ALLIED VISION TECHNOLOGIES CANADA Inc.
101 – 3750 North Fraser Way | Burnaby, B.C. | Canada
www.alliedvisiontec.com

Application Note:

RS-232 Port on AVT GigE Cameras

AVT GigE cameras can send and receive serial commands using its RS232 port. Tx and Rx data lines are available via the Hirose connection 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 AVT SampleViewer.

Requirements

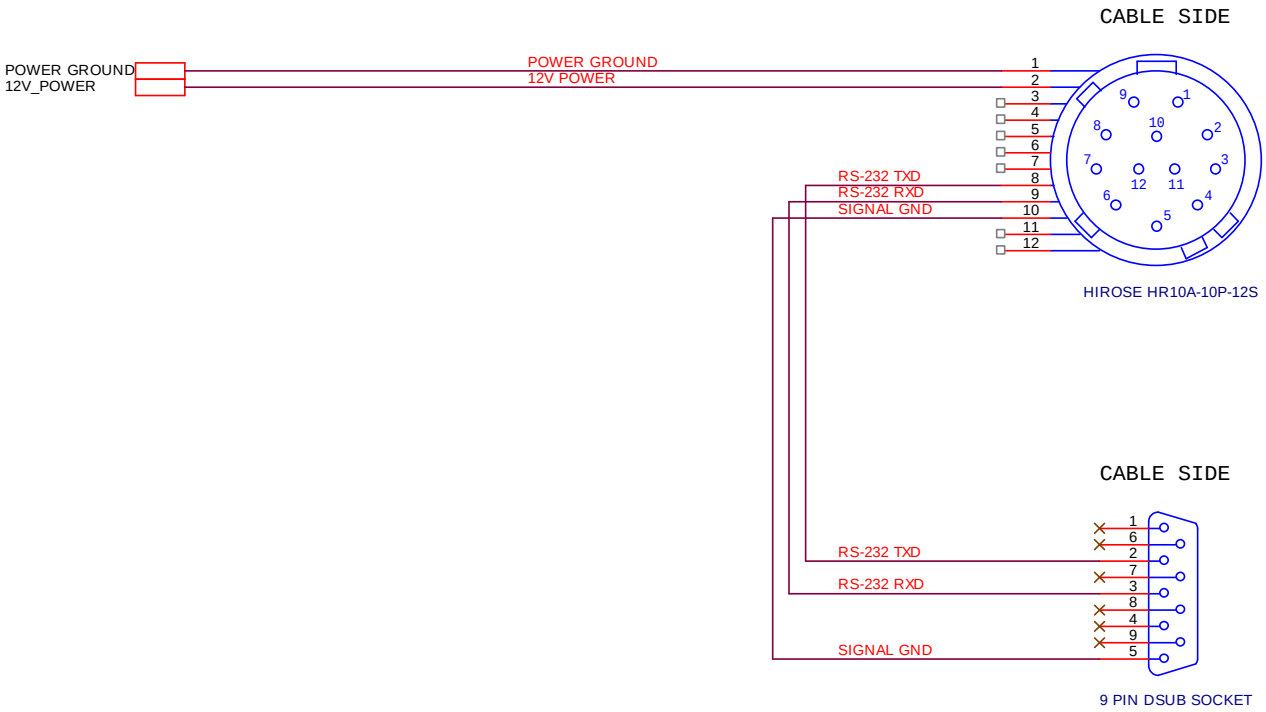
- AVT GigE Camera with an open ended Hirose connector allowing access to RS-232 pins.
- A 9 pin DSUB connector for connection to host computer.
- A computer with a working serial port, or USB to Serial cable.
- A terminal application which allows serial communication (HyperTerminal, etc).

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:

- GC: RxD – Pin 9. TxD – Pin 8. GND = Signal Ground – Pin 10.
- GB: RxD – Pin 11. TxD – Pin 10. GND = GND Power - Pin 9.
- GE: RxD – Pin 4. TxD – Pin 5. GND = Isolated Ground - Pin 10/11/12.
- GX: RxD – Pin 8. TxD – Pin 9. GND = Power Ground – Pin 1.
- Manta: RxD – Pin 8. TxD – Pin 9. GND = External Ground – Pin 1.

See Camera User Manual for reference. See diagram below for 9 Pin DSUB wiring.



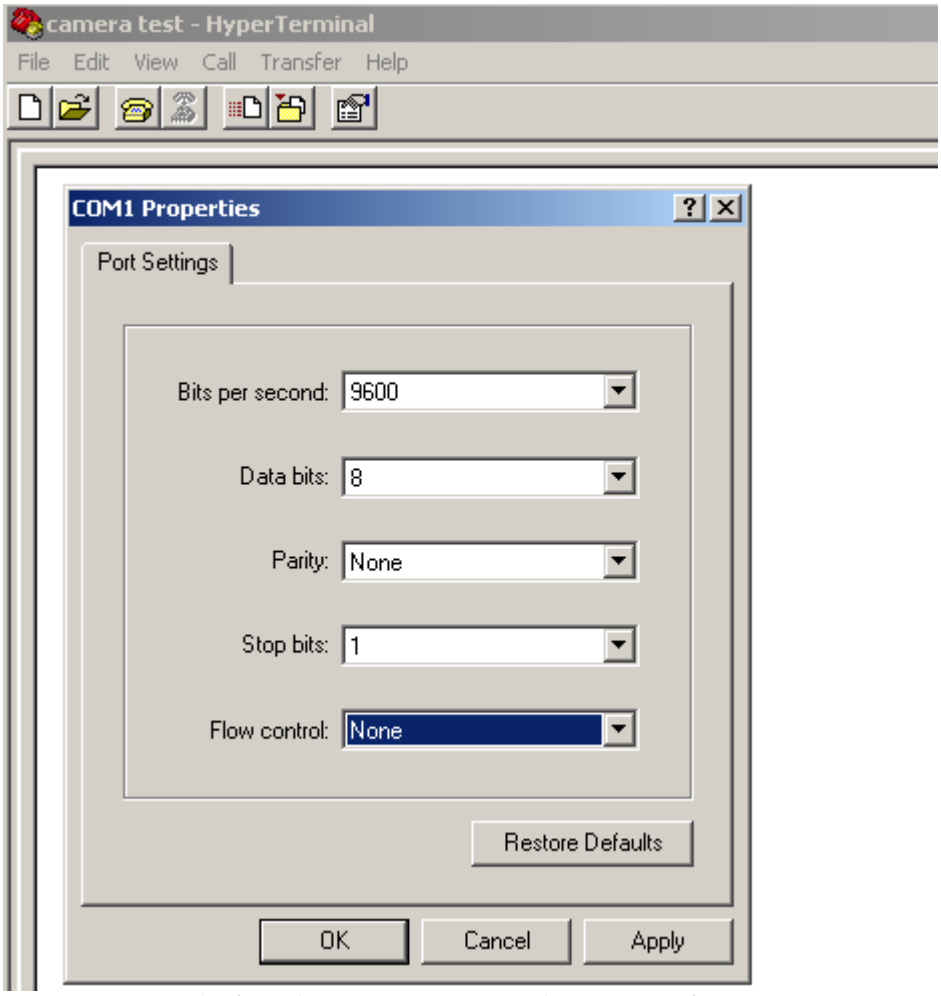
RS-232 Wiring Diagram. Hirose plug is GC series wiring ONLY.

Connect AVT GigE camera to the host PC, ensuring it is recognized by the GigE network and that the RS232 lines are connected to your computer serial port.

Terminal Setup

Open your terminal application, settings:

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



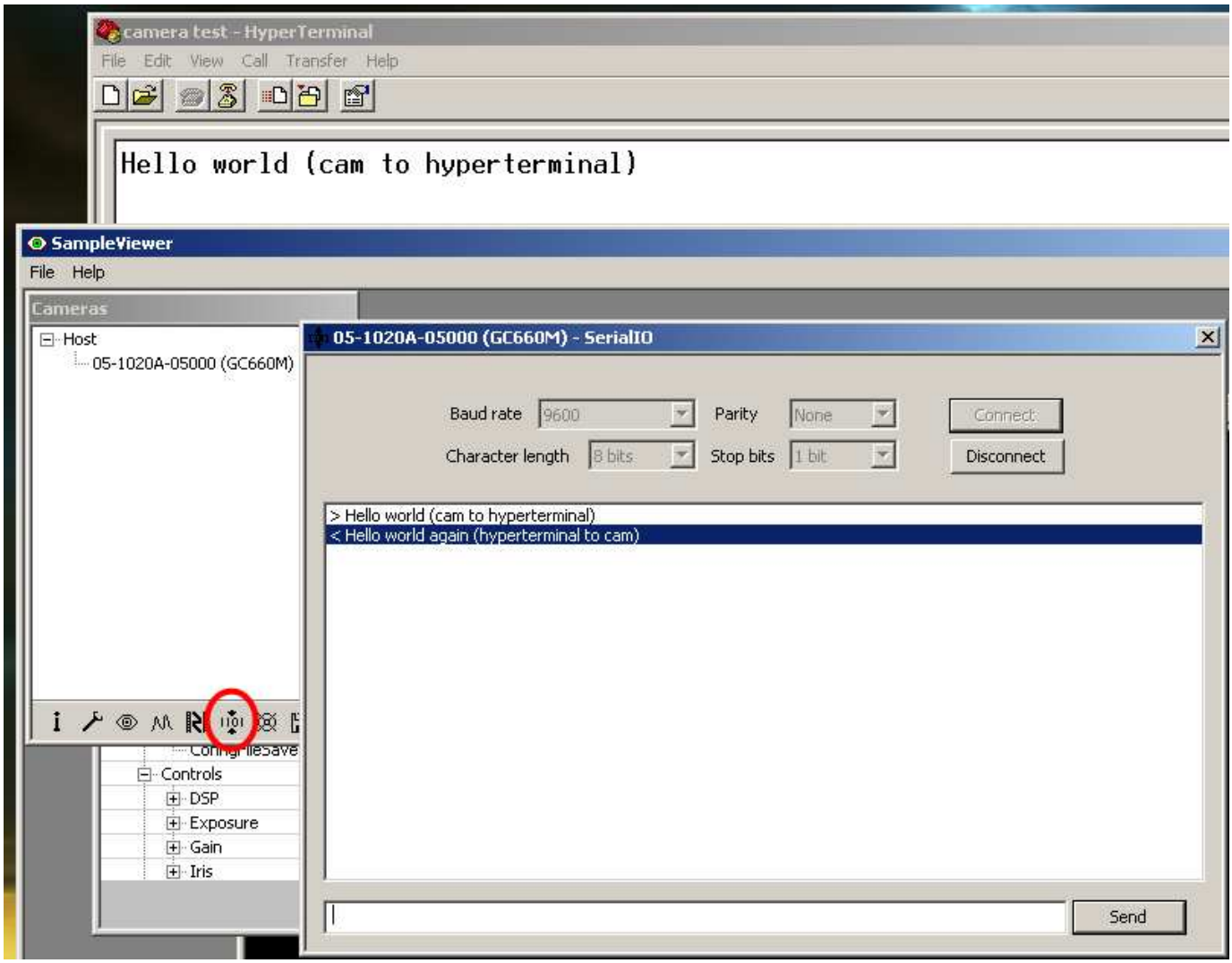
Terminal Settings. Note: Manta series uses Baud 115200.

SampleViewer Test

Open the AVT SampleViewer, and click on the “IIOI” icon. Set Baud rate, parity, character length, stop bits to same as terminal settings. Click “Connect”.

Type in the SampleViewer RS-232 dialog window and click send. The text will appear in terminal application. This verifies camera TXD communication.

Type in the terminal application and hit enter. The text will appear in SampleViewer. This verifies camera RXD communication.



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.

SerialIo Control

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.
		[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
		[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.
1612Ch	SerialTxLength	[15..0]	[W] Transmit data length, in bytes. When this is written, the data in *SerialTxBuffer* is sent through the serial port. *SerialTxLength* may not be written if "*Transmitter Ready*" is zero. [R] Always zeros.
		[other]	[R] Reserved. All zeros.
16140h	SerialRxInquiry	[15..0]	[R] Receiver buffer size, in bytes.

		[other]	[R]	Reserved. All zeros.
16144h	SerialRxStatus	[0]	[R] [W]	Receive overrun. Write 1 to clear.
		[1]	[R] [W]	Receive framing error. Write 1 to clear.
		[2]	[R] [W]	Receive parity error. Write 1 to clear.
		[other]	[R]	Reserved. All zeros.
16148h	SerialRxControl	[0]	[W] [R]	Receiver reset when 1. Occurs immediately. Always zero.
		[1]	[RW]	Receiver enable, when 1.
		[2]	[RW]	Enable timestamp mode. See *SerialRxBuffer* for a description of timestamp mode. (Don't change this bit on the fly, otherwise some data will be timestamped and some will not.) Check *SerialIoInquiry* to see if this feature is available.
		[other]	[R]	Reserved. All zeros.
1614Ch	SerialRxLength	[15..0]	[R] [W]	Number of bytes in the receive buffer. Number of bytes read from receive buffer. The counter (see [R] above) is decremented by this amount.
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. 0x11223344AABBCCDD41 is "A" received at timestamp 0x11223344AABBCCDD

We invite comments or suggestions on this document at any time.
Please write to: support@alliedvisiontec.com