Document Number: 8498820
Revision: F
CAGE Code: 81395

MICRO**IR**™

SOFTWARE INTERFACE CONTROL DOCUMENT

TWV640 UNCOOLED 12 µm (MICRON) THERMAL IMAGER

October 26, 2017

Prepared By:

BAE SYSTEMS

Information & Electronic Systems Integration Inc.
2 Forbes Road
Lexington, MA 02421

EXPORT- CONTROLLED DATA. This document contains technical information whose export is governed by the U.S. Export Administration Regulations (EAR). This information must not be transferred to a foreign person without proper authorization of the U.S. Department of Commerce. Violations may result in administrative, civil or criminal penalties.

Revision History

Rev	Date	Description
-	10/27/2015	Initial release per ECO EOB57783
A	05/12/2016	Revised IAW ECO EOB67860
B	06/24/2016	Revised IAW ECO EOB68352
C	02/07/2017	Revised IAW ECO EOB73836
D	05/09/2017	Revised IAW ECO EOB77729
E	09/14/2017	Revised IAW ECO EOB81672
F	10/26/2017	Revised IAW ECO EOB83197

Table of Contents

Abbreviations and Acronyms vi
1. Introduction 1
1.1 Applicable Documents 1
1.2 Interfacing with TWV640 Camera Cores 1
1.3 Camera Models and Their Interfaces 1
2. Communicating with the Camera 2
2.1 USB Identifying Information 2
2.2 USB Connection Endpoints 2
3. ASCII Commands 3
3.1 Understanding the ASCII Response 3
3.2 USB Settings 4
3.3 UART Settings 4
4. List of Software Commands 5
4.1 Adaptive Color Command (adaptive_color) 6
4.2 Auto Contrast Enhancement Level Command (auto_ce_lvl) 7
4.3 Manual Gain Manual Level Command (mgml) 8
4.4 Color Enhanced Imagery Command (color) 10
4.5 User Configurable Graphics Overlays Command (usr_overlays) 11
4.6 Display Command (display) 12
4.7 Fine Offset Calibration Command (foc) 13
4.8 Flash Frame Capture Command (framecapture) 14
4.9 Set Frame Rate Command (framerate) 15
4.10 Display Gamma Command (gamma) 16
4.11 Parameter Command (param) 17
4.12 Serial Number Command (serial_num) 17
4.13 Change Infrared Polarity Command (polarity) 18
4.14 Version Command (version) 18
4.15 Sharpness Command (sharpness) 19
4.16 Temperature Sensor Command (temp_sen) 19
4.17 Default video output command (video_output) 20
4.18 Output Video to Analog Command (video_analog) 20
4.19 Output Video to USB Command (video_usb) 21
4.20 Output Video to 16-Bit Parallel Command (video_16bit) 22
4.21 Output Video to 8-bit YCbCr (video_ycbcr) 22
4.22 Output Video to eMagin (video_emagin) 23
4.23 Electronic Zoom Functionality Command (zoom) 24
4.24 Nanomotion Control Commands (nanomot) 25
4.25 Shutter Control Commands (shutter) 25
4.26 Overlay Commands (overlay) 26
4.27 Crop Commands (crop) 27
4.28 Test Pattern Commands (testpattern) 27
4.29 Vertical Video Direction Control Commands (v_dir) 28

4.30 Horizontal Video Direction Control Commands (h_dir) ... 28
4.31 Delay Command Processing (delay) ... 29
4.32 Help Command (help) ... 29
5. Command Quick Reference ... 30
6. Parameters ... 34
7. Data Interface ... 39

List of Figures

Figure 2-1: USB Endpoints ... 2
Figure 3-1: Optional command formatting ... 3
Figure 4-1: Command Structure ... 5
Figure 4-2: mgml example ... 9
Figure 4-3: mgml example ... 9

List of Tables

Table 3-1: USB Serial Port Settings ... 4
Table 3-2: UART Serial Port Settings ... 4
Table 4-1 adaptive_color command ... 6
Table 4-2 auto_ce_lvl command ... 7
Table 4-3 mgml command ... 8
Table 4-4 color command ... 10
Table 4-5 usr_overlays command ... 11
Table 4-6 display command ... 12
Table 4-7 foc command ... 13
Table 4-8 framecapture command ... 14
Table 4-9 framerate command ... 15
Table 4-10 gamma command ... 16
Table 4-11 param command ... 17
Table 4-12 serial_num command ... 17
Table 4-13 polarity command ... 18
Table 4-14 version command ... 18
Table 4-15 sharpness command ... 19
Table 4-16 temp_sen command ... 19
Table 4-17 video_ouput command ... 20
Table 4-18 video_analog command ... 20
Table 4-19 video_usb command ... 21
Table 4-20 video_16bit command ... 22
Table 4-21 video_ycbcr command ... 22
Table 4-22 video_emagin command ... 23
Table 4-23 zoom command ... 24

Table 4-24 zoom limits 24
Table 4-25 nanomot command 25
Table 4-26 shutter command 25
Table 4-27 overlay command 26
Table 4-28 crop command 27
Table 4-29 testpattern command 27
Table 4-30 v_dir command 28
Table 4-31 h_dir command 28
Table 4-32 delay command 29
Table 4-33 help command 29
Table 5-1: Command List 30

Abbreviations and Acronyms

Abbreviation	Description
ASCII	American Standard Code for Information Exchange
CDC	Communications Device Class
EEPROM	Electrically Erasable Programmable Read-Only Memory
FOC	Fine Offset Calibration
FPA	Focal Plane Array
FPGA	Field Programmable Gate Array
GUI	Graphical User Interface
ICD	Interface Control Document
IR	Infrared
NTSC	National Television System Committee
NUC	Non-Uniformity Correction
OLED	Organic Light Emitting Diode
PAL	Phase Alternative Line
UART	Universal Asynchronous Receiver Transmitter
ROI	Region Of Interest
USB	Universal Serial Bus
USB HS	Universal Serial Bus High Speed
VGA	Video Graphics Array (640x480)

1. Introduction

The TWV640 and TWV640i camera cores can be controlled by the provided MicroIR™ GUI application running on a personal computer or through customer-designed software or embedded systems.

This document describes the software interface for the TMV640/TMV640i infrared camera core family of products. Its purpose is to provide the available commands and a concise description of each command. All commands are in an easy to read ASCII format and can be communicated to the unit via Universal Serial Bus (USB) or a Universal Asynchronous Receiver Transmitter (UART) interface.

1.1 Applicable Documents

The following documents will be useful to readers of this ICD

Document Title	Document Number
Mechanical ICD: *Camera Core*	8536474
Electrical ICD	8498744
USB 2.0 Specification	
libusb c library	

1.2 Interfacing with TWV640 Camera Cores

There are two standard hardware configurations of the TWV640 cores; a base configuration with everything needed to image and a configuration with an added interface board.

- Base configuration is ideal for embedded systems and has two methods for control – 3 pins for the industry standard USB 2.0 HS and an RS-232 port for a UART.
- Added interface board has individual standard connectors for micro USB and analog video

While some of the timing and electrical details of the UART and USB protocols are different, the commands and command structure described in this document applies to both interfaces.

NOTE: For detailed information on all electrical interfaces to and from the camera units, see the Electrical Interface Control Document.

1.3 Camera Models and Their Interfaces

The TWV640 and TWV640i are very similar in terms of commands available and what function each command represents, and unless noted the user should assume the command is the same across all cameras.

The TWV640 supports 30 and 60 Hz frame rates and TWV640i only supports a 7.5 Hz frame rate. Although the TWV640i does not support the 30 Hz and 60 Hz frame rates, the command interface is the same.

2. Communicating with the Camera

The base configuration of the TWV640 can be configured and monitored via an ASCII interface through either USB or UART, while the configuration with an interface board has only a micro USB port. Refer to the Electrical ICD document for more detailed information on each connector. The serial ASCII protocols are designed to be intuitive and this document provides the details necessary so that the commands may be used and called properly. The USB interface also has a high speed data endpoint for video transfers.

2.1 USB Identifying Information

The TWV640 has a Vendor ID (VID) of 8765 and a Product ID (PID) of 1235

2.2 USB Connection Endpoints

The TWV640 enumerates as a composite USB device consisting of a generic bulk interface and a CDC serial port using endpoints 1 IN, 1 OUT, and 2 IN, 2 OUT and 3 IN.

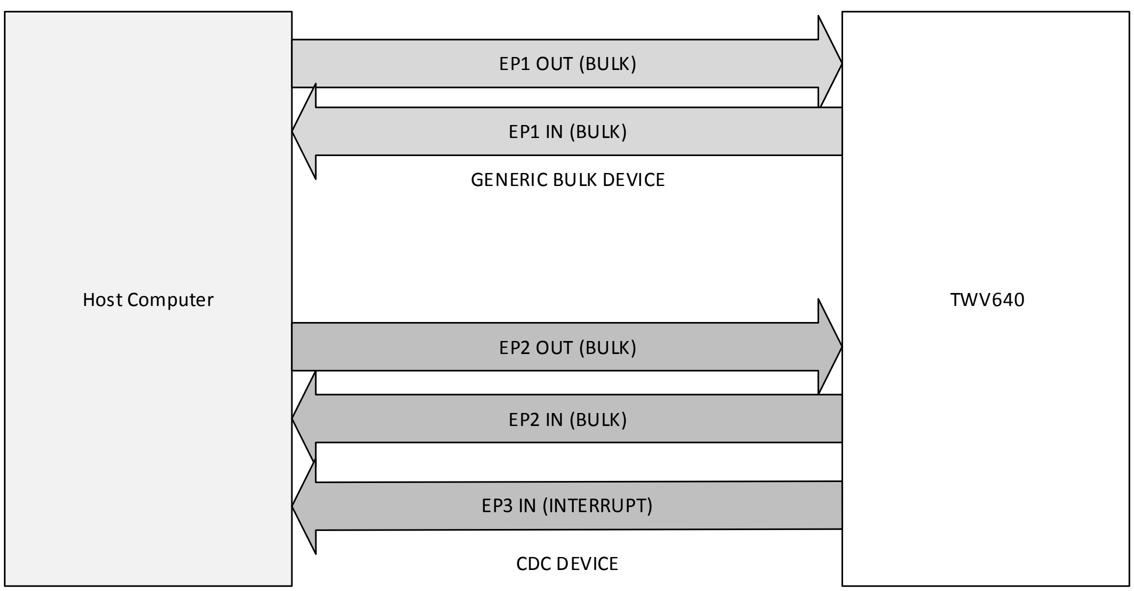


Figure 2-1: USB Endpoints

3. ASCII Commands

For formatting purposes, special characters are omitted from command descriptions in this document and the TWV640 cameras can accept commands with or without the special formatting. This section describes the optional full formatting for commands that may make system communication more robust. There are two places special characters may appear in the command and response of a unit – before the command and after the command. Each command can be prepended with the total ASCII character count. The end of the command must be terminated with carriage return and newline characters.

For example, if the system intends to send the command 'polarity set black' to a camera core, the message is the text of the command followed by the carriage return character. The total character count of the command does not need to be added to the command string manually.

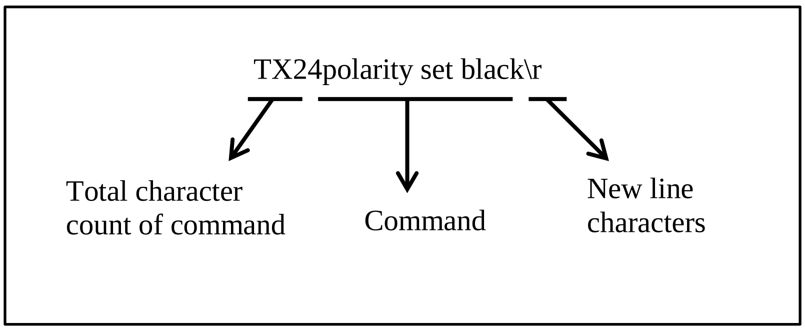


Figure 3-1: Optional command formatting

Before the command the following text is added - TX24. The section before the command includes the letters TX following by a two-digit number that represents the total character count of the command, including any special characters.

NOTE: If new line characters are used, verify that software communicating with the device doesn't automatically send the new line characters as well. The system will parse the command incorrectly and respond with "badly formed command".

3.1 Understanding the ASCII Response

Each ASCII command results in a response that demonstrates the unit has received and is processing the command. The response is in the following format:

TX*##MESSAGE\r*

Where:

- ## = Two digit decimal ASCII number with the number of bytes in the message, including the carriage return \r
- *MESSAGE* = is the text that describes the status of the command

NOTE: For documentation purposes only, the responses noted in section 3 do not show the "TX##".

The following is an example of how the *polarity set black* command is documented with its response:

ASCII Command:	polarity *set black*
Response:	Processing: polarity set black >

The response can be further separated into two parts – processing and the end of the command. 'Processing: polarity set black' means the unit has received the command successfully and is currently executing the command. The '>' character indicates the unit is ready for another command. There is a large variation in how much time each command takes to finish. Instead of setting a constant time, the system should wait for the receipt of the '>' character before sending another command. While receiving the '>' means the unit is ready for a new command, it doesn't always signify that the previous command is 100% finished executing. There are cases where a command can be executed, but up to 1 frame of latency occurs before the changes are seen.

NOTE: Commands that change default settings will result in a write to flash. Power to the unit must not be interrupted until the '>' character has been returned.

3.2 USB Settings

See **Error! Reference source not found.** for the USB serial port settings. Currently the serial port settings cannot be changed.

Table 3-1: USB Serial Port Settings

Serial Port Configuration Parameter	Value
Baud Rate	115,200 bps (bits per second)
Parity	NONE
Data Bits	8 bits per byte
Stop Bits	1 stop bit per byte
Encoding	ASCII

3.3 UART Settings

See **Error! Reference source not found.** for the UART serial port settings.

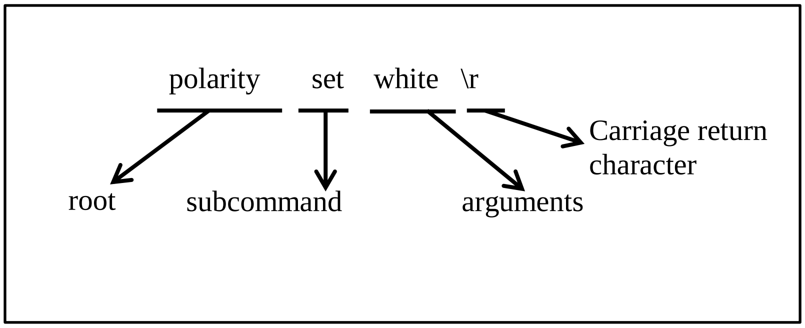
Table 3-2: UART Serial Port Settings

Serial Port Configuration Parameter	Value
Baud Rate	19,200 bps (bits per second)
Parity	NONE
Data Bits	8 bits per byte
Stop Bits	1 stop bit per byte
Encoding	ASCII

4. List of Software Commands

This section lists the supported commands. Depending on the product, there may be additional commands, and some commands listed here may not be available for your product. To determine the version of the embedded software build, send the **version info** command.

The commands are broken into three parts – root command, subcommand and arguments. The root command is the first word of the command. The subcommand is the second word and any remaining parts are the argument(s). Not all commands have all three parts, see **Error! Reference source not found.** below



for an example.

Figure 4-1: Command Structure

The most helpful command in the list is **help**, which displays all the root commands that are available. To further explain each command, subcommand and arguments, type *HELP <ROOT CMD>* (for example, **help version**). See the remainder of Section 3, as well as Section 4, for more details about individual commands.

NOTE - Commands are not case sensitive.

4.1 Adaptive Color Command (adaptive_color)

The camera units allow custom graphics to overlay over active imagery. The adaptive color command allows the color of the additional custom graphics to be dependent upon the color of the displayed imagery. For example, if the custom graphics to be overlaid over an image are white and imagery under the custom graphics end up white, the graphic overlay would become invisible. The adaptive color allows the color of the custom graphic to be changed if the average of the region of interest (ROI) underneath the custom graphic crosses either low or high thresholds.

Note on the region of interest – the shape is actually a square with a side of (radius * 2 + 1) with the center of the square falling on (x-center, y-center). If the radius is zero, the ROI is a single pixel (x-center, y-center). For example, if the radius is 5 the ROI is an 11-pixel by 11-pixel square.

Root Command: adaptive_color

Sample ASCII Command: adaptive_color *read*

Sample Response: Processing: adaptive_color read
AVG = 126.315750
>

Table 4-1 adaptive_color command

Subcommand	Argument	Description
roi	x-center	Establishes a region of interest (ROI) that defines average value of what is considered underneath the custom graphic. All pixels in region of interest are averaged and that value is used to determine where background stands in relation to adaptive color threshold.
	y-center	
	radius	
high	r_component	If the average value of the ROI pixels is above the set threshold, this setting sets the new color of custom graphic pixels in Red, Blue, and Green format.
	g_component	
	b_component	
low	r_component	If the average value of the ROI pixels is below the set threshold, this setting sets the new color of custom graphic pixels in Red, Blue, and Green format.
	g_component	
	b_component	
thresh	high_threshold	Sets the threshold for the adaptive_color command.
	low_threshold	
enable	true | false	Enables (true) or disables (false) adaptive color setting.
read	-----	Returns ROI average pixel value

4.2 Auto Contrast Enhancement Level Command (auto_ce_lvl)

The contrast enhancement level represents a tradeoff between image dynamic range and contrast. Higher enhancement level makes smaller emittance differences more discernable but reduces the visible dynamic range. By default, the enhancement algorithm uses the majority of the image for enhancement of the entire image – a rectangle with (x-position, y-position) coordinates of (23, 16) and width of 591 and height of 445. The window can be changed using the auto_ce_lvl command, but must be at least 10 pixels wide and 10 pixels high.

Root Command: auto_ce_lvl

Sample ASCII Command: auto_ce_lvl *set 5*

Sample Response: Processing: auto_ce_lvl set 5
>

Table 4-2 auto_ce_lvl command

Subcommand	Argument	Description
default	new_gain_level x-position y-position height width	Sets the default contrast enhancement level. Valid range of 0 to12. Window ROI defaults can also be set, but is optional. Note: If window defaults are to be changed, all 5 arguments must be present
restore	-----	Returns contrast enhancement window values to factory settings
get	-----	Returns the auto-contrast enhancement level and window information.
set	new_gain_level	Sets the contrast enhancement level to a number between 0 and 12.
window	x-position y-position height width	Sets the window in the image the algorithm uses to set the dynamic range of the entire image. The x-position and y-position are relative to the upper left corner of the image, and width increases from left to right and height increases up to down.

4.3 Manual Gain Manual Level Command (mgml)

Manual gain and level commands allow you to manually set gain and level based on 16-bit video values. Gain and level values can be updated at once per frame with up to one full frame of latency before observing the result.

Root Command: mgml

Sample ASCII Command: mgml *enable*

Sample Response: Processing: mgml enable
>

Table 4-3 mgml command

<table>
<tr><th>Subcommand</th><th>Argument</th><th>Description</th></tr>
<tr><td>enable</td><td>-----</td><td>Enables manual gain/level module</td></tr>
<tr><td>disable</td><td>-----</td><td>Disables manual gain/level module (return to auto_ce)</td></tr>
<tr><td rowspan="2">setthresh</td><td>low_threshold</td><td rowspan="2">Alternate method of setting gain and level settings by setting the upper and lower limits having gain and level values be determined automatically</td></tr>
<tr><td>high_threshold</td></tr>
<tr><td>autolvl</td><td>true | false</td><td>Enable/Disable auto level mode</td></tr>
<tr><td rowspan="3">gainlevel</td><td>get | set</td><td rowspan="3">If using get, no additional arguments are required.
If using set, both gain and level values must be entered in order for them to be accepted.
NOTE: When in auto level, the level argument becomes a level offset.</td></tr>
<tr><td>gain</td></tr>
<tr><td>level / level offset</td></tr>
<tr><td>gain</td><td>+ | -</td><td>Increment/Decrement gain value based on the parameter MGML.GAIN_STEP</td></tr>
<tr><td>level</td><td>+ | -</td><td>Increment/Decrement level value based on the parameter MGML.LEVEL_STEP</td></tr>
<tr><td>seed</td><td>-----</td><td>Pulls in gain and level values from auto contrast enhancement.</td></tr>
</table>

Gain: The gain value is defined as the width of the histogram in 16-bit video counts for the mgml module. The lower the gain value, the tighter the histogram will be, allowing for the ability to pull more contrast out of a scene that otherwise wouldn't have much. Higher gain values can be used in high contrast scenes to prevent too much of the image from being saturated.

Manual Level Mode: The level value is defined as the center value for the histogram used in the mgml module. The level value is an unsigned value with acceptable inputs ranging from 0-65535. Minimum and maximum values are controlled by the parameters MGML.LEVEL_MIN and MGML.LEVEL_MAX.

Auto Level Mode: The level is defined as an offset to the frame average that is used to automatically center the video. The level value that is used to determine the center of the histogram is frame average + level offset. The level offset value in this instance can be either positive or negative. Minimum and maximum offset values are controlled by the parameters MGML.LEVEL_OFFSET_MIN and MGML.LEVEL_OFFSET_MAX.

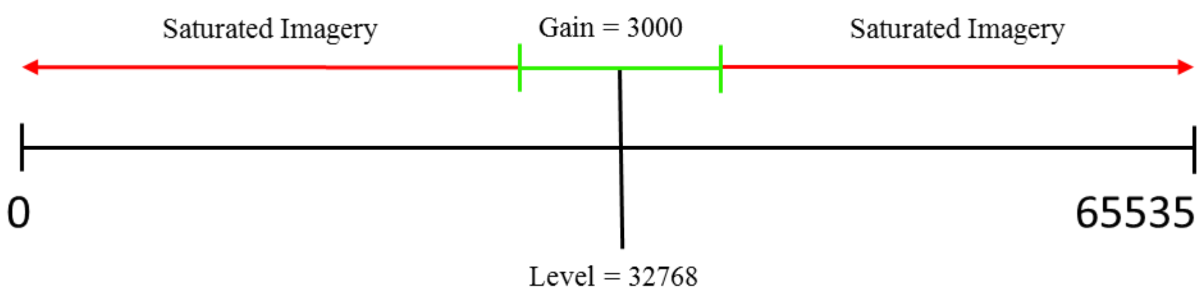


Figure 4-2: mgml example

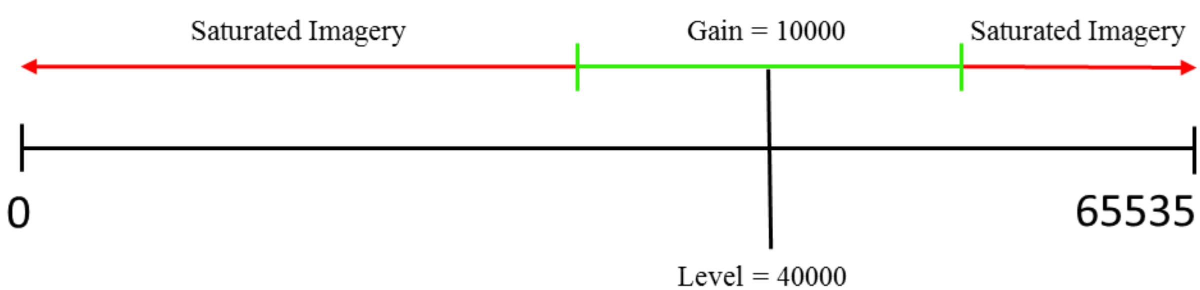


Figure 4-3: mgml example

4.4 Color Enhanced Imagery Command (color)

Instead of an image ranging between black and white to represent different infrared emittance levels, there is a “false color” option to instead have an imagery that ranges from blue to red. In all settings, blue represents black and red represents white. Therefore, if the polarity of the image is set to white-hot then the color red would represent higher infrared emittance. If the polarity of the image is set to black-hot then the color blue would represent higher infrared emittance. Color is only available in 24-bit display mode.

Root Command: color

Sample ASCII Command: color *enable true*

Sample Response: Processing: color enable true
>

Table 4-4 color command

Subcommand	Argument	Description
enable	true | false	Enables or disables color look up table in the infrared imagery.

The following bars illustrate the mapping of black and white to “false colors” for white-hot video.

Black White

Blue Red

4.5 User Configurable Graphics Overlays Command (usr_overlays)

The TWV640 camera core allows graphics to be overlaid on live imagery. The usr_overlays command can be used to manipulate already existing overlay graphics on the camera core in terms of where it shows up in the imagery and its duration. Generating overlay graphics is described in the MicroIR GUI User's Guide.

Root Command: usr_overlays

Sample ASCII Command: usr_overlays *show splash*

Sample Response: Processing: usr_overlays show splash
>

The usr_overlays list command returns a list of existing graphic overlays through the ASCII interface:

Sample ASCII Command: usr_overlays *list*

Sample Response: Processing: usr_overlays list
Start Listing Gui Object Item Names:
gui_overlay_1
gui_overlay_2
gui_overlay_3
End Listing Gui Object Item Names
>

Table 4-5 usr_overlays command

Subcommand	Argument	Description
enable	true | false	Enable or disable user overlays.
hide	overlay_name	Disables custom graphic overlay named overlay_name.
list	-----	Returns a list of graphic overlays stored on the camera core.
move	overlay_name	Moves center of graphic overlay named overlay_name to (x-position, y-position) relative to upper left corner of live imagery.
	x-position	
	y-position	
movec	overlay_name	Moves center of graphic overlay named overlay_name to (x-position, y-position) relative to center.
	x-position	
	y-position	
layer	overlay_name	Set the layer of a user defined overlay to overlay_name to layer (1=foreground, 2=midground, 3=background)
	layer	
refresh	-----	Updates list of available user-defined overlays.
show	overlay_name	Enables custom graphic overlay named overlay_name.
timer	overlay_name	Shows user-defined overlay overlay_name for time_in_seconds amount of time on the screen, after which overlay is no longer shown.
	time_in_seconds	

4.6 Display Command (display)

The TWV640 allows direct access to the settings of the eMagin VGA family of OLED displays. Using this command, the settings of an eMagin VGA OLED connected to the unit can be queried or set.

Root Command: display

Sample ASCII Command: display *dimctl 00*

Sample Response: Processing: display dimctl 00
Finished DIMCTL update
>

Table 4-6 display command

Subcommand	Argument	Description
dimctl	brightness_level	Sets display brightness level, hexadecimal value that can range from 0 to 7F without 0x.
idrf	final_idrf_value	Populated by the factory.
read	register_address	Read eMagin register value at register_address.
readp	register_address	Read eMagin EEPROM register values.
shift	x-offset y-offset	Sets image offset relative to center of display. x-offset and y-offset are signed integers.
startup	-----	Starts display if not already started
write	register_address data	Overwrite eMagin register value at address register_address and value data.
writep	register_address data	Write eMagin EEPROM register values.

4.7 Fine Offset Calibration Command (foc)

There are two ways of calibrating each unit with non-uniformity correction (NUC) – using the scene or a blade if available. Using a blade shutter does not require any particular kind of scene, but the resulting calibration does not include effects from the lens and any physical variables past the shutter. Calibrating to the scene requires a uniform scene, with something in front of the camera that covers the entire frame of view that is uniform, but results in calibration that includes the lens in front of the camera. The unit must have a shutter in order to calibrate with a blade and currently the base configuration employs the use of a Nanomotion shutter. If a shutter is not available, the unit will display an error letting the user know that the I2C interface failed.

Every time a unit calibrates, even if automated, there is a message through ASCII on the USB and UART ports. The output states "Starting FOC" on shutter start and "FOC Complete" when calibration has been finished:

Root Command: foc

Sample ASCII Command: foc *scene*

Sample Response:
Processing: foc scene
Starting FOC
FOC Complete
>

A similar message is also output when the camera calibrates automatically:

Response:
Starting FOC
FOC Complete

Automated shuttering is depending on three system parameters, the units being seconds:

FINE_OFFST_CAL.FOCSHUTTERINCR - INCREMENT

FINE_OFFST_CAL.FOCSHUTTERMAX - MAXIMUM

FINE_OFFST_CAL.FOCSHUTTERMIN – MINIMUM

The first shutter happens after the first MINIMUM amount of seconds, from then on the period between shutters is incremented by the INCREMENT amount of seconds until it reaches the MAXIMUM. For example, if the MINIMUM is 95 seconds, INCREMENT is 90 seconds and MAXIMUM is 270 seconds:
The first shutter is at 95 seconds, the MINIMUM. The second shutter is at 95 seconds + 185 seconds (95 seconds + 90 seconds). The third shutter is at 95 seconds + 185 seconds (second shutter) + 270 seconds (270 < 185 + 90). From then on a shutter will occur every 270 seconds.

Table 4-7 foc command

Subcommand	Argument	Description
auto	on | off	Enables or disables automatic fine offset calibration.
blade	-----	Calibrates with a blade.
scene	-----	Calibrates unit to scene.
reset	-----	Resets a hung FOC.
debug	-----	Prints debug information.

4.8 Flash Frame Capture Command (framecapture)

As many as 60 frames can be captured to memory and then selectively displayed according to an index number. The frames are indexed in the order which they were saved. The saved images can then be displayed instead of live imagery.

Root Command: framecapture

Sample ASCII Command: framecapture *print norm*

Sample Response:
Processing: framecapture print norm
Begin Stored Frames
7 64040000 n=8 p=-1
8 640e0000 n=-1 p=7
End Stored Frames
>

The sample response above shows that there are two captured frames with index numbers 7 and 8, and to display the frame captured with index 7 instead of live imagery:

ASCII Command: framecapture display 7

Response:
Processing: framecapture display 7
addr = 0x64040030
>

The address of the image is always the framecapture print norm address plus 0x30, and the numbers following the hexadecimal address printed by the print subcommand representing the previous and next image numbers.

Table 4-8 framecapture command

Subcommand	Argument	Description
capture	-----	Saves current frame to memory.
clear	------	Clears the entire area of Flash memory dedicated to imagery regardless of how many images are saved.
delall	-----	Deletes all saved frames.
delete	-----	Deletes frame that you are viewing from memory.
display	frame_index	Displays frame at index frame_index.
first	-----	Displays first image stored in flash
next	-----	Displays next image in flash
prev	-----	Displays previous image in flash
last	-----	Displays most recent picture stored to flash
live	-----	Returns to live imagery from displaying a saved frame
print	norm	Returns USB ASCII text with location and index number of all saved frames.

4.9 Set Frame Rate Command (framerate)

Change the default frame rate of the camera; the default frame rate is saved to flash memory and will be reverted to at startup, but does not change the current operating frame rate. Analog output supports both 30 and 60Hz in the TWV640 core and 7.5Hz in the TWV640i. USB output is supported on all of the frame rates in both cores.

Root Command: framerate

Sample ASCII Command: framerate *get*

Sample Response: Processing: framerate get
frameRate = 1
>

When using the framerate *get* command on a TWV640, a framerate of 0 represents 30 Hz, and a frameRate of 1 represents 60 Hz. If the framerate get command is used on TWV640i, 0 is always returned.

Table 4-9 framerate command

Subcommand	Argument	Description
default*	30 | 60*	Sets camera default (startup) frame rate to either 30 frames per second or 60 frames per second, does not change the current frame rate
get	-----	Gets number representing current frame rate.

* Not available in TWV640i.

4.10 Display Gamma Command (gamma)

This command allows manual gamma calculation mode if an eMagin VGA OLED display is used with the camera core. For more information on the purpose of gamma calculation or what the command should do, please refer to eMagin documentation.

This command allows manual gamma calculation mode.

Root Command: gamma

Sample ASCII Command: gamma *calc*

Sample Response:

```
Processing: gamma calc
Get Gamma Coeffs
gammaCoeff[0] = 428, grayLevel[0] = 255
gammaCoeff[1] = 479, grayLevel[1] = 128
gammaCoeff[2] = 517, grayLevel[2] = 64
gammaCoeff[3] = 571, grayLevel[3] = 32
gammaCoeff[4] = 645, grayLevel[4] = 16
gammaCoeff[5] = 740, grayLevel[5] = 8
gammaCoeff[6] = 857, grayLevel[6] = 4
gammaCoeff[7] = 1009, grayLevel[7] = 2
Update LUT
Write Gamma Table final to FPGA
Done!
```

Table 4-10 gamma command

Subcommand	Argument	Description
calc	-----	Calculates gamma table.
default	true | false	Gamma calc uses default values.
print	-----	Print the gamma table.
shape	[0-2] 1 for normal	Set gamma shape.
vgn	-----	Sample the VGN value.
wait	true | false	Set the gamma wait flag.

4.11 Parameter Command (param)

Set, read and save system parameter values. Parameters are normally used for product or user specific data and are stored in Flash memory so they survive a power cycle.

A parameter value set with the set command is only used in the current session. Parameters can be saved using the save command so the system boots with the saved parameter values on the next restart or power cycle.

Root Command: param

Sample ASCII Command: param *get MICRO_IR.FOC_FRM_MODE*

Sample Response: Processing: param get MICRO_IR.FOC_FRM_MODE
PARAM MICRO_IR.FOC_FRM_MODE = 0
>

Table 4-11 param command

Subcommand	Argument	Description
get	item_name	Returns value of parameter item_name.
set	item_name	Changes parameter item_name. Changes are not saved if unit is shut down.
	value	
save	-----	Saves all current parameters to Flash memory. If unit is restarted, parameters will be set to the current values.

4.12 Serial Number Command (serial_num)

Get the serial number of the core.

Root Command: serial_num

Sample ASCII Command: serial_num *get*

Sample Response: Processing: serial_num get
Serial Number = 367842
>

Table 4-12 serial_num command

Subcommand	Argument	Description
get	-----	Returns serial number of the unit.

4.13 Change Infrared Polarity Command (polarity)

The command changes the polarity of the imagery in terms of whether white or black represents higher infrared emittance. There is also a mode in which the camera outlines regions of the image with similar infrared emittance named “edge”. The edge mode does not allow switching between white-hot and black-hot modes, and displays the majority of the image as black and outlines similar emittance regions in white. In other words, edge only works in white-hot.

Root Command: polarity

Sample ASCII Command: polarity *get*

Sample Response: Processing: polarity get
Polarity is WHITE
>

Table 4-13 polarity command

Subcommand	Argument	Description
get	-----	Returns current polarity of camera.
set	white | black | edge	Sets system polarity to white, black or edge: • White: • Black: • Edge: Represents system mode that outlines regions with similar infrared emittance.
default	white | black	Sets default polarity of imagery to either white or black, the value is saved to Flash memory.
startup	-----	Run post startup polarity.

4.14 Version Command (version)

The version command returns the version of the different components of the camera.

Root Command: version

Sample ASCII Command: version *info*

Sample Response: Processing: version info
MicroIR Version Athena2 Lex 2.3.0, built on 09:29:32 May 2 2016 by Ace.Coder
>

Table 4-14 version command

Subcommand	Argument	Description
info	-----	Returns the version of the embedded software in the unit.
fpga	-----	Returns the version of the FPGA firmware in the unit.

4.15 Sharpness Command (sharpness)

Live imagery can be enhanced by either sharpening or introducing slight blurring. The default setting of the camera is no sharpness enhancement and currently there is no way to change the default.

Root Command: sharpness

Sample ASCII Command: sharpness *set 0*

Sample Response: Processing: sharpness set 0
>

Table 4-15 sharpness command

Subcommand	Argument	Description
set	0-4	Sets level of sharpness processing of the live imagery: • 0: Introduces slight imagery blurring. • 1: Leaves imagery as is. • 2 – 4: Introduces progressive level of sharpening.

4.16 Temperature Sensor Command (temp_sen)

The command returns the reading of the interface board temperature sensor in ASCII text. This command will only operate correctly if the provided interface board is used. If an interface isn't used, I2C errors will be returned.

Root Command: temp_sen

Sample ASCII Command: temp_sen *read*

Sample Response: Processing: temp_sen read
Amb Temp = 0x305 (48.312500 deg C)
>

To convert the hexadecimal value returned to a decimal temperature, first convert the hexadecimal into a decimal number (two's complement decode) and then multiply the result by 0.0625.

Table 4-16 temp_sen command

Subcommand	Argument	Description
read	-----	Returns interface board temperature sensor output.

4.17 Default video output command (video_output)

Sets up the unit to output desired video mode as default.

Root Command: video_ouput

Sample ASCII Command: video_output *default USB*

Sample Response: Processing: video_output default USB
>

Table 4-17 video_ouput command

Subcommand	Argument	Description
default	USB | ANALOG | EMAGIN | 16BIT | YCBCR | ANALOG_EMAGIN | ANALOG_16BIT | ANALOG_YCBCR	Sets default output video format.

4.18 Output Video to Analog Command (video_analog)

Sets up the unit to output to Analog video. This command is used to turn on BT. 656 signals to feed the ADV7391 analog encoder.

Root Command: video_analog

Sample ASCII Command: video_analog *default NTSC*

Sample Response: Processing: video_analog default NTSC
>

Table 4-18 video_analog command

Subcommand	Argument	Description
default	NTSC | PAL	Sets default output video format of Analog.
NTSC	-----	Sets analog video output format to NTSC.
PAL	-----	Sets analog video output format to PAL.
disable	-----	Disable analog video.

4.19 Output Video to USB Command (video_usb)

Sets up the unit to output to USB video and disables all other video modes.

The raw imagery from the camera core is a 640x480 resolution (VGA), 16-bits per pixel black and white. Color optimizations and customizations, for example overlays, require signal processing that makes the imagery 24-bits per pixel. To be able to display color optimized imagery or custom overlay, the mode of the imagery is 24-bits per pixel.

Root Command: video_usb

Sample ASCII Command: video_usb *auto_overlays*

Sample Response: Processing: video_usb auto_overlays
>

Table 4-19 video_usb command

Subcommand	Argument	Description
Manual	-----	Switches output video to 16-bit, non-uniformity corrected but not contrast enhanced video
Auto	-----	Switches output video to 16-bit auto-contrast mode.
auto_overlays	-----	Switches output video to 24-bit pixel mode to allow color or custom GUI overlays.
Default	AUTO | AUTO_OVERLAYS | MANUAL	Sets default output video format of USB.
Disable	-----	Disables video from being output over USB.

4.20 Output Video to 16-Bit Parallel Command (video_16bit)

Sets up the unit to output 16-bit parallel video. 16-bit output is only available when the core is running at 60Hz. This video option does not work in TWV640i configuration.

The imagery from the camera core is a 640x480 resolution (VGA), 16-bits per pixel black and white. There are two selections that provide 640x480; Auto and Manual. There is a raw video output option that provides 664x520 video.

Root Command: video_16bit

Sample ASCII Command: video_16bit *auto*

Sample Response: Processing: video_16bit auto
>

Table 4-20 video_16bit command

Subcommand	Argument	Description
auto	-----	Switches output video to 16-bit auto-contrast mode.
manual	-----	Switches output video to 16-bit, non-uniformity corrected but not contrast enhanced video
raw	-----	Switches output video to coarse-corrected, uncropped imagery that is not contrast enhanced
default	AUTO | MANUAL | RAW	Sets default output format of 16-bit video

4.21 Output Video to 8-bit YCbCr (video_ycbcr)

Sets up the unit to output 8-bit YCbCr video. The imagery from the camera core is a 640x480 resolution (VGA).

Root Command: video_ycbcr

Sample ASCII Command: video_ycbcr *enable*

Sample Response: Processing: video_ycbcr enable
>

Table 4-21 video_ycbcr command

Subcommand	Argument	Description
enable	-----	Switches output video to 8-bit YCbCr
disable	-----	Disables 8-bit YCbCr output

4.22 Output Video to eMagin (video_emagin)

Sets up the unit to output eMagin video. The imagery from the camera core is a 640x480 resolution (VGA).

Root Command:	video_emagin
Sample ASCII Command:	video_emagin *enable*
Sample Response:	Processing: video_emagin enable >

Table 4-22 video_emagin command

Subcommand	Argument	Description
enable	-----	Switches output video to eMagin
disable	-----	Disables eMagin output

4.23 Electronic Zoom Functionality Command (zoom)

Electronic zoom makes a selected region of the image the entire image displayed; currently 0.5x, 1x, 2x, 4x and 8x settings are supported.

Root Command: zoom

Sample ASCII Command: zoom *get*

Sample Response: Processing: zoom get
Centered on 320, 240, Zoom is 1X
>

Table 4-23 zoom command

Subcommand	Argument	Description
get	-----	Returns center point and current electronic zoom level unit is set to (0.5X, 1X, 2X, 4X, or 8X).
set	zoom_level x-center (optional) y-center (optional)	Sets window that electronic zoom displays as entire 640 by 480 image, with new image center set at (x-center, y-center) relative to upper left corner of imagery. Valid zoom level values – ".5x", "1x"," 2x", "4x" and "8x" If x-center or y-center are not entered, their defaults of 320 and 240 are used.

Table 4-24 zoom limits

Zoom Level	X Low Limit	X High Limit	Y Low Limit	Y High Limit
0.5X	160	480	120	360
1X	--	--	--	--
2X	160	480	120	360
4X	80	520	60	420
8X	40	600	30	450

4.24 Nanomotion Control Commands (nanomot)

Nanomotion is the manufacturer of the motor used to drive the shutter blade. This command allows the user to send Nanomotion commands directly to the shutter blade motor.

Root Command: nanomot

Sample ASCII Command: nanomot exec calibr

Sample Response: Processing: nanomot exec calibr
Shutter Calibration Done
>

Table 4-25 nanomot command

Subcommand	Argument	Description
exec	calibr | sparam | rparam | getvar	Execute Nanomotion command.
	<data>*	

*See Nanomotion documentation.

4.25 Shutter Control Commands (shutter)

Controls the operation of the FPA shutter blade. The shutter setup function is part of system startup.

Root Command: shutter

Sample ASCII Command: shutter drive open

Sample Response: Processing: shutter drive open
>

Table 4-26 shutter command

Subcommand	Argument	Description
drive	open | close	Open or Close the shutter.
setup	-----	Run shutter setup function.

4.26 Overlay Commands (overlay)

Demonstrate overlay capability. A “Hello World” overlay can be displayed, moved and cleared.

Root Command: overlay

Sample ASCII Command: overlay hello

Sample Response: Processing: overlay hello
>

Table 4-27 overlay command

Subcommand	Argument	Description
enable	true | false	Enable or disable.
hello	-----	Display “Hello World” overlay.
time	-----	Display “Hello World” for 10 seconds.
move	x	Move “Hello World”.
	y	
clear	-----	Clear “Hello World”.
refresh	true | false	Updates the overlay.

4.27 Crop Commands (crop)

This commands controls where you are shifting the crop of visible imagery. Imagery can only be shifted vertically by 10 up to rows in either direction.

Root Command: crop

Sample ASCII Command: crop *shift 10*

Sample Response: Processing: crop *shift 10*
>

Table 4-28 crop command

Subcommand	Argument	Description
shift	-10-10	Shifts cropped imagery by up to 10 rows up or down.

4.28 Test Pattern Commands (testpattern)

This commands allows you to display one of two different test patterns on any non-usb video mode that supports color imagery

Root Command: testpattern

Sample ASCII Command: testpattern *gradient*

Sample Response: Processing: testpattern *gradient*
>

Table 4-29 testpattern command

Subcommand	Argument	Description
color_bars	-----	Shows image of vertical color bars
gradient	-----	Shows a gradient from left to right of high-intensity color to black
off		Turns off the display of any test pattern being shown

4.29 Vertical Video Direction Control Commands (v_dir)

Vertical video direction control allows you to flip the imagery coming out of the unit vertically. This command works in all video modes except 16-bit modes.

Root Command:	v_dir
Sample ASCII Command:	v_dir *flip true*
Sample Response:	Processing: v_dir flip true >

Table 4-30 v_dir command

Subcommand	Argument	Description
flip	true | false	Enables or disables vertical flip of imagery

4.30 Horizontal Video Direction Control Commands (h_dir)

Horizontal video direction control allows you to flip the imagery coming out of the unit horizontally.

Root Command:	h_dir
Sample ASCII Command:	h_dir *flip true*
Sample Response:	Processing: h_dir flip true >

Table 4-31 h_dir command

Subcommand	Argument	Description
flip	true | false	Enables or disables horizontal flip of imagery

4.31 Delay Command Processing (delay)

Insert a delay of command processing in milliseconds.

Root Command: delay

Sample ASCII Command: delay 100

Sample Response: Processing: delay 100
>

Table 4-32 delay command

Argument	Description
ms	Delay time in milliseconds.

4.32 Help Command (help)

Root Command: help

Sample ASCII Command: help *framerate*

Sample Response: Processing: help framerate
FRAMERATE GET – Get the current frame rate
FRAMERATE DEFAULT – Set the device default frame rate
>

Table 4-33 help command

Subcommand	Argument	Description
-----	-----	Returns all available commands with short description of each command.
command	-----	Returns details about command along with available subcommands and arguments.

5. Command Quick Reference

See **Error! Reference source not found.** for a list of commands with their available subcommands and arguments. Refer to the section 3 for additional information.

Table 5-1: Command List

Commands	Subcommands and Arguments	Reference
adaptive_color	roi x-center y-center radius high r_component g_component b_component low r_component g_component b_component thresh high_threshold low_threshold enable true | false read	4.1
auto_ce_lvl	default new_gain_level restore get set new_gain_level window x-position y-position height width	4.2
mgml	enable disable autolevel true | false setthresh low_threshold high_threshold gainlevel get gainlevel set gain level gain + | - level + | - seed	4.3
color	enable true | false	4.4
usr_overlays	enable true | false hide overlay_name list move overlay_name x-position y-position movec overlay_name x-position y-position layer overlay_name layer refresh	4.5

Commands	Subcommands and Arguments	Reference
	show overlay_name timer overlay_name time_in_seconds	
display	dimctl final_idrf_value idrf final_idrf_value read register_address readp register_address shift x-offset y-offset startup write register_address data writep data	4.6
foc	auto off | on blade scene reset debug	4.7
framecapture	capture clear delall delete display frame_index first next prev last live print norm	4.8
framerate	default 30 | 60* get	4.9

Commands	Subcommands and Arguments	Reference
gamma	calc default print shape 0 | 1 | 2 vgn wait false | true	4.10
param	get item_name set item_name value save	4.11
serial_num	get	4.12
polarity	get set white | black | edge default white | black startup	4.13
version	info fpga	4.14
sharpness	set 0 | 1 | 2 | 3 | 4	4.15
temp_sen	read	4.16
video_output	default	4.17
video_analog	default ntsc pal disable	4.18
video_usb	auto auto_overlays default manual	4.19

Commands	Subcommands and Arguments	Reference
video_16bit*	manual auto raw default	4.20
video_ycbcr	enable disable	4.21
video_emagin	enable disable	4.22
zoom	get set zoom_level x-center y-center	4.23
nanomot	exec calibr | sparam | rparam | getvar data	4.24
shutter	drive open | close setup	4.25
overlay	enable false | true hello time move x y clear refresh	4.26
crop	shift number_rows	4.27
testpattern	color_bars gradient off	4.28
v_dir	flip true | false	4.29
h_dir	flip true | false	4.30
delay	ms	4.31
help	command	4.32

* Not available in TWV640i.

6. Parameters

Name	Description	Information
GAMMA.MONITOR_RATE	The rate to check for gamma changes at. Requires a reboot to take effect.	Data Type: int32_t Range: 0 (disables) to 10000 Default: 0 Engineering Units: ms
GAMMA.TS_EXT	Use external temp sensor in gamma monitor to determine when to recalculate gamma table	Data Type: int32_t Range: 0 for don’t use temp sensor, 1 for external temp sensor Default: 0 Engineering Units: boolean
GAMMA.TEMP_DIFF	The difference in temperature that requires a gamma change. Note: This will only operate if TS_EXT is a 1.	Data Type: int32_t Range: 0 to 100 Default: 10 Engineering Units: °C
GAMMA.LUM_DIFF	The difference in display brightness that requires a gamma change	Data Type: int32_t Range: 0 to 100 Default: 10 Engineering Units: %
MICRO_IR. FOC_FRM_MODE	Frame mode when FOC is activated	Data Type: int32_t Range: 0 for default legacy, 1 for freeze frm. See Enumeration type, FOC_Frm_Mode_t Default: 0 Engineering Units: None
MICRO_IR. DEFAULT_VIDEO	Default video output	Data Type: int32_t Range: 0 - USB, 1 - Analog, 2 - eMagin, 3 – YcbCr, 4 – 16-bit Parallel, 5 – Analog/eMagin, 6 – Analog/YCbCr, 7 – Analog/16-bit Default: 0 Engineering Units: None

MICRO_IR. EMAGIN_GRN_OUT	Default eMagin output bus **NOTE:** If this parameter is set to a 1, simultaneous outputs will not work	Data Type: int32_t Range: 0 – Red bus only, 1 - Red/Green bus Default: 0 Engineering Units: None
MICRO_IR. DEFAULT_CONTRAST	Default USB contrast mode	Data Type: int32_t Range: 0 for Manual, 1 for Auto, 2 for Auto Overlays Default: 1 Engineering Units: none
MICRO_IR. DEFAULT_16BIT_OUT	Default 16-bit parallel output mode	Data Type: int32_t Range: 0 for Raw, 1 for Manual, 2 for Auto Default: 1 Engineering Units: none
MICRO_IR. DEFAULT_CE_GAIN	Default contrast enhancement gain value.	Data Type: int32_t Range: 0 to 12 (0x0 to 0xC) Default: 5 Engineering Units: none
MICRO_IR. DEFAULT_CROP_SHIFT	Default crop shift value	Data Type: int32_t Range: -10 to 10 Default: 0 Engineering Units: Rows
MICRO_IR. COLOR_EN	Enable color LUT by default	Data Type: int32_t Range: 0 false, 1 true Default: 0 Engineering Units: none
MICRO_IR.V_DIR_FLIP	Enable vertical flip by default	Data Type: int32_t Range: 0 false, non-zero true Default: 0 Engineering Units: none

MICRO_IR.H_DIR_FLIP	Enable horizontal flip by default	Data Type: int32_t Range: 0 false, non-zero true Default: 0 Engineering Units: none
MICRO_IR. SHARPNESS	Value to initialize sharpness	Data Type: int32_t Range: 0 to 4 Default: 1 Engineering Units: 0 is blurred, 4 is very sharp
MICRO_IR. BAUD	Specifies the baud rate of the UART	Data Type: int32_t Range: 9600 * number Default: 0 Engineering Units: 9600 baud increments
MICRO_IR. MIN_SPLASH	Specifies the minimum time to display splash screen	Data Type: uint32_t Range: uint32 Default: 1000 Engineering Units: ms
MICRO_IR. ANALOG_MODE	Specifies the mode for analog video output, NTSC or PAL.	Data Type: int32_t Range: 0 (NTSC) or 1 (PAL) Default: 0 (NTSC) Engineering Units: boolean
ADV739X. SUBCAR_FREQ_NTSC	32-bit decimal representation of sub carrier frequency.	Data Type: int32_t Range: int32 Default: Engineering Units: hz
ADV739X. SUBCAR_FREQ_PAL	32-bit decimal representation of sub carrier frequency.	Data Type: int32_t Range: int32 Engineering Units: hz
LWNR. SCENE_EN	Enable scene threshold in LWNR.	Data Type: int32_t Range: 0 (disable) or 1 (enable) Default: 1 (enable) Engineering Units: boolean

LWNR. COLUMN_EN	Enable column threshold in LWNR.	Data Type: int32_t Range: 0 (disable) or 1 (enable) Default: 1 (enable) Engineering Units: boolean
LWNR. RANK_ORDER_EN	Enable rank order in LWNR.	Data Type: int32_t Range: 0 (disable) or 1 (enable) Default: 1 (enable) Engineering Units: boolean
MGML.GAIN_STEP	Amount to change gain value when gain +/- is used	Data Type: int32_t Range: 0 - 65535 Default: 1 Engineering Units: 16-bit gain counts
MGML.LEVEL_STEP	Amount to change level value when gain +/- is used	Data Type: int32_t Range: 0 - 65535 Default: 1 Engineering Units: 16-bit level counts
MGML.GAIN_MIN	Minimum gain value allowed to be set	Data Type: int32_t Range: 0 - 65535 Default: 0 Engineering Units: 16-bit gain counts
MGML.GAIN_MAX	Maximum gain value allowed to be set	Data Type: int32_t Range: 0 - 65535 Default: 65535 Engineering Units: 16-bit gain counts
MGML.LEVEL_MIN	Minimum level value allowed to be set	Data Type: int32_t Range: 0 - 65535 Default: 0 Engineering Units: 16-bit level counts
MGML.LEVEL_MAX	Maximum level value allowed to be set	Data Type: int32_t Range: 0 - 65535 Default: 65535 Engineering Units: 16-bit level counts

MGML.LEVEL_OFFSET_MIN	Minimum level offset value allowed to be set	Data Type: int32_t Range: -65535 - 65535 Default: -5000 Engineering Units: video counts
MGML.LEVEL_OFFSET_MAX	Maximum level offset value allowed to be set	Data Type: int32_t Range: -65535 - 65535 Default: 5000 Engineering Units: video counts

7. Data Interface

The TWV640 generic data interface is a pair of IN/OUT BULK endpoints. Data transactions are initiated with a header packet from the host computer to the TWV640. This header packet gets sent to the OUT endpoint before each read from the IN endpoint.

The header packet consists of the following fields:

All fields are 4 bytes wide in little endian order.

Name	Value
TYPE	0xA0
ADDRESS	0
SIZE	0xE1000 (24-bit) 0x96000 (16-bit)
RESERVED	0

After writing the header packet, read SIZE bytes from the IN endpoint to get a video frame.

The video type can be changed with the VIDEO_USB command