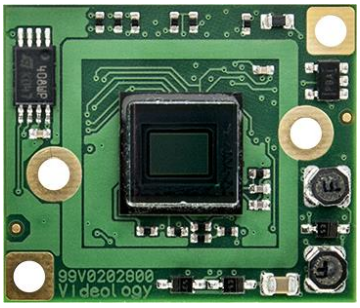
VIDEOLOGY

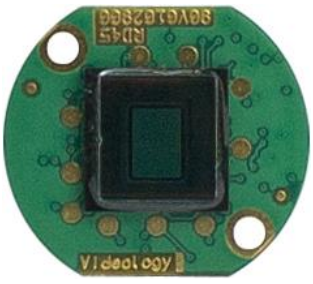
IMAGING SOLUTIONS INC.

Original Equipment Manufacturer

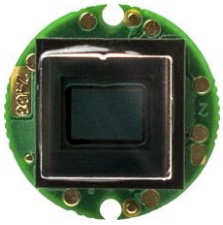
User Manual
SFT-13500 Control Software
USB to I²C Drivers



2XB14X
2XB14XDIG



2XRD45



2XRD45S

Prior to Using This Document: *Videology reserves the right to modify the information in this document as necessary and without notice. It is the user's responsibility to be certain they possess the most recent version of this document by going to www.videologyinc.com, searching for the model number, and comparing revision letters on the respective document, located in the document's footer.*

For technical assistance with this product, please contact the supplier from whom the product was purchased.

Videology® Imaging Solutions, Inc.
37M Lark Industrial Parkway
Greenville, Rhode Island 02828 USA
Tel: (401) 949 – 5332 Fax: (401) 949 – 5276
North/South American Sales: sales@videologyinc.com
www.videologyinc.com

Videology® Imaging Solutions, Europe B.V.
Neutronenlaan 4
5405 NH Uden, The Netherlands
Tel: +31 (0) 413 256261 Fax: +31 (0) 413 251712
European Sales: info@videology.nl
www.videology.nl

License Agreement (Software):
This Agreement states the terms and conditions upon which Videology Imaging Solutions, Inc. USA and Videology Imaging Solutions, B.V. Europe (hereafter referred to as "Videology®") offer to license to you the software together with all related documentation and accompanying items including, but not limited to, the executable programs, drivers, libraries, and data files associated with such software.
The Software is licensed, not sold, to you for use only under the terms of this Agreement.
Videology grants to you, the purchaser, the right to use all or a portion of this Software provided that the Software is used only in conjunction with Videology's family of products.
In using the Software you agree not to:

- Decompile, disassemble, reverse engineer, or otherwise attempt to derive the source code for any Product (except to the extent applicable laws specifically prohibit such restriction);
- Remove or obscure any trademark or copyright notices.

Limited Warranty (Hardware and Software):
ANY USE OF THE SOFTWARE OR HARDWARE IS AT YOUR OWN RISK. THE SOFTWARE IS PROVIDED FOR USE ONLY WITH VIDEOLOGY'S HARDWARE. THE SOFTWARE IS PROVIDED FOR USE "AS IS" WITHOUT WARRANTY OF ANY KIND, TO THE MAXIMUM EXTENT PERMITTED BY LAW, VIDEOLOGY DISCLAIMS ALL WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, QUALITY AND FITNESS FOR A PARTICULAR APPLICATION OR PURPOSE. VIDEOLOGY IS NOT OBLIGATED TO PROVIDE ANY UPDATES OR UPGRADES TO THE SOFTWARE OR ANY RELATED HARDWARE.

Limited Liability (Hardware and Software):
In no event shall Videology or its Licensor's be liable for any damages whatsoever (including, without limitation, incidental, direct, indirect, special or consequential damages, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use this Software or related Hardware, including, but not limited to, any of Videology's family of products.

Table of Contents

1. Document History

Revision	Issue Date	Reason	CN#
A	5/2/2016	Initial release	16-0049

2. Introduction

The purpose of this document is to instruct the user on how to modify the settings and control the Videology camera. The flow will be as follows:

A. Initial Setup

- Installation of software/ files needed
- Hardware setup
- Driver(s) installation

B. Register Modification

- Modify Register settings using the GUI interface
- Modify Sensor settings using the GUI interface

Additional Equipment Needed:

All the necessary hardware has been supplied with the exception of

1. A PC with an available USB port
2. Video Monitor
3. Power supply
4. BNC cable

3. Graphical Overlay creation

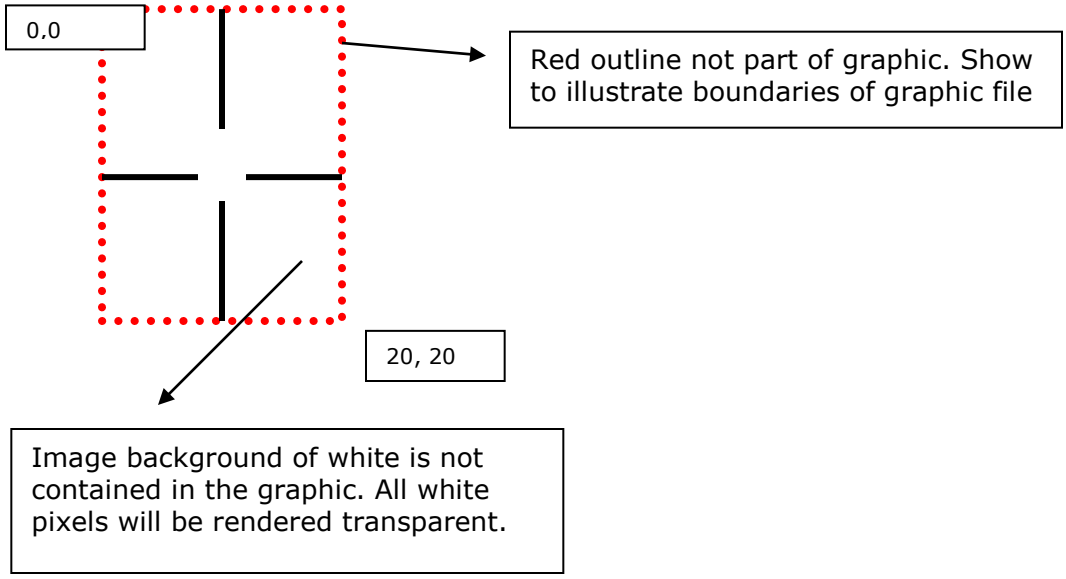
a. Create Bitmap for crosshair overlay

Overlays can be created with any program that will save in a 24-bit .bmp format. The maximum overlay size is 360 x 480 pixels which will be rendered into 720 x 480 pixel display format. What this means is that the overlay in the x-axis will be stretched (pixels will be doubled). This must be kept in mind when creating the artwork for the overlays.

In order to have the maximum flexibility in positioning the graphical overlay, it should be created at location 0, 0 in the drawing package. Further, the background should be the exact size required to contain the image and no larger.

In order for the overlay's background to be rendered transparent (so the camera image can be seen) select a solid color for the background. This color should be unique and not contained in the desired overlay. For example, for a black cross hair, create it on a white background. We then can make all the white pixels in the graphic transparent. For a white crosshair create it on a black background. We then can make all the white pixels in the graphic transparent.

Note that the camera defines the upper left corner of the overlay as (0, 0).



3. Installing the USB to I²C Drivers

1. Plug the USB to I2C cable into the PC

2. Run the executable on the CD, or download zip file at http://www.videologyinc.com/download.htm

   "BD_camControl_1_6.exe"
   NOTE: file name might be slightly different due to revision versions.

4. Installing the Camera Control Software

There is nothing to install.

As long as the executable file "**BD_camControl_1_6.exe**" and the DLL "**vdiicdll.dll**" are located in the same directory, you just need to double click on the executable file to run the program.

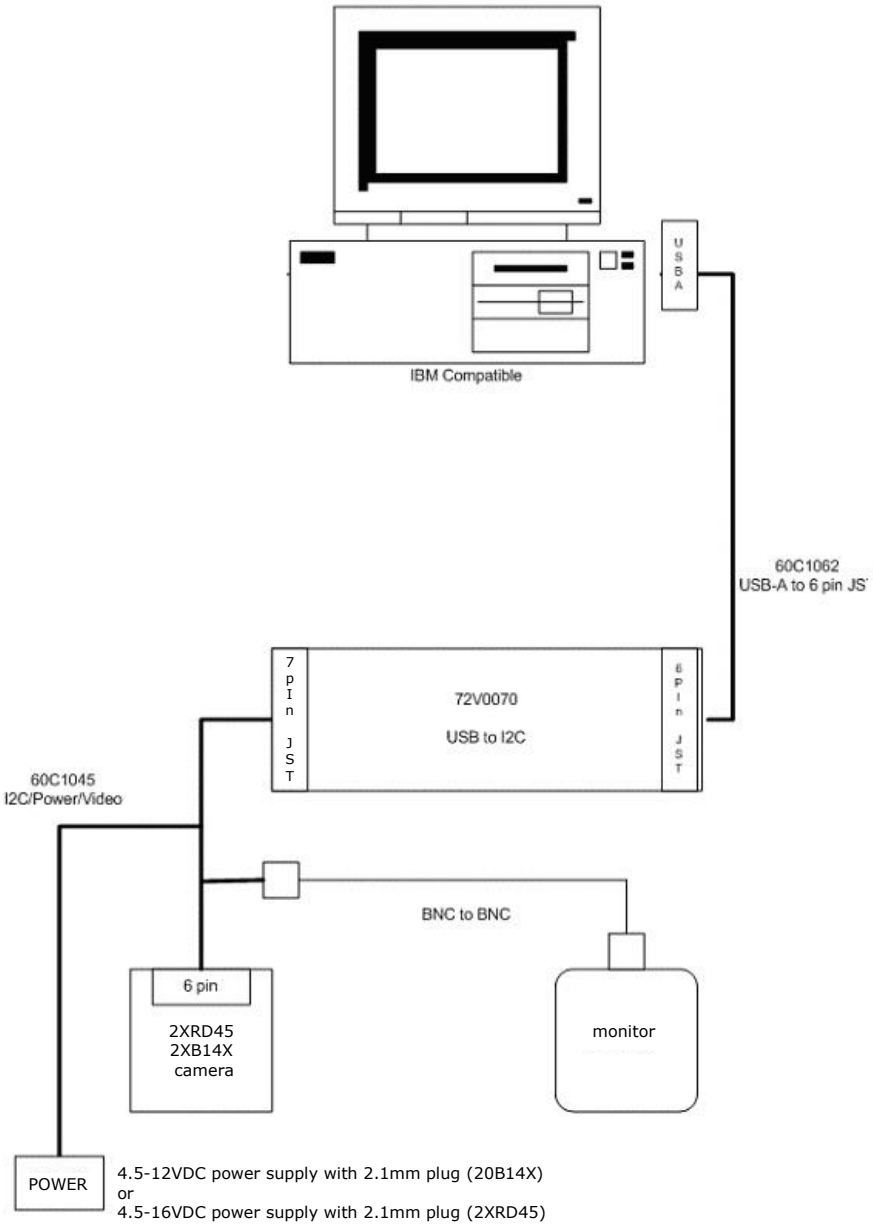
5. Hardware Setup

5.1. Equipment Needed

- Power Supply
  - 20B14X (4.5-12VDC =/- 5%)
  - 20RD45 (4.5-16VDC)
- USB to I2C Kit 60B6-U
  - 60C1062 cable
  - 60C1045 cable
  - 72V0070 pcb
  - SFT-13500
- BNC to BNC cable
- Camera (2XB14X or 2XRD45)

4. A PC with an available USB port
5. Video Monitor

The next figure shows how to assemble the system.

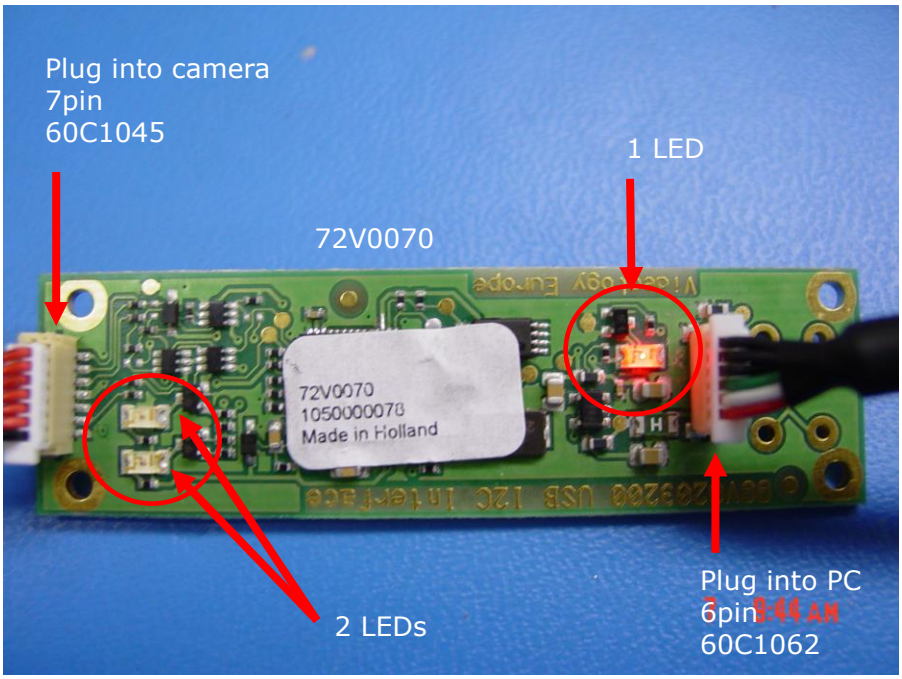


5.2. I²C Board Connections **!IMPORTANT!**

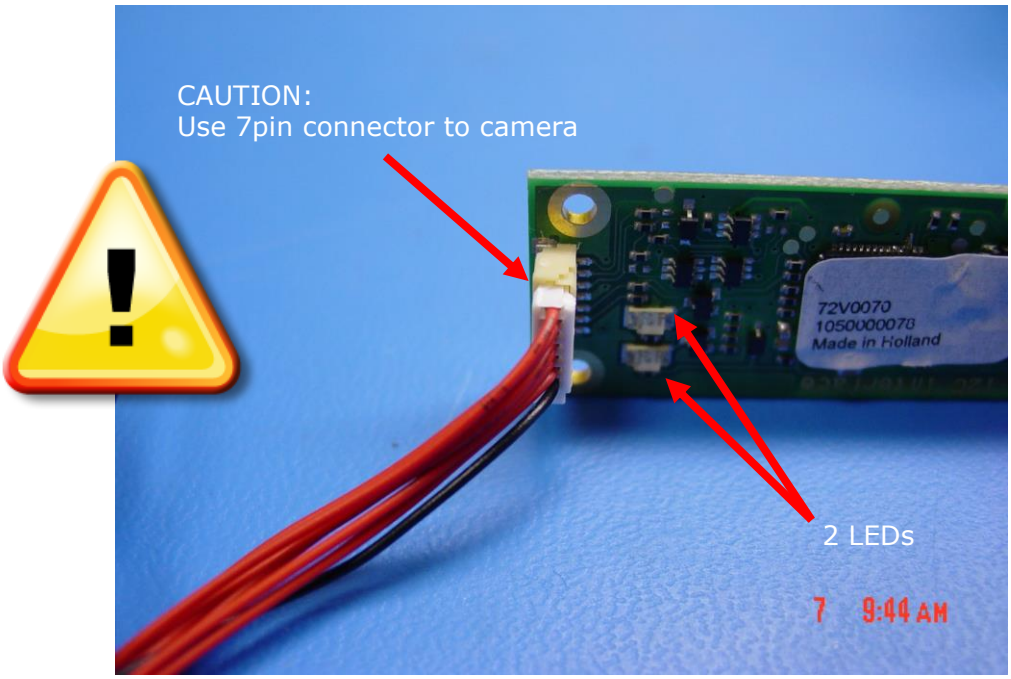
60C1045	7pin cable (connects to camera, 2 LEDs side of I²C board)
60C1062	6pin cable (connects to computer, 1 LED side of I²C board)

Using these figures as reference, follow the steps outlined below:

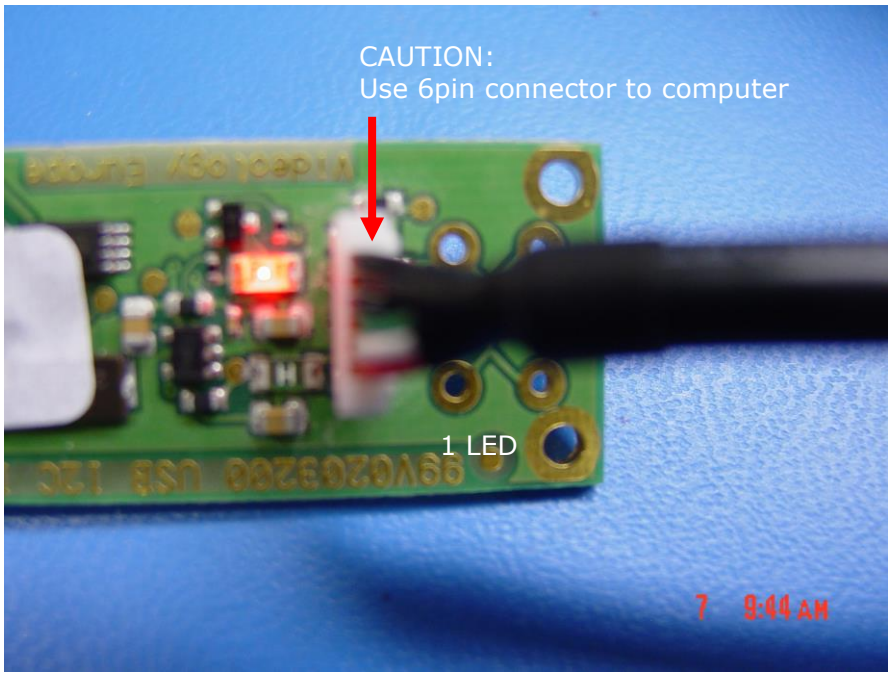
- Lay out the Camera, USB-I²C interface board (72V0070), computer and monitor on your work area
- Locate the 72V0070 USB to I²C Interface board



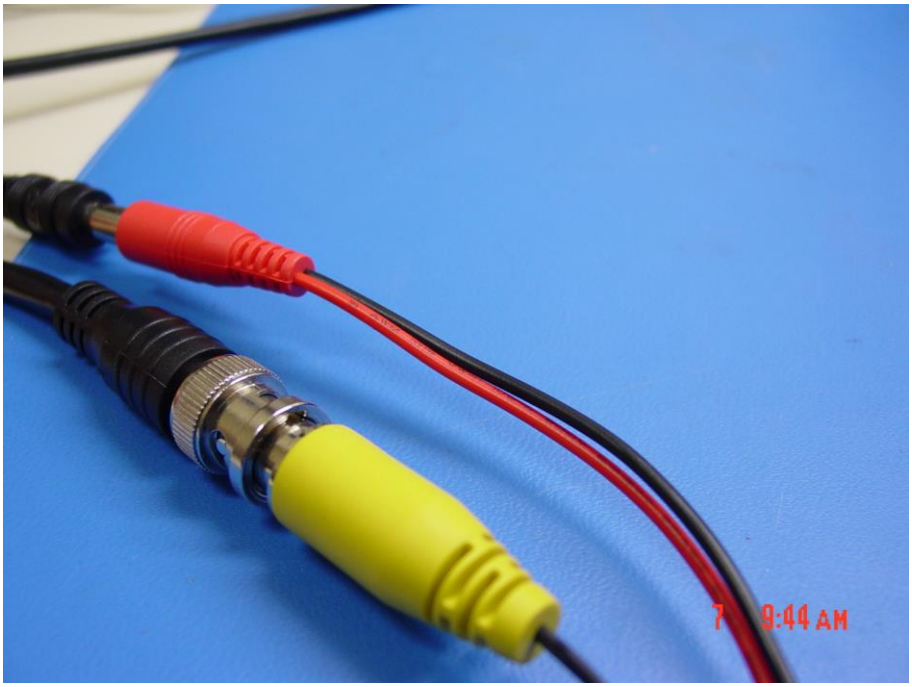
- Connect the 7pin 60C1045 cable to the USB-I²C interface board
  CAUTION: ensure you are **not** using the **6pin** 60c1062 cable for this connector!
  Damage to USB camera may occur!



- Connect the 6pin 60C1062 cable to the USB-I²C interface board
  DO NOT CONNECT THIS TO THE PC YET…

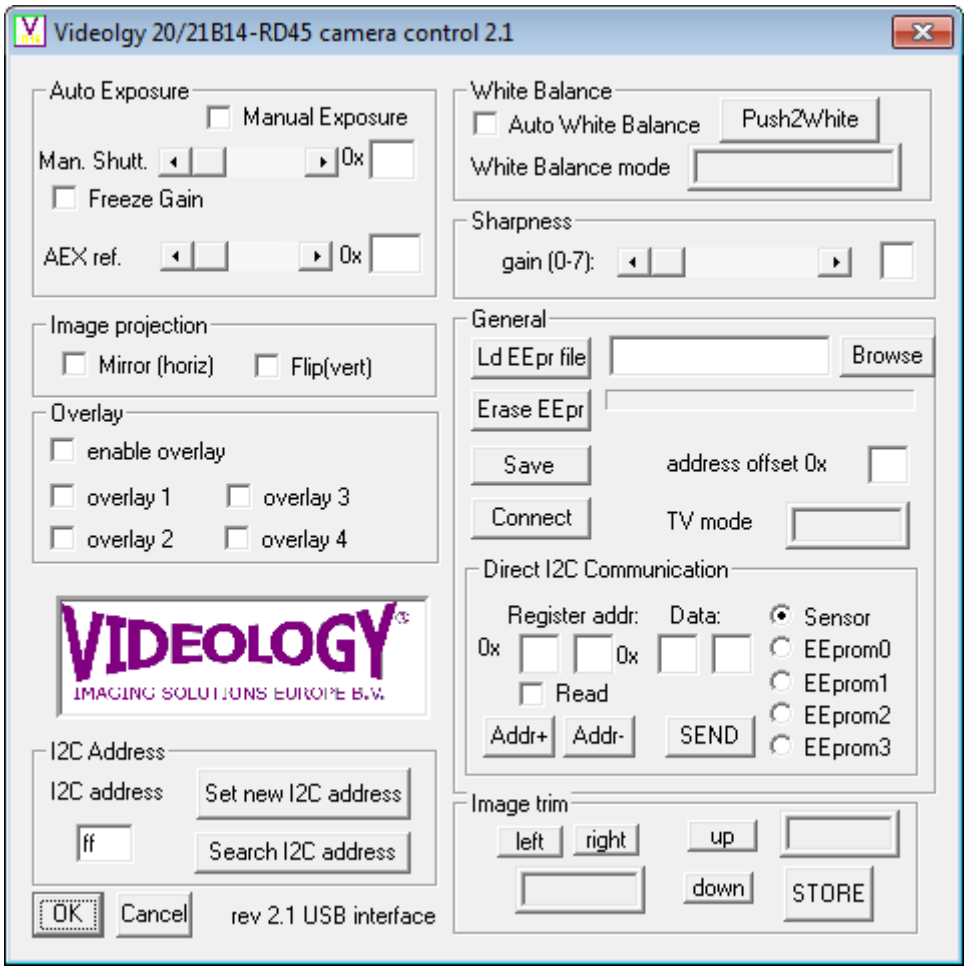


- Connect the 3.3VDC power supply to the 60C1045 cable
- Connect the BNC to BNC cable to the monitor and to the 60C1045 cable



6. Camera Modes

Using the GUI below, camera settings can be modified and stored.
NOTE: Camera power must be cycled to verify settings have been stored correctly.
Note: Revision in title might be different on software you install.



Please note: Videology will not give any warranty in case settings are stored incorrectly, or settings are overwritten!

In the tables below you find per command the required actions to change the camera mode:

Mirror mode:

1. Read register 0x3254
2. Read register 0x301c (mirror status = bit[0] (0=normal, 1=mirror)
3. If mode needs to be changed
    a. Set register 0x301c bit[0] accordingly. All other bits should NOT be changed.

Set register 0x3254 bit[0]. Incase of mirror b[0]=1, in normal mode b[0]=0. All other bits should NOT be changed.

Flip mode:

1. Read register 0x3254.
2. Read register 0x301c (flip status = bit[1] (0=normal, 1=flip)
3. If mode needs to be changed:
    a. Set register 0x301c bit[1] accordingly. All other bits should not be changed.

Set register 0x3254. In case of flip b[1] = 1, in normal mode b[1] = 0;

Auto exposure saturation point

1. Read register 0xa804 to check if the camera is in AEX mode. Value 0x000f is AEX mode, 0x0000 is manual exposure mode.
2. If mode is not correct load register 0xa804 with 0x000f.
3. Read current saturation point from register 0xa812

Load register 0xa812 with the new saturation level data

Shutter speed

1. Read register 0xa804 to check if the camera is in manual mode. Value 0x000f is AEX mode, 0x0000 is manual exposure mode.
2. If mode is not manual load register 0xa804 with 0x0000.
3. Read current shutter value from register 0x3012.

Load register 0x3012 with the new shutter value

Auto white balance mode

1. Read the current white balance mode status from register 0xac04 (if value is 0x00ff than it is Auto white balance mode).

If mode is not correct load register 0xac04 with value 0x00ff.

Freeze mode:

To use this mode the camera must be first put in the freeze mode. The camera first needs to run in AWB mode. It will look for the most optimal white balance setting. When this is reached the AWB must be hold, and preferable the R/B gains must be stored and used when the camera is powered up again.

Put the camera in the freeze mode:

1. Make sure camera is in AWB mode by reading register 0xac04. Value 0x00ff means AWB mode. If the camera is not in the correct mode see table Auto White balance Mode.
2. If the AWB reached it's most optimal position stop the AWB function by loading register 0xac04 with value 0x0000.
3. Wait 10 mSec.
4. Read the following registers: 0xac02, 0xac04, 0xac0a, 0xac0c, 0xac0e, 0xac10, 0xac12, 0xac14, 0xac16, 0xac18, 0xac10, 0xac1a, 0xac1c, 0xac1e, 0xac32, 0xac36, 0xac38, 0xac3a, 0xac3c, 0xac3e, 0xac40, 0xac42, 0xac44, 0xac46, 0xac48, 0xac4a, 0xac4c and 0xac4e. wait 5 mS between each read command.

These values must be stored so that the can be loaded when the camera is powered up.

Enable or disable overlay

In the camera 4 overlay images can be stored. Depending on the situation overlay's can be enabled or disabled. Note that these overlay images must be loaded by Videology. Please contact us for more information.

Global enabling of the overlay's: The camera can have up to 4 images loaded. By disabling the global overlay functions all active overlays will be stopped in one command.

1. Read current status global overlay by reading register 0x4f02. If the MSbit (b[15] is set the global overlay is active. This does not mean that the overlays are visible. Also the individual overlay must be active to make the visible.

Enable the global overlay by loading register 0x4f02 with value 0x8000, or disable global overlay by loading register 0x4f02 with 0x0000.

Enabling individual overlays:
As mentioned above the camera can be loaded with maximum 4 overlay images. This can all be enabled or disabled individually.

1. Each overlay has its own overlay enable address. Overlay1 = 0x4f08, overlay2 = 0x4f0a, overlay3 = 0x4f0c and overlay4 = 0x4f0e. To read the status per overlay read the corresponding address. The MSbit (b[15]) indicates the status. 1 overlay is enabled, 0 overlay is disabled.
2. Set each overlay as required by setting in each register b[15] to the desired enable or disable mode.
3. The lowest two bits indicates which overlay should be loaded. We advise to use the following values for the lowest two bits per register:
    - in reg 0x4f08→b[1,0] = 00
    - in reg 0x4f0a→b[1,0] = 01
    - in reg 0x4f0c→b[1,0] = 10

in reg 0x4f0e→b[1,0] = 3

If settings need to be stored:
You can use the camera's EEprom to store your settings. Please be careful when you do this. The camera has an 8K EEprom on board, which is 4 pages. The last page is meant to store certain settings. But this page may contain settings already stored from the factory! You should not overwrite the factory settings, since this may influence the start up and final behavior of the camera!

To find out if settings are loaded , read from the EEProm. Settings are stored in groups of 4 bytes (two register address bytes and two data bytes). Keep reading until you find a group of 4 bytes with all 0xff. Here you should start storing your values. **Do not leave unused spaces**!!!
In figure two you find a graphical example:

addr	0x0	0x 1	0x 2	0x 3	0x 4	0x 5	0x 6	0x 7	0x 8	0x 9	0xa
data	D1	D2	D3	D4	0xff	0xff	0xff	0xff	0xff	0xff	0xff

First address to load your settings

Figure 1.

Note that you always should store 4 bytes in the following order: Reg Addr MSB, Reg.Addr LSB, Data MSB and Data LSB. So in the case of figure two the MSB of the register address must be stored in register 0x04.

Please be very careful when you store settings. We recommend using Videology's **2XB45 camera control software**.

***Please note**: Videology will not give any warranty in case settings are stored incorrectly, or settings are overwritten!*

The Eeprom has a different I2C format. The register addresses and data addresses are only one byte. The device address for the EEProm page 4 is 0xa6 (write) and 0xa7 (read).

7. Camera Control for 2xB14X and 2XRD45(S) cameras

7.1. Connect

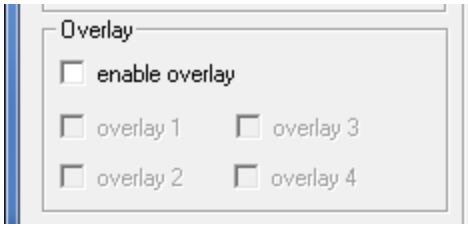
In case the camera was not connected yet when the software was started, or a communication error occurred you can re-connect to the camera by pushing the connect button.

The tool will load the configuration from the camera and will set the buttons and check boxes accordingly.

7.2. Overlay Functions

The camera can be loaded with maximal 4 different overlay graphics. Please note that they must be loaded inside the camera during the production process. For more details please contact your Videology customer service representative.

To control the overlay graphics you can click one of the check boxes:

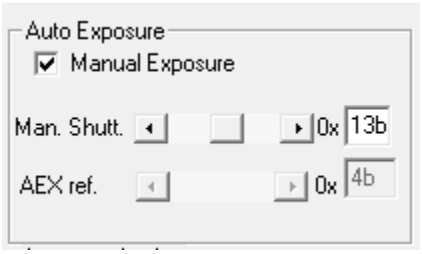


The check box for "enable overlay" is the global enable. If this is unchecked, none of the overlays will be visible. If the global overlay is checked you can select via the other 4 which overlay you want to be visible (if they are loaded).

7.3. Auto Exposure Settings (AEX)

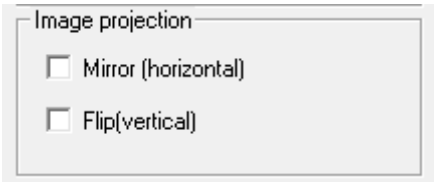
The camera offers two types of exposure: they are auto exposure and manual exposure. The camera will run in Auto exposure mode if the check box “Manual Exposure" is not checked. In this case the user can set the working point of the camera. Either by changing the scroll-bar AEX ref or editing the edit box at the right hand side of the scroll-bar. Nominal value for this setting is 0x48.

In the case manual Exposure is checked, the auto exposure routine is stopped and the camera will stay with the gain at its current position. The rolling shutter exposure time can be controlled with the scroll-bar "Man.Shutt” (Manual shutter). This can also be done with the edit box at the right hand side of the scroll-bar.



7.4. Mirror and Flip

To exchange the orientation of the camera in horizontal direction check the "Mirror" check box. For vertical use the "Flip" check box.

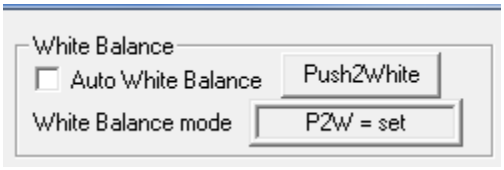


7.5. White Balance

The default mode for the camera is the auto white balance mode. However, if the camera is always used under one fixed specific type of light you can un-check the (Auto White Balance) box.

In the box for white balance mode you see that this changes from AWB! To Push2White running. This means that the auto mode is still running till you push the "Push2White" button. At that moment the auto mode will stop and the current reached settings will be hold inside the sensor (note they are not stored yet!). The text in the window will change to "P2W = set".

If you are not satisfied with the current settings press the "Push2White" button again, the system will start running again.



7.6. General Communication

In this block you can load if required an EEPROM file. Only use a file that you received from Videology. Loading another file can cause that you camera will not work anymore!

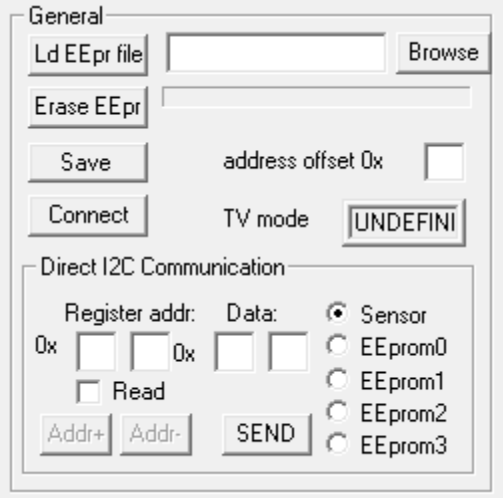
The "erase Eepr" button will erase all special settings of the camera. If you do not want to erase all, you can set an offset address.

The TV mode box indicates if your camera is PAL or NTSC.

With the "Save" button you can save all settings you made via this tool (outside the General area!) in the EEPROM of the camera. Please note at the moment you press Save, previous setting will be erased and overwritten.

Via the Direct I2C communication area you can send direct I2C commands to the camera. This tool lets you set registers, read registers and modify the EEPROM. You should only use this part if you are very familiar with the sensor and the camera.

Writing wrong data to the EEPROM can cause the camera not to work anymore! Therefore we strongly recommend not the use this part of the tool.



7.7. I2C Addresses

If you want to develop you own software to control the camera you need to know the following details:

Communication to the sensor: I2C write address = 0xBA, read address = 0xBB.

The register addresses are 16 bits (2 bytes- 2 x 8 bits). Also data is 16 bits.

The EEPROM uses max 8Kbites. These 8K are divided over 4 pages of each 256 bytes.

The register addresses of the eeprom are 8 bites (one byte). This is also the case of for the data. However page write and read can be done on the eeprom.

The addresses are:

page 0 : write 0xa0 read 0xa1
page 1 : write 0xa2 read 0xa3
page 2 : write 0xa4 read 0xa5
page 3 : write 0xa6 read 0xa7

8. Contact Information

For technical assistance with this product, please contact the supplier from whom the product was purchased.

For OEM inquiries, contact Videology® Imaging Solutions:

Americas, Middle East, Far East & Australia:
Videology® Imaging Solutions Inc.
37M Lark Industrial Parkway
Greenville, RI 02828
USA

Tel: (401) 949-5332
Fax: (401) 949-5276

Europe & N. Eurasia:
Videology® Imaging Solutions Europe B.V.
Neutronenlaan 4
5405 NH Uden
The Netherlands

Tel: +31 (0) 413-256261
Fax: +31 (0) 413-251712

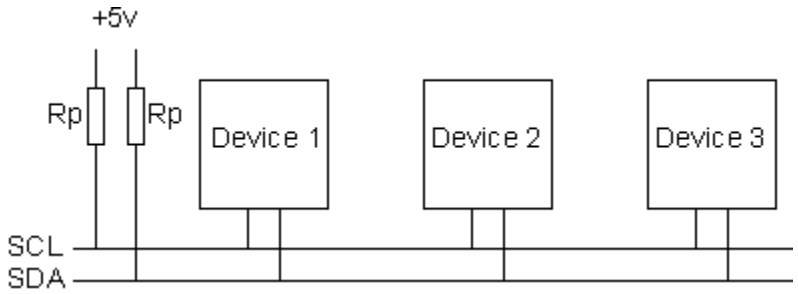
Please visit our website: videologyinc.com

VIDEOLOGY IMAGING SOLUTIONS is an ISO 9001 registered video camera developer and manufacturer serving industrial, machine vision, biometric, security, and specialty OEM markets. Videology designs, develops, manufactures, and distributes video, image acquisition, and display technologies and products to OEMs worldwide.

9. Appendix 1

9.1. Using the I²C Bus

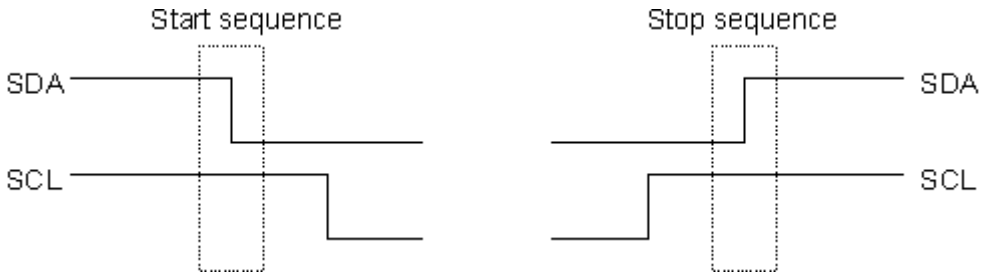
The physical I2C bus
This is just two wires, called SCL and SDA. SCL is the clock line. It is used to synchronize all data transfers over the I2C bus. SDA is the data line. The SCL & SDA lines are connected to all devices on the I2C bus. Both SCL and SDA lines are open drain drivers. What this means is that the chip can drive its output low, but it cannot drive it high. For the line to be able to go high you must provide pull-up resistors to the 5v supply. There should be a resistor from the SCL line to the 5v line and another from the SDA line to the 5v line. You only need one set of pull-up resistors for the whole I2C bus, not for each device, as illustrated below:



The value of the resistors is not critical. Anything from 1k8 (1800 ohms) to 47k (47000 ohms) can be used. If the resistors are missing, the SCL and SDA lines will always be low - nearly 0 volts - and the I2C bus will not work.

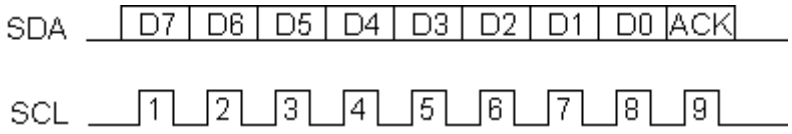
Masters and Slaves
The devices on the I2C bus are either masters or slaves. The master is always the device that drives the SCL clock line. The slaves are the devices that respond to the master. A slave cannot initiate a transfer over the I2C bus, only a master can do that. There can be, and usually are, multiple slaves on the I2C bus, however there is normally only one master. Slaves will never initiate a transfer. Both master and slave can transfer data over the I2C bus, but that transfer is always controlled by the master.

The I2C Physical Protocol
When the master (your controller) wishes to talk to a slave (our camera) it begins by issuing a start sequence on the I2C bus. A start sequence is one of two special sequences defined for the I2C bus, the other being the stop sequence. The start sequence and stop sequence are special in that these are the only places where the SDA (data line) is allowed to change while the SCL (clock line) is high. When data is being transferred, SDA must remain stable and not change while SCL is high. The start and stop sequences mark the beginning and end of a transaction with the slave device.



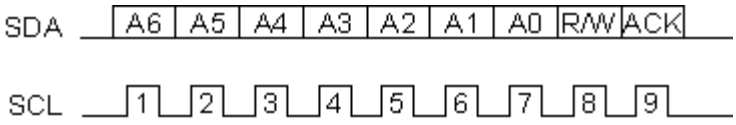
Data is transferred in sequences of 8 bits. The bits are placed on the SDA line starting with the MSB (Most Significant Bit). The SCL line is then pulsed high, then low. Remember that the chip cannot really drive the line

high, it simply "lets go" of it and the resistor actually pulls it high. For every 8 bits transferred, the device receiving the data sends back an acknowledge bit, so there are actually 9 SCL clock pulses to transfer each 8 bit byte of data. If the receiving device sends back a low ACK bit, then it has received the data and is ready to accept another byte. If it sends back a high then it is indicating it cannot accept any further data and the master should terminate the transfer by sending a stop sequence.



How fast?
The standard clock (SCL) speed for I2C up to 100KHz. Philips do define faster speeds: Fast mode, which is up to 400KHz and High Speed mode which is up to 3.4MHz. All of our cameras are designed to work at up to 100KHz.

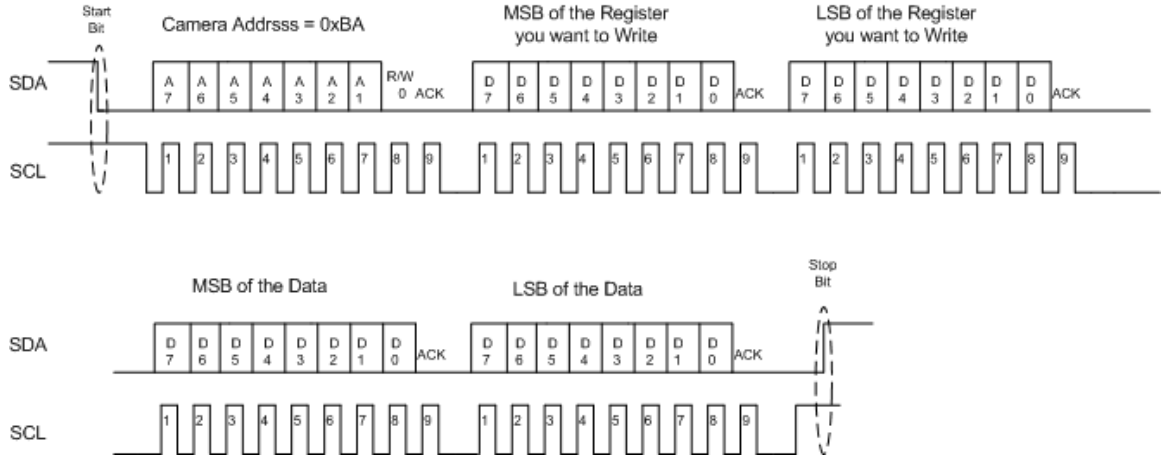
I2C Device Addressing
All I2C addresses are 7 bits. This means that you can have up to 128 devices on the I2C bus, since a 7bit number can be from 0 to 127. When sending out the 7 bit address, we still always send 8 bits. The extra bit is used to inform the slave if the master is writing to it or reading from it. If the bit is zero are master is writing to the slave. If the bit is 1 the master is reading from the slave. The 7 bit address is placed in the upper 7 bits of the byte and the Read/Write (R/W) bit is in the LSB (Least Significant Bit).



It is probably easier to think of the I2C bus addresses as 8 bit addresses, with even addresses as write only, and the odd addresses as the read address for the same device. To take our camera for example, this is at address 0xBA. You would uses 0xBA to write to the camera and 0xBB to read from it. So the read/write bit just makes it an odd/even address.

The I2C Software Protocol
The first thing that will happen is that the master will send out a start sequence. This will alert all the slave devices on the bus that a transaction is starting and they should listen in incase it is for them. Next the master will send out the device address. The slave that matches this address will continue with the transaction, any others will ignore the rest of this transaction and wait for the next. Having addressed the slave device the master must now send out the internal location or register number inside the slave that it wishes to write to or read from. Having sent the I2C address and the internal register address the master can now send the 2 data bytes. When the master has finished writing all data to the slave, it sends a stop sequence which completes the transaction. So to write to a slave device:
1. Send a start sequence
2. Send the I2C address of the slave with the R/W bit low (even address)
3. Send the internal register number you want to write to
4. Send the two data bytes
5. Send the stop sequence.

The bit sequence will look like this:

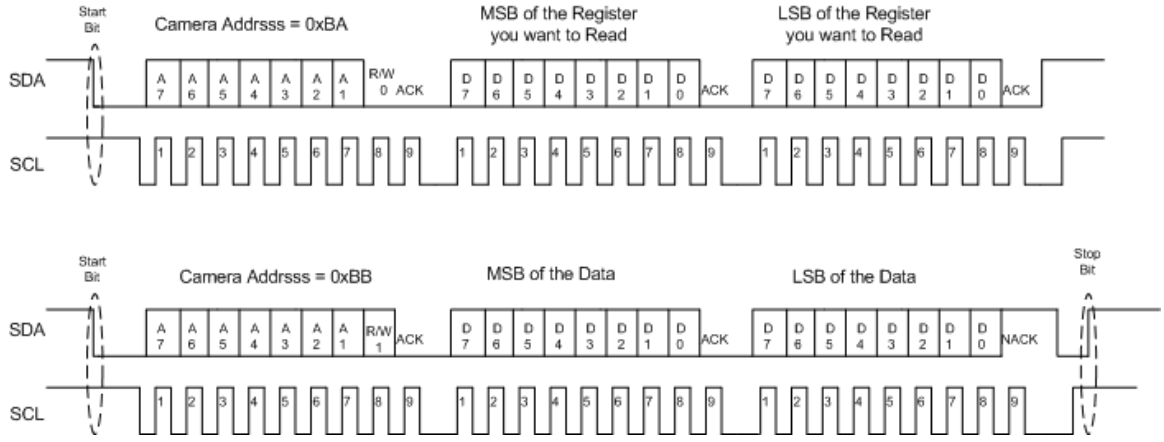


Reading from the Slave

This is a little more complicated - but not too much more. Before reading data from the slave device, you must tell it which of its internal addresses you want to read. So a read of the slave actually starts off by writing to it. This is the same as when you want to write to it: You send the start sequence, the I2C address of the slave with the R/W bit low (even address) and the internal register number you want to read from. Now you send another start sequence (sometimes called a restart) and the I2C address again - this time with the read bit set. You then read 2 data bytes and terminate the transaction with a stop sequence.

1. Send a start sequence
2. Send 0xBA (I2C address of the camera with the R/W bit low (even address)
3. Send 0x?? (MSB of the register address to read from)
3. Send 0x?? (LSB of the register address to read from)
4. Send a start sequence again (repeated start)
5. Send 0xBB (I2C address of the camera with the R/W bit high (odd address)
6. Read 2 data bytes from the camera.
7. Send the stop sequence.

The bit sequence will look like this:



Wait a moment

That's almost it for simple I2C communications, but there is one more complication. When the master is reading from the slave, its the slave that places the data on the SDA line, but its the master that controls the clock. What if the slave is not ready to send the data! With devices such as EEPROMs this is not a problem, but when the slave device is actually a microprocessor with other things to do, it can be a problem. The microprocessor on the slave device will need to go to an interrupt routine, save its working registers, find out what address the master wants to read from, get the data and place it in its transmission register. This can take many uS to happen, meanwhile the master is blissfully sending out clock pulses on the SCL line that the slave cannot respond to. The I2C protocol provides a solution to this: the slave is allowed to hold the SCL line low! This is called clock stretching. When the slave gets the read command from the master it holds the clock line low. The microprocessor then gets the requested data, places it in the transmission register and releases the clock line allowing the pull-up resistor to finally pull it high. From the master's point of view, it will issue the first clock pulse of the read by making SCL high and then check to see if it really has gone high. If it's still low then it's the slave that holding it low and the master should wait until it goes high before continuing. Luckily the hardware I2C ports on most microprocessors will handle this automatically.

The definitive specs on the I2C bus can be found on the Philips website. It is currently here.

10.Appendix 2

10.1. I²C commands

Technical note: 2xB14X/2xRD45(S) Camera module:

This is a description to set or change some of the settings inside the camera module.
The settings which can be changed are:

- Mirror, this is a horizontal flip of the image
- Flip, this is a vertical flip of the image
- Auto exposure saturation point
- Shutter speed
- AWB auto and freeze mode
- Enable/disable overlay

To set these functions and store them in the camera we strongly recommend to use the software tool SFT-13500 "camera control". The advantage is that settings can be stored easier inside the camera when you use this tool, without the risk of damaging the camera.

The I2C communication protocol has a device address, two register addresses and two data bytes. See figures 1 and 2 below.

Write action

S	Device Address (0xBA)	A	Reg byte 1	A	Reg byte 2	A	Data byte1	A	Data byte2	A	P

A= acknowledge
NA=Not acknowledge
S= Start condition
P=Stop condition

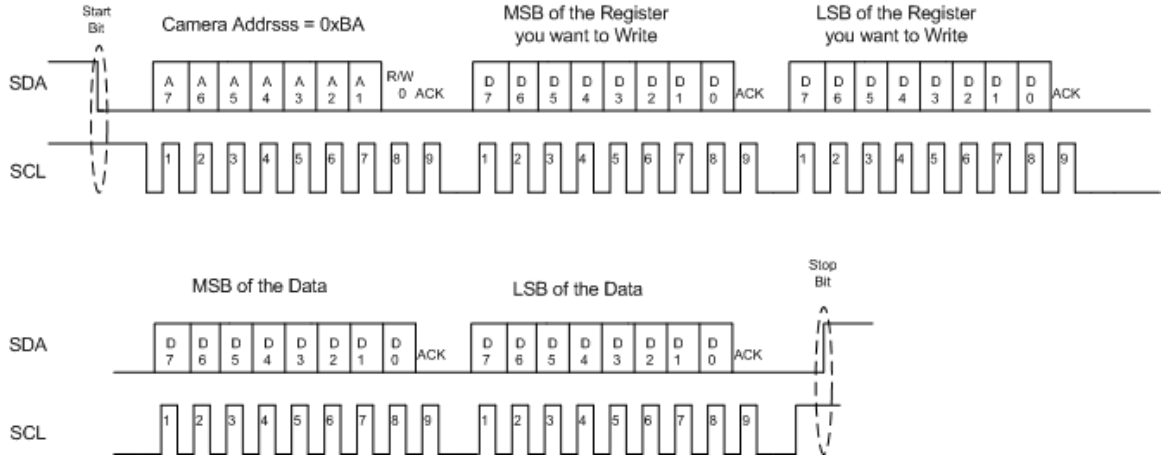


Figure 1 – Sensor Write Sequence.

Read action

S	Device Address (0xBB)	A	Reg byte 1	A	Reg byte 2	A	S	Data byte1	A	Data byte2	N A	P

A= acknowledge
NA=Not acknowledge
S= Start condition
P=Stop condition

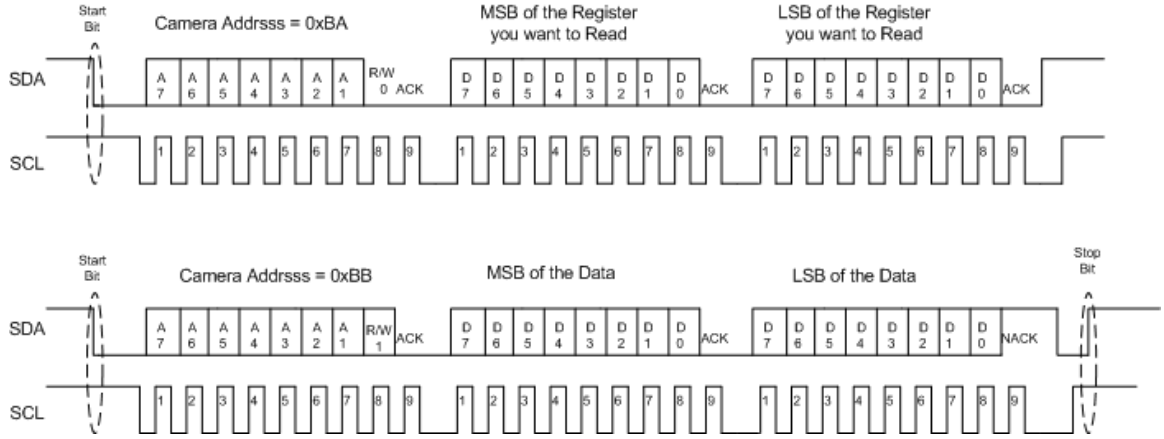


Figure 2 – Sensor Read Sequence.

Clock speed should not go above 100KHz!.
The sensor **device address** for writing is 0xBA and for reading is 0xBB.

Table of Byte sequence to write to Sensor

Command	**I2C Write Address**	**Data**			
		Reg byte 1	**Reg byte 2**	**Data byte 1**	**Data byte 2**
Global Overlay Enable	0xBA	0x4F	0x02	0x80	0x00
Global Overlay Disable	0xBA	0x4F	0x02	0x00	0x00
Enable Overlay 1	0xBA	0x4F	0x08	0x80	0x00
Disable Overlay 1	0xBA	0x4F	0x08	0x00	0x00
Enable Overlay 2	0xBA	0x4F	0x0A	0x80	0x01
Disable Overlay 2	0xBA	0x4F	0x0A	0x00	0x01
Enable Overlay 3	0xBA	0x4F	0x0C	0x80	0x02
Disable Overlay 3	0xBA	0x4F	0x0C	0x00	0x02
Enable Overlay 4	0xBA	0x4F	0x0E	0x80	0x03
Disable Overlay 4	0xBA	0x4F	0x0E	0x00	0x03
Move Overlay 1 in X	0xBA	0xFC	0x00	0x00	0x00
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x horiz. Offset MSB	0x horiz. Offset LSB
	0xBA	0x00	0x40	0x82	0x03
Move Overlay 1 in Y	0xBA	0xFC	0x00	0x00	0x01
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x vert. offset MSB	0x vert. offset LSB
	0xBA	0x00	0x40	0x82	0x03
Move Overlay 2 in X	0xBA	0xFC	0x00	0x01	0x00
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x horiz. Offset MSB	0x horiz. Offset LSB
	0xBA	0x00	0x40	0x82	0x03
Move Overlay 2 in Y	0xBA	0xFC	0x00	0x01	0x01
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x vert. offset MSB	0x vert. offset LSB
	0xBA	0x00	0x40	0x82	0x03

Move Overlay 3 in X	0xBA	0xFC	0x00	0x02	0x00
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x horiz. Offset MSB	0x horiz. Offset LSB
	0xBA	0x00	0x40	0x82	0x03
Move Overlay 3 in Y	0xBA	0xFC	0x00	0x02	0x01
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x vert. offset MSB	0x vert. offset LSB
	0xBA	0x00	0x40	0x82	0x03
Move Overlay 4 in X	0xBA	0xFC	0x00	0x03	0x00
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x horiz. Offset MSB	0x horiz. Offset LSB
	0xBA	0x00	0x40	0x82	0x03
Move Overlay 4 in Y	0xBA	0xFC	0x00	0x03	0x01
	0xBA	0xFC	0x02	0x00	0x00
	0xBA	0xFC	0x04	0x vert. offset MSB	0x vert. offset LSB
	0xBA	0x00	0x40	0x82	0x03

Table of Byte sequence to read from Sensor

Command	I2C Write Address	I2C Read Address	Data			
			Reg byte 1	Reg byte 2	Data byte 1	Data byte 2
Read Global Overlay Status	0xBA		0x4F	0x02		
		0xBB			0x00	if 0x01 enabled if 0x00 disabled
Read Status of Overlay 1	0xBA		0x4F	0x08		
		0xBB			If 0x80 enabled If 0x00 disabled	0x00
Read Status of Overlay 2	0xBA		0x4F	0x0A		
		0xBB			If 0x80 enabled If 0x00 disabled	0x00
Read Status of Overlay 3	0xBA		0x4F	0x0C		
		0xBB	0x4F	0x0C	If 0x80 enabled If 0x00 disabled	0x00
Read Status of Overlay 4	0xBA					
		0xBB	0x4F	0x0E	If 0x80 enabled If 0x00 disabled	0x00

10.2. EEProm - If settings need to be stored:

You can use the camera's EEprom to store your settings. But please take the maximum attention when you do this. The camera has an 8K eeprom on board. This eeprom has 4 pages. The last page is meant to store certain settings. But please take care in this page may contain already settings from the factory! You should not overwrite the factory settings, since this may influence the start up and final behavior of the camera!

To find out if settings are loaded, read from the EEProm. Settings are stored in groups of 4 bytes (two register address bytes and two data bytes). Keep reading until you find a group of 4 bytes with all 0xff. Here you should start storing your values. Do not leave unused spaces!!!

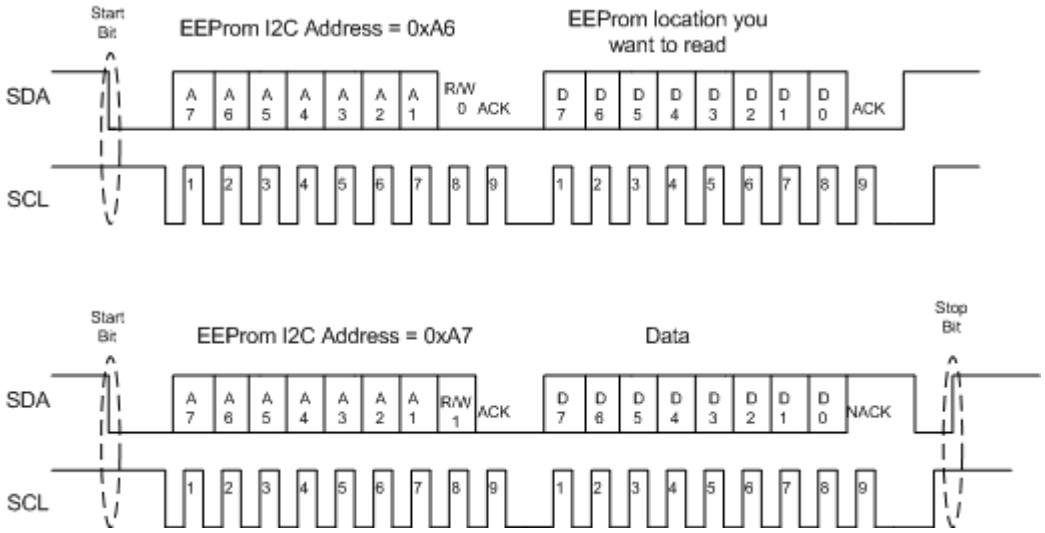


Figure 3 – EEProm Read Sequence.

In figure two you find a graphical example:

addr	0x0	0x 1	0x 2	0x 3	0x 4	0x 5	0x 6	0x 7	0x 8	0x 9	0xa
data	D1	D2	D3	D4	0xff	0xff	0xff	0xff	0xff	0xff	0xff

First address to load your settings

Figure 2.

Note that you always should store 4 bytes in the following order: Reg Addr MSB, Reg.Addr LSB, Data MSB and Data LSB. So in the case of figure two the MSB of the register address must be stored in register 0x04.

Please be very careful when you store settings. Preferably use Videology's SFT-13500 camera control software.

The Eeprom has a different I2C format. The register addresses and data addresses are only one byte. The device address for the EEProm page 4 is 0xa6 (write) and 0xa7 (read).

Table of Byte sequence to write to EEProm

NOTE: This table assumes the EEProm is in the factory default mode. As it comes from the factory, the EEProm is blank. Therefore, the first available address is 0x00. If this is not the case, the user must determine what the first blank address is as described above and enter the appropriate address for the EEProm Address. This table also assumes that the user wishes to store the overlay global enable and the X and Y position for all four overlays.

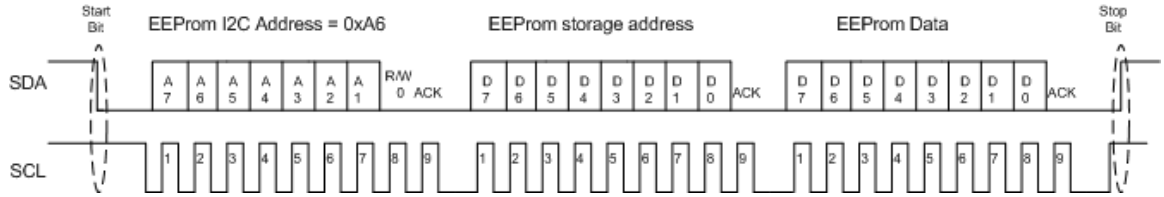


Figure4 – EEProm Write Sequence.

Command	**I2C Write Address**				
		EEProm Address	**EEProm Data**		
Global Overlay	0xA6	0x00	0x4F		
	0xA6	0x01	0x02		
	0xA6	0x02	0x80 = enable 0x00 = disable		
	0xA6	0x03	0x00		
Overlay 1	0xA6	0x04	0x4F		
	0xA6	0x05	0x08		
	0xA6	0x06	0x80 = enable 0x00 = disable		
	0xA6	0x07	0x00		

Overlay 2	0xA6	0x08	0x4F		
	0xA6	0x09	0x08		
	0xA6	0x0A	0x80 = enable 0x00 = disable		
	0xA6	0x0B	0x01		
Overlay 3	0xA6	0x0C	0x4F		
	0xA6	0x0D	0x08		
	0xA6	0x0E	0x80 = enable 0x00 = disable		
	0xA6	0x03F	0x02		
Overlay 4	0xA6	0x10	0x4F		
	0xA6	0x11	0x08		
	0xA6	0x12	0x80 = enable 0x00 = disable		
	0xA6	0x13	0x03		
Set Overlay 1s position in X	0xA6	0x14	0xFC		
	0xA6	0x15	0x00		
	0xA6	0x16	0x00		
	0xA6	0x17	0x00		
	0xA6	0x18	0xFC		
	0xA6	0x19	0x02		
	0xA6	0x1A	0x00		
	0xA6	0x1B	0x00		
	0xA6	0x1C	0xFC		
	0xA6	0x1D	0x04		
	0xA6	0x1E	0x horiz. Offset MSB		
	0xA6	0x1F	0x horiz. Offset LSB		
	0xA6	0x20	0x00		
	0xA6	0x21	0x40		
	0xA6	0x22	0x82		
	0xA6	0x23	0x03		
Set Overlay 1s position in Y	0xA6	0x24	0xFC		
	0xA6	0x25	0x00		
	0xA6	0x26	0x00		
	0xA6	0x27	0x01		

	0xA6	0x28	0xFC		
	0xA6	0x29	0x02		
	0xA6	0x30	0x00		
	0xA6	0x31	0x00		
	0xA6	0x32	0xFC		
	0xA6	0x33	0x04		
	0xA6	0x34	0x horiz. Offset MSB		
	0xA6	0x35	0x horiz. Offset LSB		
	0xA6	0x36	0x00		
	0xA6	0x37	0x40		
	0xA6	0x38	0x82		
	0xA6	0x39	0x03		
Set Overlay 2s position in X	0xA6	0x3A	0xFC		
	0xA6	0x3B	0x00		
	0xA6	0x3C	0x01		
	0xA6	0x3D	0x00		
	0xA6	0x3E	0xFC		
	0xA6	0x3F	0x02		
	0xA6	0x40	0x00		
	0xA6	0x41	0x00		
	0xA6	0x42	0xFC		
	0xA6	0x43	0x04		
	0xA6	0x44	0x horiz. Offset MSB		
	0xA6	0x45	0x horiz. Offset LSB		
	0xA6	0x46	0x00		
	0xA6	0x47	0x40		
	0xA6	0x48	0x82		
	0xA6	0x49	0x03		
Set Overlay 2s position in Y	0xA6	0x4A	0xFC		
	0xA6	0x4B	0x00		
	0xA6	0x4C	0x01		
	0xA6	0x4D	0x01		
	0xA6	0x4E	0xFC		
	0xA6	0x4F	0x02		
	0xA6	0x50	0x00		
	0xA6	0x51	0x00		
	0xA6	0x52	0xFC		

	0xA6	0x53	0x04		
	0xA6	0x54	0x horiz. Offset MSB		
	0xA6	0x55	0x horiz. Offset LSB		
	0xA6	0x56	0x00		
	0xA6	0x57	0x40		
	0xA6	0x58	0x82		
	0xA6	0x59	0x03		
Set Overlay 3s position in X	0xA6	0x5A	0xFC		
	0xA6	0x5B	0x00		
	0xA6	0X5C	0x02		
	0xA6	0x5D	0x00		
	0xA6	0x5E	0xFC		
	0xA6	0x5F	0x02		
	0xA6	0x60	0x00		
	0xA6	0x61	0x00		
	0xA6	0x62	0xFC		
	0xA6	0x63	0x04		
	0xA6	0x64	0x horiz. Offset MSB		
	0xA6	0x65	0x horiz. Offset LSB		
	0xA6	0x66	0x00		
	0xA6	0x67	0x40		
	0xA6	0x68	0x82		
	0xA6	0x69	0x03		
Set Overlay 3s position in Y	0xA6	0x6A	0xFC		
	0xA6	0x6B	0x00		
	0xA6	0x6C	0x02		
	0xA6	0x6D	0x01		
	0xA6	0x6E	0xFC		
	0xA6	0x6F	0x02		
	0xA6	0x70	0x00		
	0xA6	0x71	0x00		
	0xA6	0x72	0xFC		
	0xA6	0x73	0x04		
	0xA6	0x74	0x horiz. Offset MSB		
	0xA6	0x75	0x horiz. Offset LSB		

	0xA6	0x76	0x00		
	0xA6	0x77	0x40		
	0xA6	0x78	0x82		
	0xA6	0x79	0x03		
Set Overlay 4s position in X	0xA6	0x7A	0xFC		
	0xA6	0x7B	0x00		
	0xA6	0x7C	0x03		
	0xA6	0x7D	0x00		
	0xA6	0x7E	0xFC		
	0xA6	0x7F	0x02		
	0xA6	0x80	0x00		
	0xA6	0x81	0x00		
	0xA6	0x82	0xFC		
	0xA6	0x83	0x04		
	0xA6	0x84	0x horiz. Offset MSB		
	0xA6	0x85	0x horiz. Offset LSB		
	0xA6	0x86	0x00		
	0xA6	0x87	0x40		
	0xA6	0x88	0x82		
	0xA6	0x89	0x03		
Set Overlay 4s position in Y	0xA6	0x8A	0xFC		
	0xA6	0x8B	0x00		
	0xA6	0x8C	0x03		
	0xA6	0x8D	0x01		
	0xA6	0x8E	0xFC		
	0xA6	0x8F	0x02		
	0xA6	0x90	0x00		
	0xA6	0x91	0x00		
	0xA6	0x92	0xFC		
	0xA6	0x93	0x04		
	0xA6	0x94	0x horiz. Offset MSB		
	0xA6	0x95	0x horiz. Offset LSB		
	0xA6	0x96	0x00		
	0xA6	0x97	0x40		
	0xA6	0x98	0x82		
	0xA6	0x99	0x03		

10.3. Additional I²C Commands available

In the tables below you find per command the required actions to change the camera mode:

Mirror mode:

1) Read register 0x3254
2) Read register 0x301c (mirror status = bit[0] (0=normal, 1=mirror)
3) If mode needs to be changed
   a) Set register 0x301c bit[0] accordingly. All other bits should NOT be changed.
   b) Set register 0x3254 bit[0]. Incase of mirror b[0]=1, in normal mode b[0]=0. All other bits should NOT be changed.

Flip mode:

1) Read register 0x3254.
2) Read register 0x301c (flip status = bit[1] (0=normal, 1=flip)
3) If mode needs to be changed:
   a) Set register 0x301c bit[1] accordingly. All other bits should not be changed.
   b) Set register 0x3254. In case of flip b[1] = 1, in normal mode b[1] = 0;

Auto exposure saturation point

1) Read register 0xa804 to check if the camera is in AEX mode. Value 0x000f is AEX mode, 0x0000 is manual exposure mode.
2) If mode is not correct load register 0xa804 with 0x000f.
3) Read current saturation point from register 0xa812
4) Load register 0xa812 with the new saturation level data

Shutter speed

1) Read register 0xa804 to check if the camera is in manual mode. Value 0x000f is AEX mode, 0x0000 is manual exposure mode.
2) If mode is not manual load register 0xa804 with 0x0000.
3) Read current shutter value from register 0x3012.
4) Load register 0x3012 with the new shutter value

Auto white balance mode

1) Read the current white balance mode status from register 0xac04 (if value is 0x00ff than it is Auto white balance mode).
2) If mode is not correct load register 0xac04 with value 0x00ff.

Freeze mode:

To use this mode the camera must be first put in the freeze mode. The camera first needs to run in AWB mode. It will look for the most optimal white balance setting. When this is reached the AWB must be hold, and preferable the R/B gains must be stored and used when the camera is powered up again.

Put the camera in the freeze mode:

1) Make sure camera is in AWB mode by reading register 0xac04. Value 0x00ff means AWB mode. If the camera is not in the correct mode see table Auto White balance Mode.
2) If the AWB reached it's most optimal position stop the AWB function by loading register 0xac04 with value 0x0000.
3) Wait 10 mSec.
4) Read the following registers: 0xac02, 0xac04, 0xac0a, 0xac0c, 0xac0e, 0xac10, 0xac12, 0xac14, 0xac16, 0xac18, 0xac10, 0xac1a, 0xac1c, 0xac1e, 0xac32, 0xac36, 0xac38, 0xac3a, 0xac3c, 0xac3e, 0xac40, 0xac42, 0xac44, 0xac46, 0xac48, 0xac4a, 0xac4c and 0xac4e. wait 5 mS between each read command.
5) These values must be stored so that the can be loaded when the camera is powered up.

Please note: Videology will not give any warranty in case settings are stored incorrectly, or settings are overwritten!