmitter is available
		[1]	[R] Serial IO receiver is available
		[2]	[R] Serial IO receiver has timestamp mode
		[other]	[R] Reserved. All zeros
16010h	SerialIoErrorStatus	[other]	[R] Reserved. All zeros
16100h	SerialModeInquiry	[0]	[R] 300 baud supported
		[1]	[R] 600 baud supported
		[2]	[R] 1200 baud supported
		[3]	[R] 2400 baud supported
		[4]	[R] 4800 baud supported
		[5]	[R] 9600 baud supported
		[6]	[R] 19200 baud supported
		[7]	[R] 38400 baud supported
		[8]	[R] 57600 baud supported
		[9]	[R] 115200 baud supported
		[10]	[R] 230400 baud supported
		[16]	[R] No parity supported
		[17]	[R] Odd parity supported
		[18]	[R] Even parity supported
		[20]	[R] Character length 5 bits supported
		[21]	[R] Character length 6 bits supported
		[22]	[R] Character length 7 bits supported
		[23]	[R] Character length 8 bits supported

Address	Name	Bits	Description
		[24]	[R] 1 stop bit supported
		[25]	[R] 1.5 stop bits supported
		[26]	[R] 2 stop bits supported
		[other]	[R] Reserved. All zeros
16104h	SerialMode	[7..0]	[RW] Baud rate: 0: 300 1: 600 2: 1200 3: 2400 4: 4800 5: 9600 6: 19200 7: 38400 8: 57600 9: 115200 10: 230400
		[9..8]	[RW] Parity: 0: None 1: Odd 2: Even
16104h	SerialMode	[11..10]	[RW] Character length: 0: 5 bits 1: 6 bits 2: 7 bits 3: 8 bits
		[13..12]	[RW] Stop bits: 0: 1 Stop bit 1: 1.5 Stop bits 2: 2 Stop bits
		[other]	[R] Reserved. All zeros
16120h	SerialTxInquiry	[15..0]	[R] Transmitter buffer size, in bytes
		[other]	[R] Reserved. All zeros
16124h	SerialTxStatus	[0]	[R] Transmitter ready
		[other]	[R] Reserved. All zeros
16128h	SerialTxControl	[0]	[W] Transmitter reset when 1. Occurs immediately [R] Always zero
		[1]	[RW] Transmitter enable, when 1
		[other]	[R] Reserved. All zeros

Address	Name	Bits	Description
1612Ch	SerialTxLength	[15..0]	[W] Transmit data length, in bytes. When this is written, the data is *SerialTxBuffer* is sent through the serial port. *SerialTxLength* mat not written if *"Transmitter Ready"* is zero. [R] Always zeros
		[other]	[R] Reserved. All zeros
16140h	SerialRxInquiry	[15..0]	[W] Receiver buffer size, in bytes
		[other]	[R] Reserved. All zeros
16144h	SerialRxStatus	[0]	[R] Receive overrun [W] Write 1 to clear
		[1]	[R] Receive framing error [W] Write 1 to clear
		[2]	[R] Receive parity error [W] Write 1 to clear
		[other]	[R] Reserved. All zeros
16148h	SerialRxControl	[0]	[W] Receiver reset when 1. Occurs immediately [R] Always zero
		[1]	[RW] Receiver enable, when 1
		[2]	[RW] Enable timestamp mode. See *SerialRxBuffer* for a description of timestamp mode. (Do not change this bit on the fly, otherwise some data will timestamped and some will not.) Check *SerialIoInquiry* to see if this feature is available
		[other]	[R] Reserved. All zeros
1614Ch	SerialRxLength	[15..0]	[R] Number of bytes in the receive buffer. [W] Number of bytes read from receive buffer. The counter (see [R] above) is decremented by this amount.

Address	Name	Bits	Description
16400h	SerialTxBuffer		[W] Transmit buffer. Write your data into the buffer, then write your data length into *SerialTxLength* to begin transmission. Each serial word is stored as a byte, LSBit aligned. The bytes are packed into 32-bit registers; the MSByte of each register is the first serial-word transmitted.(When the data length is not a multiple of 4, trailing bytes are ignored.) ex. write 0x41424344 to output "ABCD" *SerialTxBuffer* may not be written if *"Transmitter Ready"* is zero
16800h	SerialRxBuffer		[R] Receive buffer. Read data from this buffer. Read *SerialRxLength* for the number of valid bytes in this receive buffer. After reading the data, you must write the length of your read to *SerialRxLength*. See *SerialTxBuffer* for data packing. Timestamp mode: each received byte is proceeded by 8 timestamp bytes. MSB first. This timestamp is the frame timestamp. ex. Oxl1223344AA6BCCDD41 is "A" received at timestamp 0x11223344AABBCCDD

Additional References

Technical manuals and GigE feature reference
https://www.alliedvision.com/en/support/technical-documentation

For technical support, please contact support@alliedvision.com.
For comments or suggestions regarding this document, please contact info@alliedvision.com.

Disclaimer

Due to continual product development, technical specifications may be subject to change without notice. All trademarks are acknowledged as property of their respective owners. We are convinced that this information is correct. We acknowledge that it may not be comprehensive. Nevertheless, Allied Vision cannot be held responsible for any damage in equipment or subsequent loss of data or whatsoever in consequence of this document.

For the latest version of this document, please visit the Allied Vision documentation website.