ON Semiconductor

Is Now

onsemi™

To learn more about onsemi™, please visit our website at www.onsemi.com

onsemi and onsemi and other names, marks, and brands are registered and/or common law trademarks of Semiconductor Components Industries, LLC dba "**onsemi**" or its affiliates and/or subsidiaries in the United States and/or other countries. **onsemi** owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of **onsemi** product/patent coverage may be accessed at www.onsemi.com/site/pdf/Patent-Marking.pdf. **onsemi** reserves the right to make changes at any time to any products or information herein, without notice. The information herein is provided "as-is" and **onsemi** makes no warranty, representation or guarantee regarding the accuracy of the information, product features, availability, functionality, or suitability of its products for any particular purpose, nor does **onsemi** assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using **onsemi** products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by **onsemi**. "Typical" parameters which may be provided in **onsemi** data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. **onsemi** does not convey any license under any of its intellectual property rights nor the rights of others. **onsemi** products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use **onsemi** products for any such unintended or unauthorized application, Buyer shall indemnify and hold **onsemi** and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that **onsemi** was negligent regarding the design or manufacture of the part. **onsemi** is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner. Other names and brands may be claimed as the property of others.

AND9338/D

High-Dynamic Range Image Signal Processor

ON Semiconductor®

www.onsemi.com

APPLICATION NOTE

ABOUT THIS DOCUMENT

This guide is a reference for hardware and software engineers developing camera systems using the ON Semiconductor AP0102AT image signal processor (ISP). The AP0102AT is ON Semiconductor's high−performance, ultra−low power in−line, digital image processor optimized for use with HDR (High Dynamic Range) sensors that integratesseamlessly in today's automotive applications.

The AP0102AT incorporates full auto−functions support (AWB and AE) and ALTM (Adaptive Local Tone Mapping) to enhance HDR images and advanced noise reduction, which enables excellent low−light performance; it is programmable through a serial interface.

This document provides information on hardware interfaces, camera control, and register programming recommendations to optimize image quality. The starting point for all tuning to use is ON Semiconductor recommended settings, which are loaded when the Demo Initialization preset is run within DevWare. The AP0102AT Datasheet, Register Reference, and Host Command Interface documents should be used along with this guide as a reference for specific register and programming information.

CONVENTIONS AND NOTATIONS

This developer guide follows the conventions and notations described below:

- Hexadecimal Numbers have 0x Prefix
- Binary Numbers have 0b Prefix
  Example: 0b1010 = 0xA
- Fixed Point Notation
  The notation is "integer−part.fractional_part" in bits.
  For example: 0.8 (zero integer bits and 8 fractional bits)
- Signed fixed point notation −8.0(0 through 0xF800)
- I/O signals can be LOW (0 or DGND), HIGH (1 or VDD_IO), or floating (high impedance or High−Z)
- Timing diagrams are not drawn to scale and do not necessarily illustrate the actual number of required clock cycles

GENERAL DESCRIPTION

The ON Semiconductor AP0102AT is a high−performance, ultra−low power in−line, digital image processor optimized for use with HDR (High Dynamic Range) sensors. The AP0102AT provides full auto−functions support (AWB and AE) and ALTM (Adaptive Local Tone Mapping) to enhance HDR images and advanced noise reduction which enables excellent low−light performance.

FUNCTIONAL OVERVIEW

Figure 1 shows the typical configuration of the AP0102AT in a camera system. On the host side, a two−wire serial interface is used to control the operation of the AP0102AT, and image data is transferred using the parallel interface between the AP0102AT and the host. The AP0102AT interface to the sensor also uses a parallel interface or HiSPi interface.

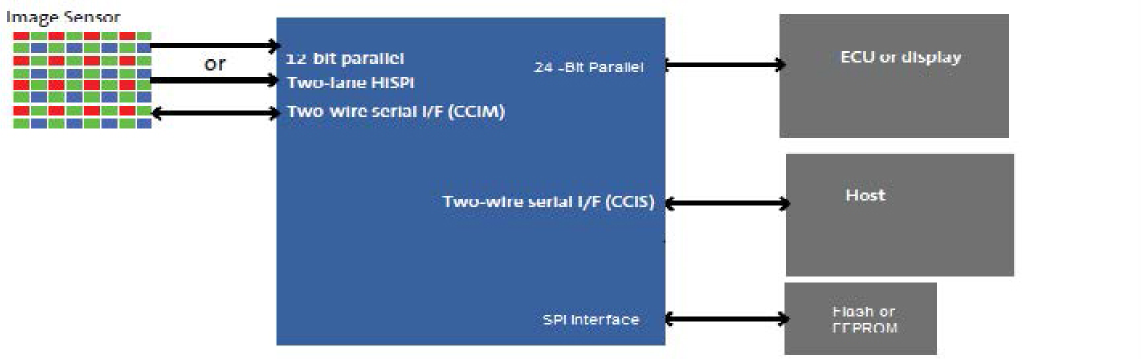
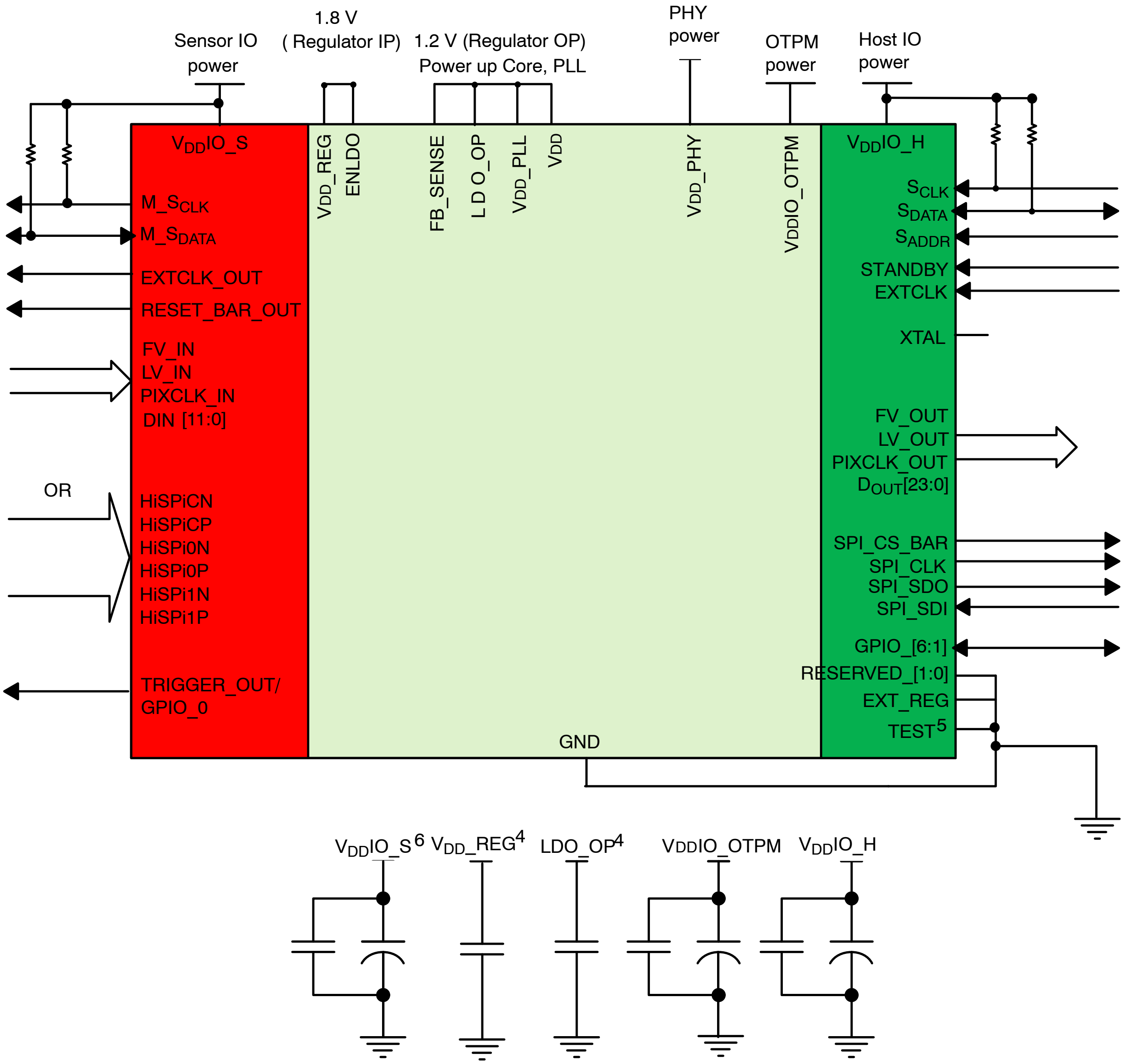


Figure 1. AP0102AT Connectivity

SYSTEM BLOCK DIAGRAM

Figure 2 shows typical AP0102AT device connections. All power supply rails must be decoupled from ground using capacitors as close as possible to the package.

The AP0102AT signals to the sensor and host interfaces can be at different supply voltage levels to optimize power consumption and maximize flexibility. Table 1 provides the signal descriptions for the AP0102AT.



1. This typical configuration shows only one scenario out of multiple possible variations for this device.
2. ON Semiconductor recommends a 1.5 kΩ resistor value for the two−wire serial interface R$_{PULL-UP}$; however, greater values may be used for slower transmission speed.
3. RESET_BAR has an internal pull−up resistor and can be left floating if not used.
4. The decoupling capacitors for the regulator input and output should have a value of 1.0 μF. The capacitors should be ceramic and need to have X5R or X7R dielectric.
5. TEST and RESERVED_[1:0] connect to GND for normal operation.
6. ON Semiconductor recommends that 0.1 μF and 1 μF decoupling capacitors for each power supply are mounted as close as possible to the pin. Actual values and numbers may vary depending on layout and design consideration.
7. The diagram is showing Legacy mode. If Crossbar is used, the 27 parallel outputs can be assigned to any pin. Refer to crossbar section for more details.

Figure 2. Typical Parallel Configuration

HiSPi and Parallel Connection

When using the HiSPi interface, connect the parallel input interface to GND.

When using the parallel input interface, it is recommended for the HiSPi interface to be connected to ground, and the power supply (VDD_PHY) to be connected to +2.8 V. Floating these pins is allowed as well.

Table 1. PIN DESCRIPTIONS

Name	Type	Description
EXTCLK	Input	Master input clock. This can either be a square−wave generated from an oscillator (in which case the XTAL input must be left unconnected) or direct connection to a crystal
XTAL	Output	If EXTCLK is connected to one pin of a crystal, the other pin of the crystal is connected to XTAL pin; otherwise this signal must be left unconnected
RESET_BAR	Input/PU	Master reset signal, active LOW. This signal has an internal pull up
S_{CLK}	Input	Two−wire slave serial interface clock (host interface)
S_{DATA}	I/O	Two−wire slave serial interface data (host interface)
S_{ADDR}	Input	Selects device address for the two−wire slave serial interface. When connected to GND the device ID is 0x90. When wired to VDDIO_H, a device ID of 0xBA is selected
FRAME_SYNC	Input	Pass through to TRIGGER_OUT. This signal should be connected to GND if not used
STANDBY	Input	Standby mode control, active HIGH
EXT_REG	Input	Select external regulator if tied high
ENDLO	Input	Regulator enable (V_{DD}_REG domain)
SPI_SCLK	Output	Clock output for interfacing to an external SPI flash or EEPROM memory
SPI_SDI	Input / PU	Data in from SPI flash or EEPROM memory. When no SPI device is fitted, this signal is used to determine whether the AP0102AT should auto−configure: 0: Do not auto−configure; Two−wire interface will be used to configure the device (host−config mode) 1: Auto−configure. This signal has an internal pull−up resistor. (Not supported for AR0230)
SPI_SDO	Output	Data out to SPI flash or EEPROM memory
SPI_CS_BAR	Output	Chip select out to SPI flash or EEPROM memory
EXT_CLK_OUT	Output	Clock to external sensor
RESET_BAR_OUT	Output	Reset signal to external signal
M_S_{CLK}	Output	Two−wire master serial interface clock (sensor interface)
M_S_{DATA}	I/O	Two−wire master serial interface clock (sensor interface)
FV_IN	Input	Sensor frame valid input
LV_IN	Input	Sensor line valid input
PIXCLK_IN	Input	Sensor pixel clock input
Din[11:0]	Input	Sensor pixel data input
HiSPiCN	Input	Differential HiSPi clock (negative)
HiSPiCP	Input	Differential HiSPi clock (positive)
HiSPi0N	Input	Differential HiSPi data, lane 0 (negative)
HiSPi0P	Input	Differential HiSPi data, lane 0 (positive)
HiSPi1N	Input	Differential HiSPi data, lane 1 (negative)
HiSPi1P	Input	Differential HiSPi data, lane 1 (positive)
TRIGGER_OUT/GPIO_0	Output	Trigger signal for external sensor
FV_OUT	Output	Host frame valid output (synchronous to PIXCLK_OUT)
META_LINE_VALID	Output	Line valid signal to indicate when Metadata is valid. In addition, there is an option to allow META_LINE_VALID to be reflected in LV_OUT

Table 1. PIN DESCRIPTIONS (continued)

Name	Type	Description
LV_OUT	Output	Host line valid output (synchronous to PIXCLK_OUT)
PIXCLK_OUT	Output	Host pixel clock output
D_{OUT}[23:0]	Output	Host pixel data output (synchronous to PIXCLK_OUT)
GPIO_[6:1]	I/O	General purpose digital I/O
TEST	Input	Must be tied to GND in normal operation
RESERVED_[1:0]	Input	Must be tied to GND in normal operation
V_{DD}IO_S	Supply	Sensor I/O power supply
V_{DD}IO_H	Supply	Host I/O power supply
V_{DD}_PLL	Supply	PLL supply
V_{DD}	Supply	Core supply
V_{DD}IO_OTPM	Supply	OTPM power supply
V_{DD}_PHY	Supply	PHY IO voltage for HiSPi
GND	Supply	Ground
V_{DD}_REG	Supply	Input to on−chip 1.8 V to 1.2 V regulator
LDO_OP	Output	Output from on chip 1.8 V to 1.2 V regulator
FB_SENSE	Input	On−chip regulator sense signal

Table 2. PACKAGE PINOUT

	1	2	3	4	5	6	7	8	9	10
A	RESERVED_0	V_{DD}IO_H	M_S_{DATA}	D_{IN}0	V_{DD}	D_{IN}5	D_{IN}10	LV_IN	V_{DD}IO_S	FV_IN
B	S_{CLK}	GPIO_6	EXTCLK _OUT	M_S_{CLK}	D_{IN}1	D_{IN}4	D_{IN}9	D_{IN}11	HiSPi1N	HiSPI1P
C	SPI_SCLK	RESERVED_1	S_{DATA}	GPIO_5	TRIGGER_OUT/GPIO_0	D_{IN}3	D_{IN}8	PIXCLK_IN	HiSPiCN	HiSPICP
D	SPI_SDO	SPI_SDI	SPI_CS_BAR	S_{ADDR}	RESET_BAR _OUT	D_{IN}2	D_{IN}7	V_{DD}_PHY	HiSPi0N	HiSPI0P
E	V_{DD}	GPIO_1	STANDBY	GND	GND	GND	Din6	GND	D_{GND}	V_{DD}IO_H
F	V_{DD}IO_OTPM	GPIO_2	GPIO_3	RESET_BAR	GND	GND	GND	EXTCLK	XTAL	V_{DD}
G	TEST	GPIO_4	FRAME_SYNC	LV_OUT	D_{OUT}16	D_{OUT}12	D_{OUT}5	EXT_REG	ENLDO	V_{DD}_PLL
H	META_LINE _VALID	FV_OUT	D_{OUT}21	D_{OUT}18	D_{OUT}14	D_{OUT}8	D_{OUT}6	D_{OUT}2	FB_SENSE	V_{DD}_REG
J	PIXCLK_OUT	D_{OUT}22	D_{OUT}19	D_{OUT}17	D_{OUT}13	D_{OUT}10	D_{OUT}7	D_{OUT}3	D_{OUT}0	LDO_OP
K	D_{OUT}23	V_{DD}IO_H	D_{OUT}20	D_{OUT}15	V_{DD}	D_{OUT}11	D_{OUT}9	D_{OUT}4	D_{OUT}1	GND

8. Pin C2 needs to be tied GND in all parallel out applications. (RESERVED[1]).
9. A1 and G1 to be tied to ground. (RESERVED[0] and TEST) for normal operation.

Crystal Usage

As an alternative to using an external oscillator, a crystal may be connected between EXTCLK and XTAL. Two small loading capacitors and a feedback resistor should be added, as shown in Figure 3.

A crystal oscillator with temperature compensation is recommended for applications that require high temperature operation.

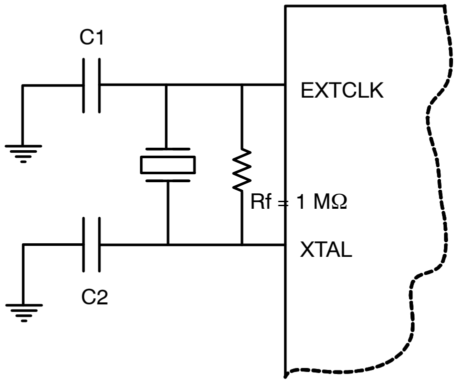


Figure 3. Using a Crystal Instead of an External Oscillator

Rf represents the feedback resistor, an Rf value of 1 MΩ would be sufficient for AP0102AT. C1 and C2 are decided according to the crystal or resonator CL specification. In the steady state of oscillation, CL is defined as (C1 × C2)/(C1 + C2). In fact, the I/O ports, the bond pad, package pin and PCB traces all contribute the parasitic capacitance to C1 and C2. Therefore, CL can be rewritten to be (C1* × C2*)/(C1* + C2*), where C1*= (C1 + CIN, stray) and C2*= (C2 + COUT, stray). The stray capacitance for the IO ports, bond pad and package pin are known which means the formulas can be rewritten as C1*= (C1 + 1.5 pF + CIN, PCB) and C2*= (C2 + 1.3 pF + COUT, PCB).

Table 3. OUTPUT STATES

	Hardware States		Firmware States				
Name	**Reset State**	**Default State**	**Hard Standby**	**Soft Standby**	**Streaming**	**Idle**	**Notes**
EXTCLK	(clock running or stopped)	(clock running)	(clock running or stopped)	(clock running)	(clock running)	(clock running)	Input
XTAL	n/a	n/a	n/a	n/a	n/a	n/a	Input
RESET_BAR	(asserted)	(negated)	(negated)	(negated)	(negated)	(negated)	Input
SCLK	n/a	n/a	(clock running or stopped)	(clock running or stopped)	(clock running or stopped)	(clock running or stopped)	Input. Must always be driven to a valid logic level
SDATA	High −impedance	High −impedance	High −impedance	High −impedance	High −impedance	High −impedance	Input/Output. A valid logic level should be established by pull−up
SADDR	n/a	n/a	n/a	n/a	n/a	n/a	Input. Must always be driven to a valid logic level
FRAME_SYNC	n/a	n/a	n/a	n/a	n/a	n/a	Input. Must always be driven to a valid logic level
STANDBY	n/a	(negated)	(asserted)	(negated)	(negated)	(negated)	Input. Must always be driven to a valid logic level
EXT_REG	n/a	n/a	n/a	n/a	n/a	n/a	Input. Must always be driven to a valid logic level
ENLDO	n/a	n/a	n/a	n/a	n/a	n/a	Input. Must be tied to VDD_REG or GND
SPI_SCLK	High −impedance	driven, logic 0	driven, logic 0	driven, logic 0			Output
SPI_SDI	Internal pull−up enabled	Internal pull−up enabled	Internal pull−up enabled	internal pull−up enabled			Input. Internal pull−up permanently enabled

Table 3. OUTPUT STATES (continued)

	Hardware States		Firmware States				
Name	Reset State	Default State	Hard Standby	Soft Standby	Streaming	Idle	Notes
SPI_SDO	High –impedance	driven, logic 0	driven, logic 0	driven, logic 0			Output
SPI_CS_BAR	High –impedance	driven, logic 1	driven, logic 1	driven, logic 1			Output
EXT_CLK_OUT	driven, logic 0	driven, logic 0	driven, logic 0	driven, logic 0			Output
RESET_BAR_OUT	driven, logic 0	driven, logic 0	driven, logic 1	driven, logic 1			Output. Firmware will release sensor reset
M_SCLK	High –impedance	High –impedance	High –impedance	High –impedance			Input/Output. A valid logic level should be established by pull–up
M_SDATA	High –impedance	High –impedance	High –impedance	High –impedance			Input/Output. A valid logic level should be established by pull–up
FV_IN ,LV_IN, PIXCLK_IN, DIN[11:0]	n/a	n/a	n/a	n/a	Dependent on interface used	n/a	Input. Must always be driven to a valid logic level
HiSPiCN	Disabled	Disabled	Dependent on interface used	Dependent on interface used	Dependent on interface used	Dependent on interface used	Input. Will be disabled and can be left floating
HiSPiCP							
HiSPi0N							
HiSPi0P							
HiSPi1N							
HiSPi1P							
FV_OUT, LV_OUT, MEAT_LINE_VALID, PIXCLK_OUT, DOUT[23:0]	High –impedance	Varied	Driven if used	Driven if used	Driven if used	Driven if used	Output. Default state dependent on configuration
GPIO[6:1]	High –impedance	Input, then high –impedance	Driven if used	Driven if used	Driven if used	Driven if used	Input/Output
TRIGGER_OUT	High –impedance	High –impedance	Driven if used	Driven if used	Driven if used	Driven if used	
TRST_BAR	n/a	n/a	(negated)	(negated)	(negated)	(negated)	Input. Must always be driven to a valid logic level

ON−CHIP REGULATOR

The AP0102AT has an on−chip regulator, the output from the regulator is 1.2 V and should only be used to power up the AP0102AT. It is possible to bypass the regulator and provide power to the relevant pins that need 1.2 V.

The following table summarizes the key signals when using/bypassing the regulator.

Table 4. KEY SIGNALS WHEN USING THE REGULATOR

Signal Name	Internal Regulator	External Regulator
V_{DD}_REG	1.8 V	Connect to V_{DD}IO_H
ENLDO	Connect to 1.8 V (V_{DD}_REG)	GND
FB_SENSE	1.2 V (input)	Float
LDO_OP	1.2 V (output)	Float
EXT_REG	GND	Connect to V_{DD}IO_H

Power−Up and Down Sequence

Powering up and down the AP0102AT requires voltages to be applied in a particular order, as seen in Figure 4. The timing requirements are shown in Table 5. The AP0102AT includes a power−on reset feature that initiates a reset upon power−up of the AP0102AT.

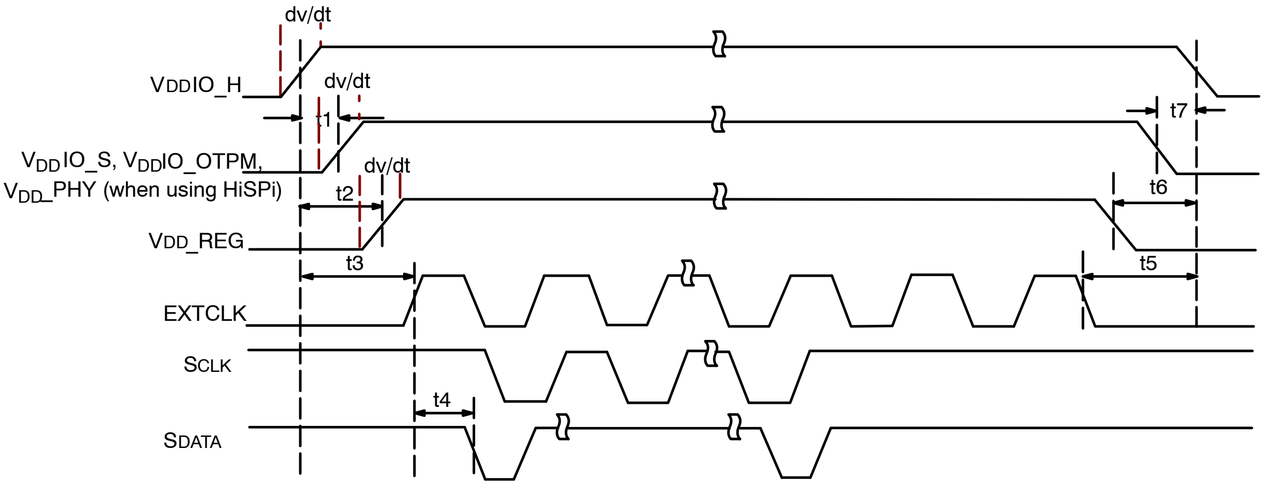


Figure 4. Power−Up and Power−Down Sequence

Table 5. POWER−UP AND POWER−DOWN SIGNAL TIMING

Symbol	Parameter	Min	Typ	Max	Unit
t1	Delay from V_{DDIO}_H to V_{DD}IO_S, V_{DD}IO_OTPM, V_{DD}_PHY (when using HiSPi)	0	–	50	ms
t2	Delay from V_{DD}IO_H to V_{DD}_REG	0	–	50	ms
t3	EXTCLK activation	t2 + 1	–	–	ms
t4	First serial command[1]	100	–	–	EXTCLK cycles
t5	EXTCLK cutoff	t6	–	–	ms
t6	Delay from V_{DD}_REG to V_{DD}IO_H	0	–	50	ms
t7	Delay from V_{DD}IO_S, V_{DD}IO_OTPM, V_{DD}_PHY (when using HiSPi) to V_{DD}IO_H	0	–	50	ms
dv/dt	Power supply ramp time (slew rate)	–	–	0.1	V/μs

10. When using XTAL the settling time should be taken into account.
11. RESET_BAR can be either high or low at power up.

REGISTERS AND VARIABLES OVERVIEW

This developer guide refers to various memory locations and registers that the host reads from or writes to for altering the device operation. Hardware registers may be read or written by sending the address and data information over the two−wire serial interface. A model of the AP0102AT is shown below. In some cases, device operation is not changed immediately; refresh or change−config must be commanded before changes are invoked.

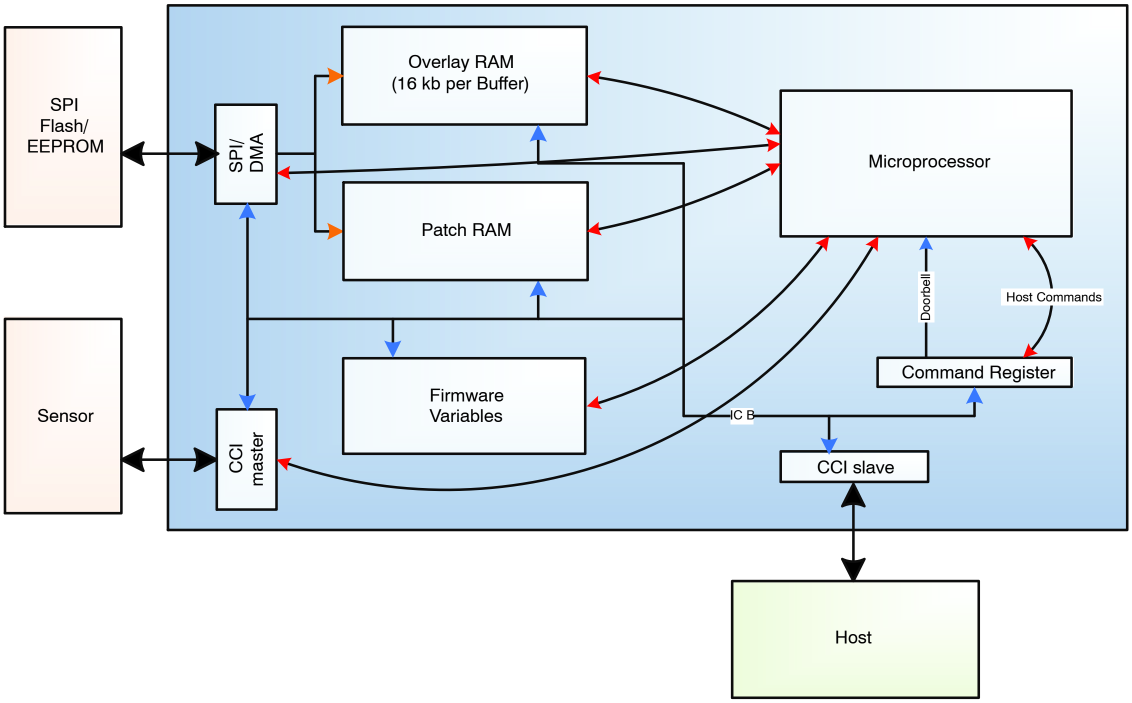


Figure 5. Host Command Interface Context

Registers

Registers can be accessed by the two−wire serial interface with addresses in the range 0x0000−0x7FFE. All registers are 16−bits in size and register access only supports 16−bit data read and write.

Variables

Variables correspond to locations in the memory space of the embedded microcontroller. Variables can be accessed by the two−wire serial interface with addresses in the range 0x8000−0xFFFF. Variables can be 8, 16 or 32−bit in size and variable access supports access of any 8−bit multiple.

Access Control

The Host can read and/or write firmware variables at any time, there is no ‘hard’ protection mechanism to prevent corruption caused by both the Host and the AP0102AT firmware writing to the same variable. Therefore, a cooperative ‘soft’ protection mechanism is employed; the Host and AP0102AT firmware agree on who can change which variables, and when.

Some variables are classified as ‘read−only’ − this means that the host should not write to them. Typically, these are status variables updated by the AP0102AT firmware, so a write by the Host will have no effect other than data−loss until the AP0102AT firmware updates the variable.

A variable classified as ‘read−write’ can be written by the Host. In most cases, the AP0102AT firmware will only read these variables.

Action type

When a change applied by the Host is acted upon by the firmware depends upon a variable’s ‘action−type’. The action types are described below.

1. After a Change−Config command has completed
2. After a Refresh command has completed
3. During Vertical blanking
4. Immediately

All variables documented in the register reference have an ‘action−type’ listed in their description.

Host Command Interface

The host issues a 16−bit command to the device by performing a register write to the command register (SYSCTL R0x0040). The host commands control many aspects of the system and more detail can be found in the Host Command Interface Specification document.

Configuration Changes

Change Config

Many of the configuration variables are only applied during a system configuration change. The Host can safely update variables of this action−type when a configuration change is not active with no effect on the currently output video. There is no actual ‘Change−Config’ command; the system configuration change is requested by the Host issuing the Set State command.

The Change−Config command performs the following operations:

1. Requests the sensor to enter standby
2. Waits until the sensor enters standby (this can take an entire frame time depending on when the command was issued)
3. When the sensor enters standby, reconfigures the AP0102AT subsystems and sensor
4. Restarts the sensor
5. Command completes

The Change−Config will result in up to two lost frames (depending on when the host sends the command), which is expected when the sensor configuration is changed.

In most real−world conditions, the host will issue Change−Config commands infrequently. Typical operation will power−up the device, write the desired configuration to the AP0102AT, issue the Change−Config command and then leave the part in streaming mode.

Refresh

The Refresh command is intended to refresh subsystems without requiring a sensor configuration change. Its primary purpose is to allow changes to multiple variables that need to be processed as a group.

The Refresh command performs the following:

1. Signals the Sequencer to refresh the system at the next EOF
2. On EOF, refresh the crop/scale settings, AE/AWB stats windows, and test−pattern settings

The sensor itself is unaffected, so no frames are lost. It is expected that the host will issue Refresh commands frequently.

Accessing External Sensor Registers

The AP0102AT firmware needs to communicate regularly with the sensor during streaming to control various aspects of the system. As the firmware is managing this, it means that the host should rarely need to access sensor registers using the CCIM interface and especially not when the sensor is streaming.

The host may need to access sensor registers during the configuration stage and this is achieved by using the CCIM host commands.

STANDBY AND RESET MODES

Hard Standby Mode

The AP0102AT can enter hard standby mode by using external STANDBY signal. In hard standby mode, the total power consumption is reduced. A further power reduction can be achieved by turning off the input clock, but this must be restored before de−asserting the STANDBY pin to LOW state to restart the device.

Entering Standby Mode

1. Assert STANDBY pin HIGH

Exiting Standby Mode

1. De−assert STANDBY pin LOW

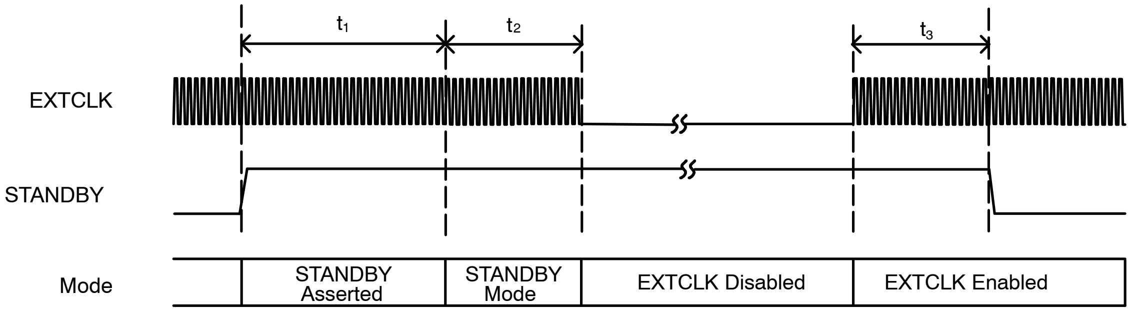


Figure 6. Hard Standby Operation

Table 6. HARD STANDBY SIGNAL TIMING

Symbol	Parameter	Min	Typ	Max	Unit
t_1	Standby entry complete	–	–	2	Frames
t_2	Active EXTCLK required after going into STANDBY mode	10	–	–	EXTCLKs
t_3	Active EXTCLK required before STANDBY de−asserted	10	–	–	EXTCLKs

Soft Standby

The system can enter a soft standby mode by issuing a host command.

In soft standby mode, the total power consumption is reduced. In this mode, the AP0102AT is powered down except the two−wire serial interface and parts of the system that control the clocks and interrupts. A further power reduction can be achieved by turning off the input clock, but this must be restored before issuing the host commands to restart the device.

Entering Soft Standby Mode

The system can enter a soft standby mode by issuing a host command interface (HCI) command, SET STATE.

1. SYSMGR_SET_STATE (STATE= SYS_STATE_ENTER_SOFT_STANDBY (0x51))
2. Wait for Host command to complete
3. OPTIONAL Turn off EXTCLK

Exiting Soft Standby Mode

The system can exit a soft standby mode by issuing a host command.

1. OPTIONAL: If EXTCLK was turned off then turn it back on
2. SYSMGR_SET_STATE(STATE = SYS_STATE_LEAVE_SOFT_STANDBY (0x55))
3. Wait for Host command to complete

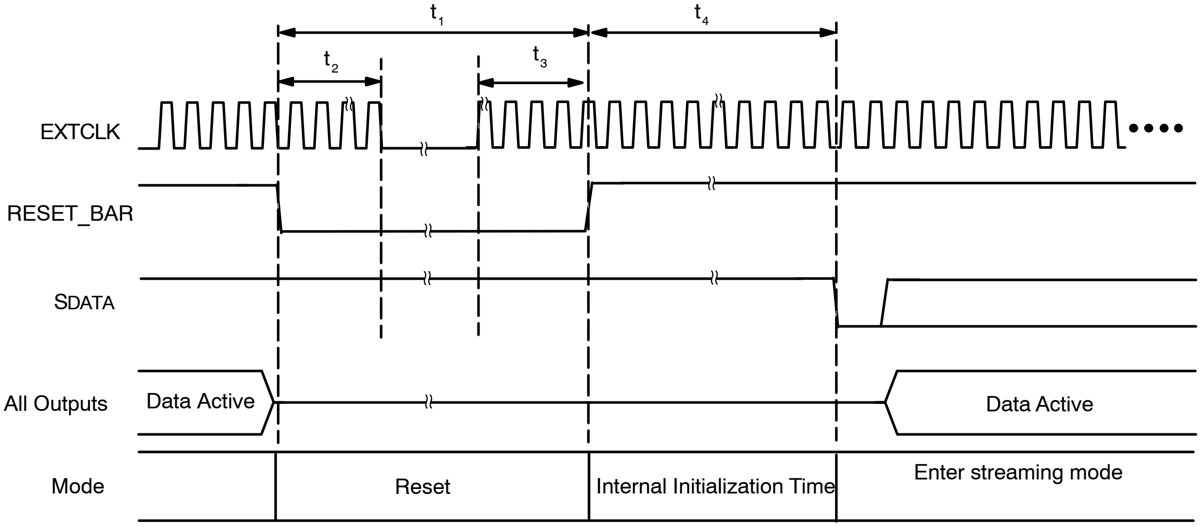
Reset

The AP0102AT has 3 types of reset available:

- A hard reset is issued by toggling the RESET_BAR signal
- A soft reset is issued by writing commands through the two−wire serial interface
- An internal power−on reset

Hard Reset

The AP0102AT enters the reset state when the external RESET_BAR signal is asserted LOW, as shown in Figure 5. All the output signals will be in High−Z state.



NOTE: This assumes auto−config.

Figure 7. Hard Reset Operation

Table 7. HARD RESET

Symbol	Definition	Min	Typ	Max	Unit
t_1	RESET_BAR pulse width	20	–	–	EXTCLK cycles
t_2	Active EXTCLK required after RESET_BAR asserted	10	–	–	
t_3	Active EXTCLK required before RESET_BAR de−asserted	10	–	–	
t_4	First two−wire serial interface communication after RESET_BAR is HIGH	100	–	–	

Soft Reset

A soft reset sequence to the AP0102AT can be activated by writing to a register through the two−wire serial interface. In soft reset mode, the two−wire serial interface is still active.

Example of entering and leaving soft reset:

1. Enter Soft Reset− Set SYSCTL 0x001A[0] to 0x1
2. Leaving Soft Reset− Set SYSCTL 0x001A[0] to 0x0

System Configuration and Usage Modes

Configuration Modes

At power−up or reset, the AP0102AT will enter a system configuration sequence to determine the correct operating mode. First, the AP0102AT enters the Flash Detection mode, which attempts to detect the presence of an SPI Flash or EEPROM device:

- If no SPI device is detected, the firmware then samples the SPI_SDI pin state to determine the next mode:
  - If SPI_SDI pin is low, then it enters the Host−Config mode
  - If SPI_SDI pin is high, then it enters the Auto−Config mode
- If an SPI device is detected, the device switches to the Flash−Config mode:
  - In the Flash−Config mode, the firmware interrogates the device to determine if it contains valid configuration records:
    If no records are detected, then the firmware enters the Host−Config mode.
    If records are detected, the firmware processes them. By default, when all Flash records are processed the firmware switches to the Host−Config mode. However, the records encoded into the Flash can optionally be used to instruct the firmware to proceed to auto−config, or to start streaming (via a Change−Config)
  - In the Host−Config mode, the firmware performs no further configuration changes, and remains idle waiting for configuration and commands from the host. The System Configuration phase is effectively complete and the AP0102AT will take no actions until the host issues commands
  - Change−Config. The firmware performs a ‘Change−Config’ operation. This applies the current configuration settings to the AP0102AT, and commences streaming

This completes the System Configuration phase.

Auto Configuration Mode

In the simplest case, the AP0102AT may operate in the Auto−Configuration mode with no customized settings. If flash config is skipped during system configuration, SPI_SDI will be sampled and if this pin is pulled high, the part will be configured for “Auto−Config”.

In the Auto−Config mode, the part will start streaming with the default settings stored in the AP0102AT internal memory.

NOTE: Because critical patches must be loaded, the Auto−Config mode is not recommended.

Flash Configuration Mode

In this mode, the AP0102AT can be configured by a serial EEPROM or Flash through the SPI Interface. Flash image sizes vary depending on the data for registers and firmware. To configure from SPI the EXTCLK must be in the range of 24−30 MHz.

This mode can be standalone (Flash boot only) or with a microprocessor in the system to communicate using the two−wire interface when to change things like overlays on the AP0102AT. When a microprocessor is in the system the host needs to know when the NVM has completed its download to the AP0102AT.

1. Poll Doorbell bit until its clear (0x0040[15])
2. Issue SYSMGR_GET_STATE repeatedly until it does not return EBUSY

When the SYSMGR_GET_STATE does not return EBUSY it indicates that the NVM download has completed.

Host Configuration Mode

In host−configuration mode, the AP0102AT is configured with settings directly passed by a host micro controller over the two−wire serial bus.

Below is the start up sequence for the Host Configuration mode (for example, SPI_SDI pin is tied low).

1. Poll Doorbell bit until its clear (0x0040[15])
2. Issue SYSMGR_GET_STATE repeatedly until it does not return EBUSY
3. Load and Apply patches (CRITICAL patches are MANDATORY and select FEATURE_RECOMMENDED patches which are required for the customer mode of operation)
4. Issue SENSOR_MGR_DISCOVER_SENSOR host command and poll until it returns ENOERR
5. Confirm that the expected sensor was found
6. Set SENSOR_MGR_SENSOR_DEFAULT_SEQUENCER_LOAD_INHIBIT = 1 to inhibit default Sequencer
7. Issue SENSOR_MGR_INITIALIZE_SENSOR host command and poll until it returns ENOERR
8. Load appropriate Sequence for the attached sensor (using CCIMGR host commands)
9. Load external sensor registers (if required)
10. Load configuration settings
11. Load tuning settings
12. Issue SYSMGR_SET_STATE (Change−Config)
13. Poll until it returns ENOERR
14. Issue SYSMGR_GET_STATE()
15. Check it returns SYS_STATE_STREAMING (0x31)

NOTE: 6−8 are optional.
If the user wishes to load the sequencer from image sensor then removes steps 7 and 8.

If the host is using an EXTCLK in the range of 24−30 MHz then the steps documented above should be followed. However if < 24 MHz is used then some additional steps are necessary.

1. Power−up (or hard reset)
2. Assert soft−reset as soon as possible
3. Set mcu_boot_pll_bypass = 1
4. De−assert soft−reset
5. Configure the correct PLL settings and enable the PLL by writing to the cam control variables and issuing a change−config
6. Follow previous sequence from step 3 in the above sequence

AP0102AT Demo Camera Hardware Jumper Settings for Configuration Modes

The printed circuit board jumpers for the AP0102AT demo system board have to be correctly set to be able to properly enter the desired configuration. The PCB will be marked with the revision of the PCB itself, but not of the AP0102AT.

Table 8. AP0102AT HEADBOARD JUMPER OPTIONS FOR PCB REVISION 1

Configuration	Jumper P5	Jumper P47	Jumper P58
Host	Short pin 1−2	Open	Short pin 2−3
Auto−config	Short pin 1−2	Short pin 1−2	Short pin 2−3
EEPROM	Open	Open	Short pin 2−3

In addition, the sensor printed circuit board must be configured to receive the EXTCLK_OUT from the AP0102AT. Table 9 specifies the correct jumper settings for correct operation.

Table 9. SENSOR PRINTED CIRCUIT BOARD JUMPER SETTINGS FOR USE WITH AP0102AT

Sensor	PCB Revision	PCB Jumper	PCB Jumper
AR0132−REV6	2	P19 (connect pins 2−3)	P20 (connect pins 1−2)
AR0230	0	P19 (connect pins 2−3)	NA
AR0231	1	P19 (connect pins 2−3)	NA
AR140 REV2/3	2	P19 (connect pins 2−3)	P20 (connect pins 1−2)

Flash/EEPROM Programming

The AP0102AT supports SPI nonvolatile memory (NVM) devices including flash and EEPROM for storing records and commands.

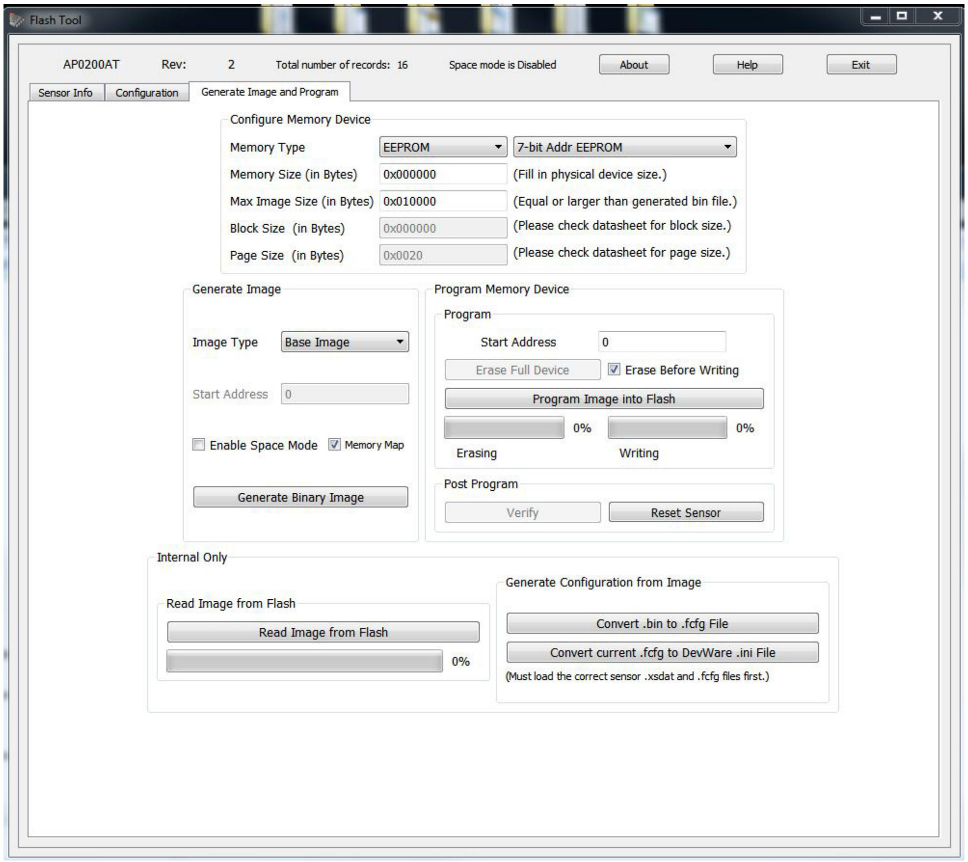
ON Semiconductor provides a FlashTool (C:\Aptina Imaging\FlashTool\flashtoolgui_new.exe) for programming the memory devices.

Supported SPI Devices

The supported devices are those that conform to the JEDEC−compliant programming interface. Please contact ON Semiconductor for specific design criteria and requirements. Maximum size that can be supported is 2 GB.

Programming NVM Devices Using the Flash Tool

Figure 8 shows the ‘Generate Image and Program’ interface tab of the flash configuration tool that can be accessed through the DevWare package located in the “ON Semiconductor\flashtool” path.



AP0200AT Rev: 2 Total number of records: 16 Space mode is Disabled

Memory Type: EEPROM, 7-bit Addr EEPROM

Memory Size (in Bytes): 0x000000 (Fill in physical device size.)

Max Image Size (in Bytes): 0x010000 (Equal or larger than generated bin file.)

Block Size (in Bytes): 0x000000 (Please check datasheet for block size.)

Page Size (in Bytes): 0x0020 (Please check datasheet for page size.)

Image Type: Base Image

Start Address: 0

□ Enable Space Mode ☑ Memory Map

Start Address: 0

☑ Erase Before Writing

0% 0%

0%

(Must load the correct sensor .xsdat and .fcfg files first.)

Figure 8. Generate Image and Program Tab

To properly program an NVM device in Flash tool, users need to specify the correct device information.

For “Memory type”, if a flash device is present, the user selects “Flash” in the first dropdown list and “Standard Flash” name from the second drop−down list.

The user then needs to enter the Memory Size, Maximum Image Size, Block Size and Page Size manually on the GUI. The “Maximum Image Size” is the physical size of the NVM device. Users may need to enter this manually if the specific NVM device is not in a drop−down list from Memory Type selection. The Block and page size can be derived from the NVM data sheet.

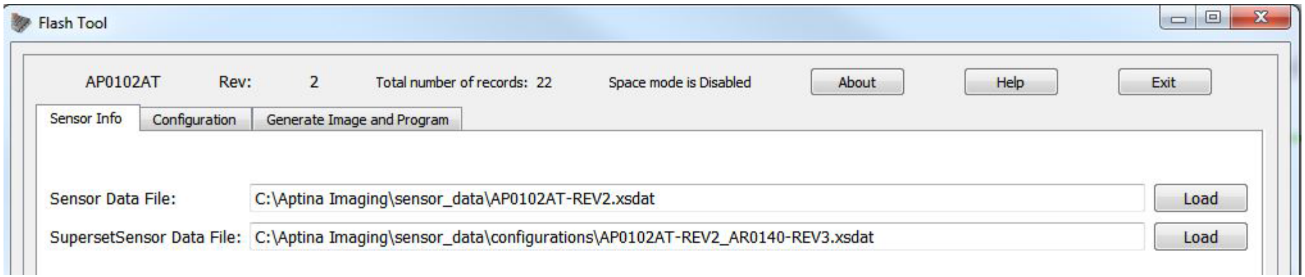
If an EEPROM device is present, the user selects “EEPROM” in the first drop−down list and the corresponding number of address bits from the second drop−down list (again, this is typically specified in the device’s data sheet).

Memory Size and Maximum Image Size are both required for user input.

SDAT (Sensor Data) Files

The AP0102AT has one XSDAT file for the AP0102AT.xsdat and another for the AP0102AT and attached sensor combination (for example, AP0102AT_AR0230.xsdat) which will contain additional variables required for the specific operation.

The device specific files are in the configurations directory and must be loaded for the flash tool to properly load the patches.



AP0102AT Rev: 2 Total number of records: 22 Space mode is Disabled

Sensor Data File: C:\Aptina Imaging\sensor_data\AP0102AT-REV2.xsdat

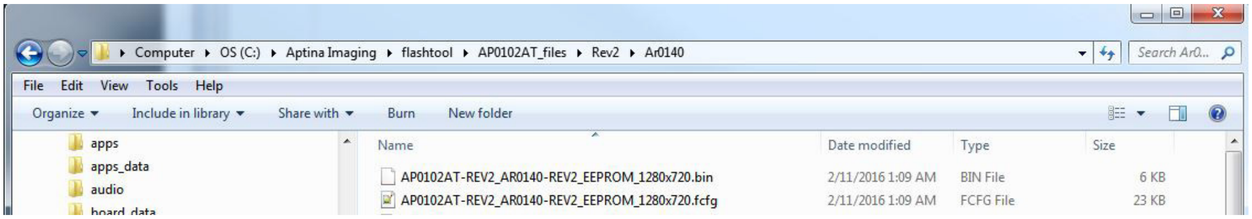
SupersetSensor Data File: C:\Aptina Imaging\sensor_data\configurations\AP0102AT-REV2_AR0140-REV3.xsdat

Figure 9. Configuration Path Example

Programming the NVM Device Using the Default Settings

To use the default settings provided for a specific AP0102AT combination:

1. Fill out the ‘Global settings’ page by loading in the XSDAT files (both standard XSDAT and sensor−specific)
2. Make sure the AP0102AT is set up for starting in NVM (EEPROM) mode (eg. correct jumpers on the demo board as shown in Table 10)

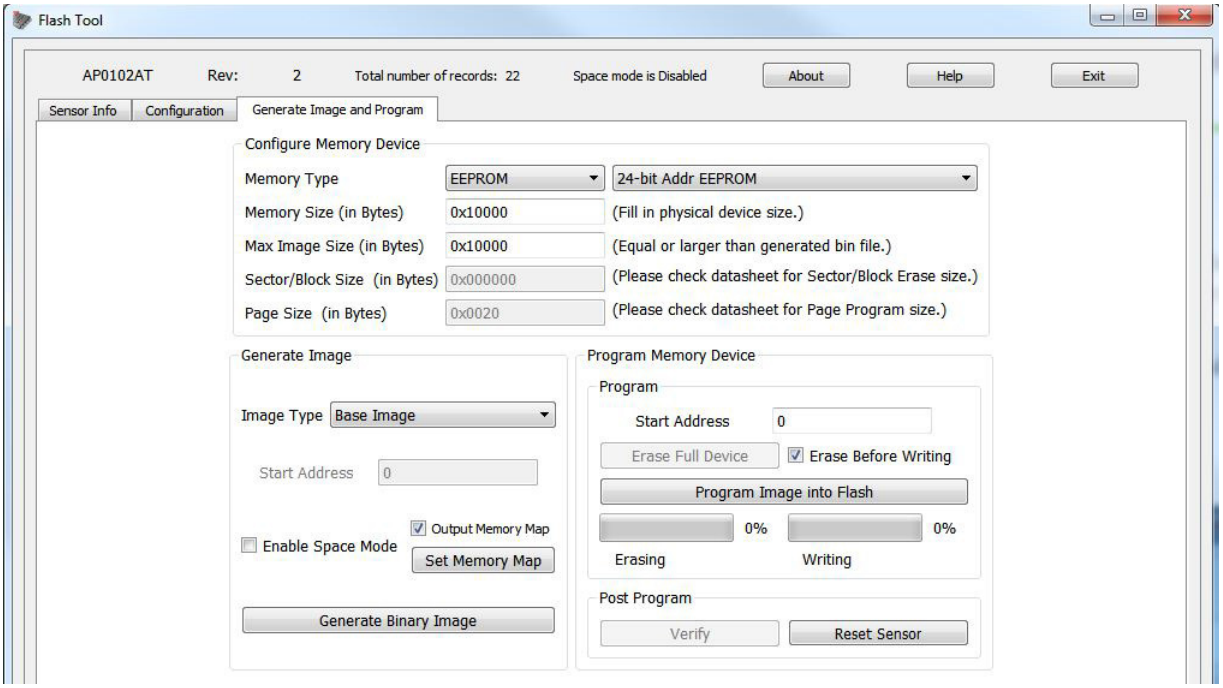


Computer ▸ OS (C:) ▸ Aptina Imaging ▸ flashtool ▸ AP0102AT_files ▸ Rev2 ▸ Ar0140

Name	Date modified	Type	Size
AP0102AT-REV2_AR0140-REV2_EEPROM_1280x720.bin	2/11/2016 1:09 AM	BIN File	6 KB
AP0102AT-REV2_AR0140-REV2_EEPROM_1280x720.fcfg	2/11/2016 1:09 AM	FCFG File	23 KB

3. Configure the headboard and part as needed for programming

4. Go to the *Generate Configuration and Program* tab. Select the proper options from the Configure memory device section and select *Program Image into Flash* and the programming should occur



AP0102AT Rev: 2 Total number of records: 22 Space mode is Disabled

Configure Memory Device

Memory Type: EEPROM; 24-bit Addr EEPROM

Memory Size (in Bytes): 0x10000 (Fill in physical device size.)

Max Image Size (in Bytes): 0x10000 (Equal or larger than generated bin file.)

Sector/Block Size (in Bytes): 0x000000 (Please check datasheet for Sector/Block Erase size.)

Page Size (in Bytes): 0x0020 (Please check datasheet for Page Program size.)

Generate Image

Image Type: Base Image

Start Address: 0

☐ Enable Space Mode

☑ Output Memory Map

Program Memory Device

Program

Start Address: 0

☑ Erase Before Writing

0% 0%

5. Press *Reset Sensor* **t**ab to reset the hardware

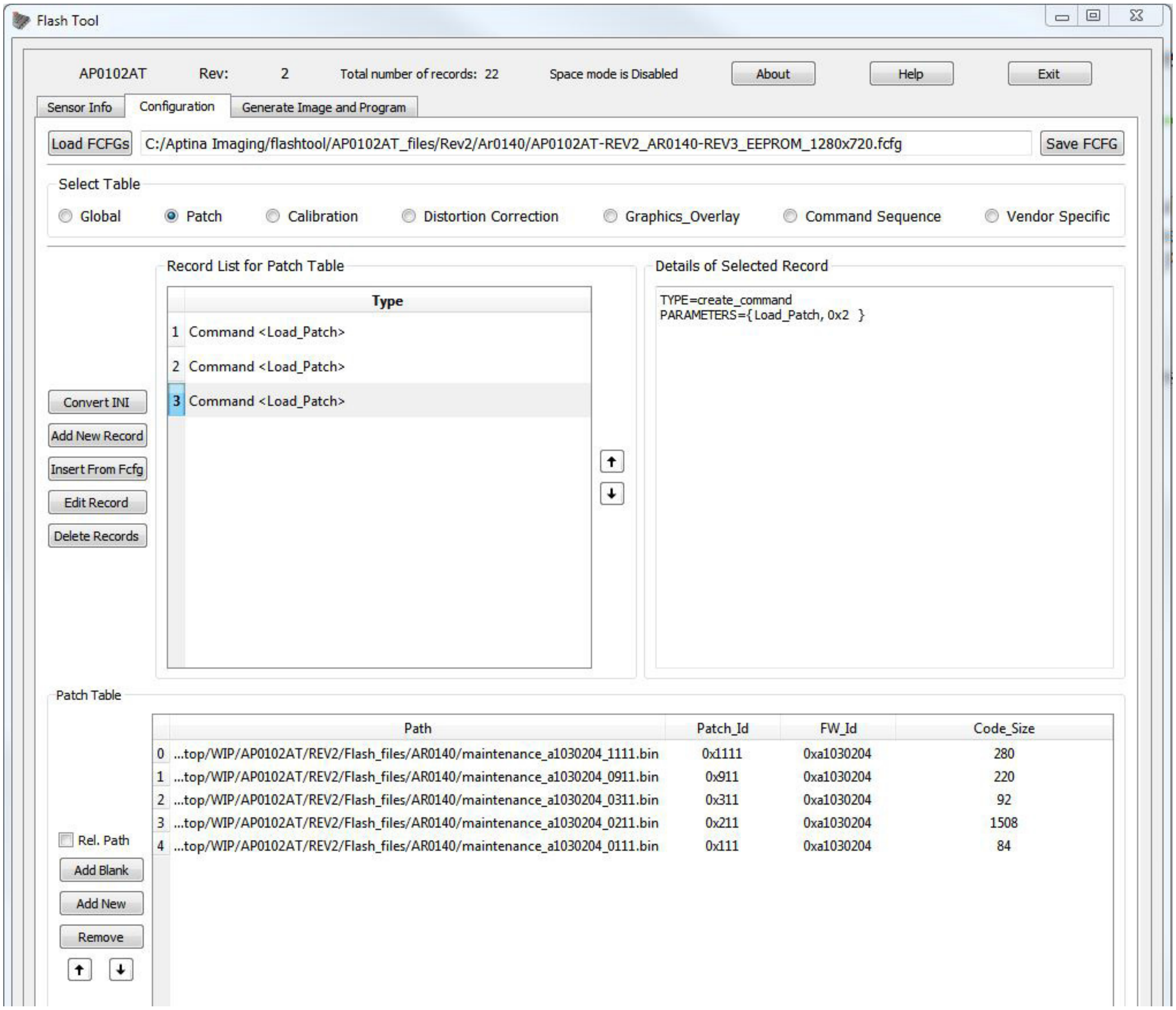
6. The part is ready to start from NVM

ISP PATCHES

In the case when Image System Processor (ISP) patches are required, they need to be included in Patch Table as shown in Figure 10 below. However, to include a patch in Patch Table does not mean it is automatically applied. In order to apply a patch, include the command “Load_Patch <Patch ID>” in “Initialization Table Setting → Patch”.

The patch ID corresponds to the number in Patch Table.

For information on applicable patches, refer to the Firmware Release Notes document.



AP0102AT Rev: 2 Total number of records: 22 Space mode is Disabled

C:/Aptina Imaging/flashtool/AP0102AT_files/Rev2/Ar0140/AP0102AT-REV2_AR0140-REV3_EEPROM_1280x720.fcfg

Select Table

○ Global ● Patch ○ Calibration ○ Distortion Correction ○ Graphics_Overlay ○ Command Sequence ○ Vendor Specific

Record List for Patch Table

	Type
1	Command <Load_Patch>
2	Command <Load_Patch>
3	Command <Load_Patch>

Details of Selected Record

```
TYPE=create_command
PARAMETERS={ Load_Patch, 0x2 }
```

Patch Table

□ Rel. Path

	Path	Patch_Id	FW_Id	Code_Size
0	...top/WIP/AP0102AT/REV2/Flash_files/AR0140/maintenance_a1030204_1111.bin	0x1111	0xa1030204	280
1	...top/WIP/AP0102AT/REV2/Flash_files/AR0140/maintenance_a1030204_0911.bin	0x911	0xa1030204	220
2	...top/WIP/AP0102AT/REV2/Flash_files/AR0140/maintenance_a1030204_0311.bin	0x311	0xa1030204	92
3	...top/WIP/AP0102AT/REV2/Flash_files/AR0140/maintenance_a1030204_0211.bin	0x211	0xa1030204	1508
4	...top/WIP/AP0102AT/REV2/Flash_files/AR0140/maintenance_a1030204_0111.bin	0x111	0xa1030204	84

Figure 10. Patch Table

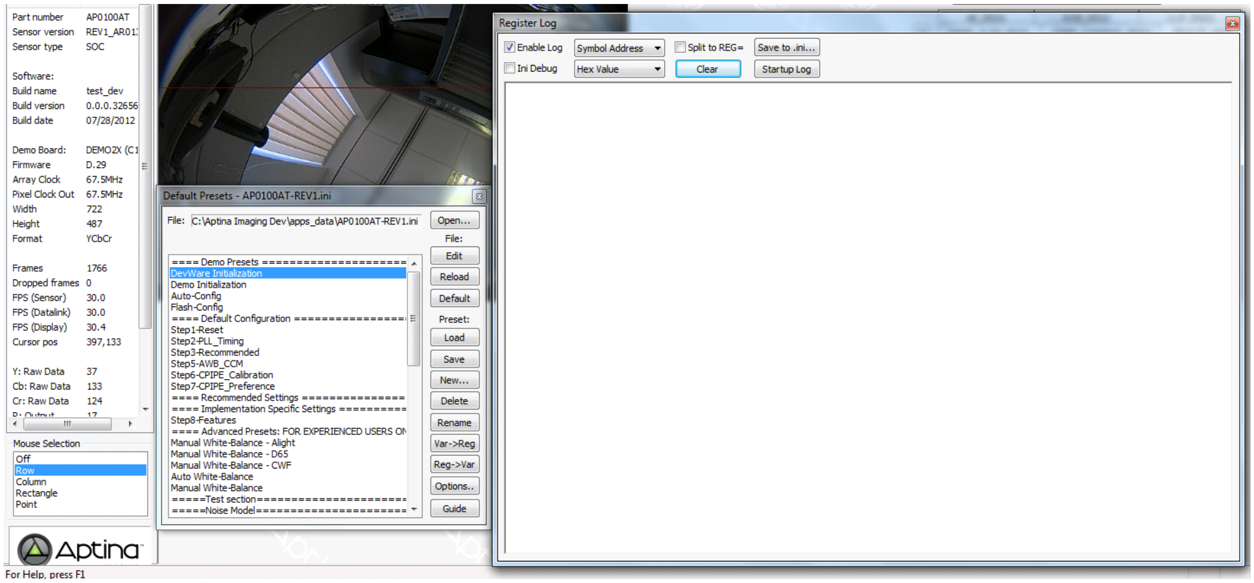
To program the NVM device using changed settings in initialization table:

1. Follow the procedure above to load the default settings
2. Initialization table can be modified to change settings along with pages described below
3. Save and program

PROCEDURE FOR UPDATING TUNING SETTINGS

As an example, run DevWare initialization to update the fcfg file

1. Use the register log to get the register settings from the default initialization file



```
Part number        AP0100AT
Sensor version     REV1_AR01
Sensor type        SOC

Software:
Build name         test_dev
Build version      0.0.0.32656
Build date         07/28/2012

Demo Board:        DEMO2X (C1
Firmware           D.29
Array Clock        67.5MHz
Pixel Clock Out    67.5MHz
Width              722
Height             487
Format             YCbCr

Frames             1766
Dropped frames     0
FPS (Sensor)       30.0
FPS (Datalink)     30.0
FPS (Display)      30.4
Cursor pos         397,133

Y: Raw Data        37
Cb: Raw Data       133
Cr: Raw Data       124
```

Mouse Selection: Off, Row, Column, Rectangle, Point

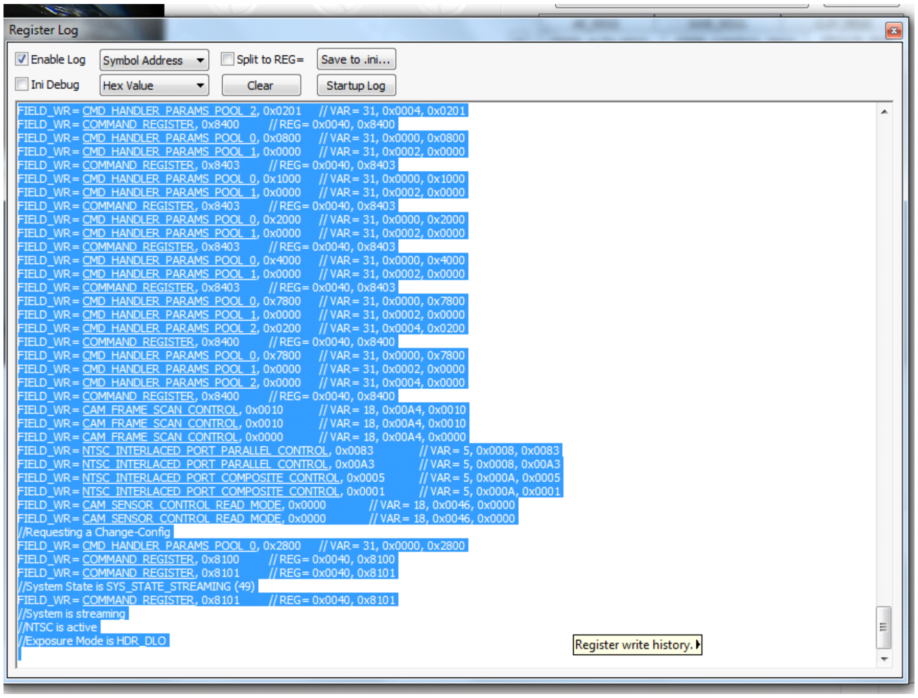
Default Presets - AP0100AT-REV1.ini

File: C:\Aptina Imaging Dev\apps_data\AP0100AT-REV1.ini

```
==== Demo Presets ======================
DevWare Initialization
Demo Initialization
Auto-Config
Flash-Config
==== Default Configuration ================
Step1-Reset
Step2-PLL_Timing
Step3-Recommended
Step5-AWB_CCM
Step6-CPIPE_Calibration
Step7-CPIPE_Preference
==== Recommended Settings ===============
==== Implementation Specific Settings ==========
Step8-Features
==== Advanced Presets: FOR EXPERIENCED USERS ON
Manual White-Balance - Alight
Manual White-Balance - D65
Manual White-Balance - CWF
Auto White-Balance
Manual White-Balance
=====Test section======================
=====Noise Model=======================
```

For Help, press F1

2. Copy the log to a text editor

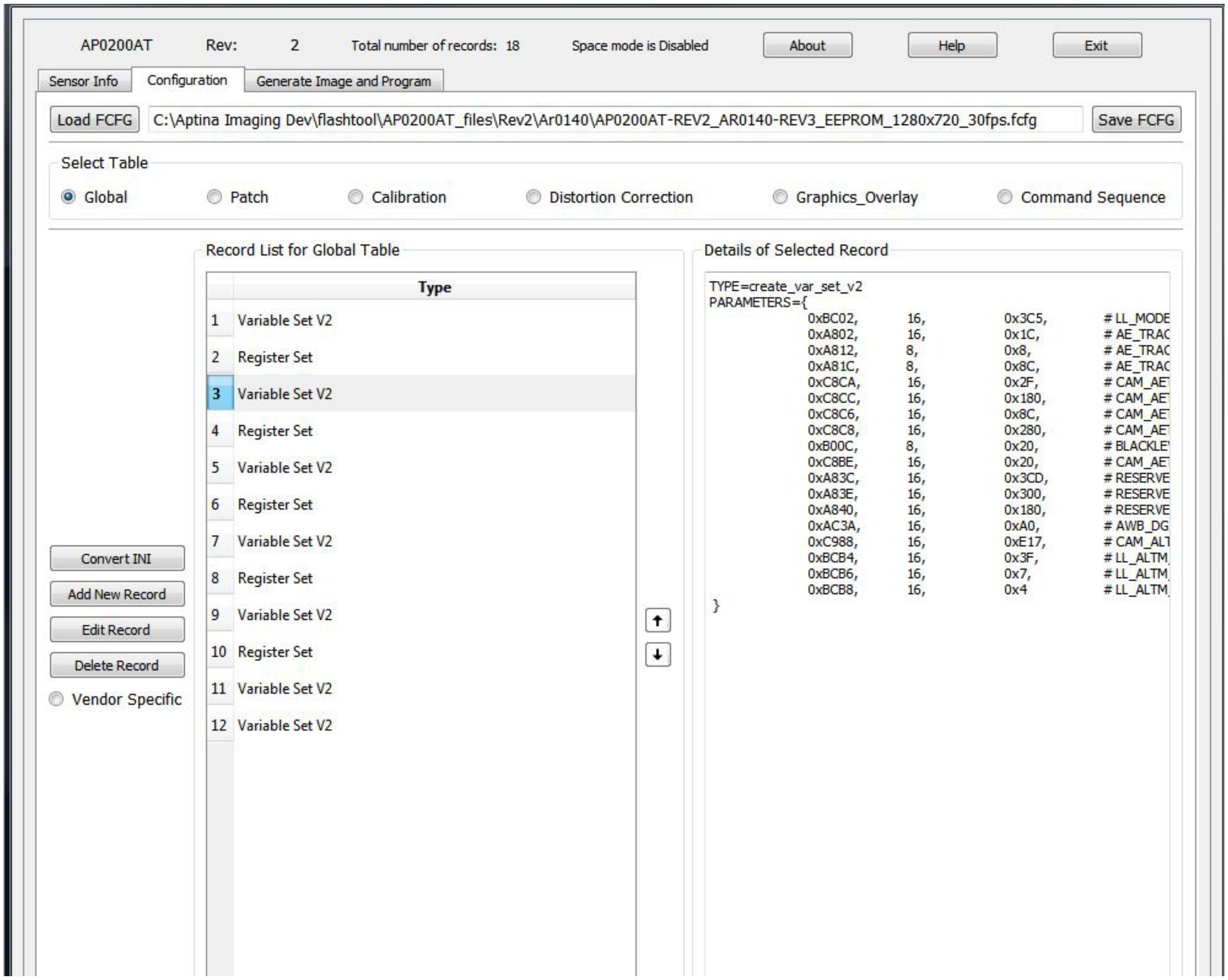


```
FIELD_WR= CMD_HANDLER_PARAMS_POOL_2, 0x0201     // VAR= 31, 0x0004, 0x0201
FIELD_WR= COMMAND_REGISTER, 0x8400       // REG= 0x0040, 0x8400
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x0800     // VAR= 31, 0x0000, 0x0800
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0x0000     // VAR= 31, 0x0002, 0x0000
FIELD_WR= COMMAND_REGISTER, 0x8403       // REG= 0x0040, 0x8403
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x1000     // VAR= 31, 0x0000, 0x1000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0x0000     // VAR= 31, 0x0002, 0x0000
FIELD_WR= COMMAND_REGISTER, 0x8403       // REG= 0x0040, 0x8403
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x2000     // VAR= 31, 0x0000, 0x2000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0x0000     // VAR= 31, 0x0002, 0x0000
FIELD_WR= COMMAND_REGISTER, 0x8403       // REG= 0x0040, 0x8403
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x4000     // VAR= 31, 0x0000, 0x4000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0x0000     // VAR= 31, 0x0002, 0x0000
FIELD_WR= COMMAND_REGISTER, 0x8403       // REG= 0x0040, 0x8403
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x7800     // VAR= 31, 0x0000, 0x7800
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0x0000     // VAR= 31, 0x0002, 0x0000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_2, 0x0200     // VAR= 31, 0x0004, 0x0200
FIELD_WR= COMMAND_REGISTER, 0x8400       // REG= 0x0040, 0x8400
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x7800     // VAR= 31, 0x0000, 0x7800
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0x0000     // VAR= 31, 0x0002, 0x0000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_2, 0x0000     // VAR= 31, 0x0004, 0x0000
FIELD_WR= COMMAND_REGISTER, 0x8400       // REG= 0x0040, 0x8400
FIELD_WR= CAM_FRAME_SCAN_CONTROL, 0x0010        // VAR= 18, 0x00A4, 0x0010
FIELD_WR= CAM_FRAME_SCAN_CONTROL, 0x0010        // VAR= 18, 0x00A4, 0x0010
FIELD_WR= CAM_FRAME_SCAN_CONTROL, 0x0000        // VAR= 18, 0x00A4, 0x0000
FIELD_WR= NTSC_INTERLACED_PORT_PARALLEL_CONTROL, 0x0083         // VAR= 5, 0x0008, 0x0083
FIELD_WR= NTSC_INTERLACED_PORT_PARALLEL_CONTROL, 0x00A3         // VAR= 5, 0x0008, 0x00A3
FIELD_WR= NTSC_INTERLACED_PORT_COMPOSITE_CONTROL, 0x0005        // VAR= 5, 0x000A, 0x0005
FIELD_WR= NTSC_INTERLACED_PORT_COMPOSITE_CONTROL, 0x0001        // VAR= 5, 0x000A, 0x0001
FIELD_WR= CAM_SENSOR_CONTROL_READ_MODE, 0x0000          // VAR= 18, 0x0046, 0x0000
FIELD_WR= CAM_SENSOR_CONTROL_READ_MODE, 0x0000          // VAR= 18, 0x0046, 0x0000
//Requesting a Change-Config
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0x2800     // VAR= 31, 0x0000, 0x2800
FIELD_WR= COMMAND_REGISTER, 0x8100       // REG= 0x0040, 0x8100
FIELD_WR= COMMAND_REGISTER, 0x8101       // REG= 0x0040, 0x8101
//System State is SYS_STATE_STREAMING (49)
FIELD_WR= COMMAND_REGISTER, 0x8101       // REG= 0x0040, 0x8101
//System is streaming
//NTSC is active
//Exposure Mode is HDR_DLO
```

NOTE: Command sequences cannot be captured this way. This method is only for tuning parameters settings.

3. In the text editor, delete the patches, sequencer, sensor settings and host commands which will leave a small set of changes

4. Delete existing tuning settings (1−11)



AP0200AT Rev: 2 Total number of records: 18 Space mode is Disabled

C:\Aptina Imaging Dev\flashtool\AP0200AT_files\Rev2\Ar0140\AP0200AT-REV2_AR0140-REV3_EEPROM_1280x720_30fps.fcfg

Select Table

● Global ○ Patch ○ Calibration ○ Distortion Correction ○ Graphics_Overlay ○ Command Sequence

○ Vendor Specific

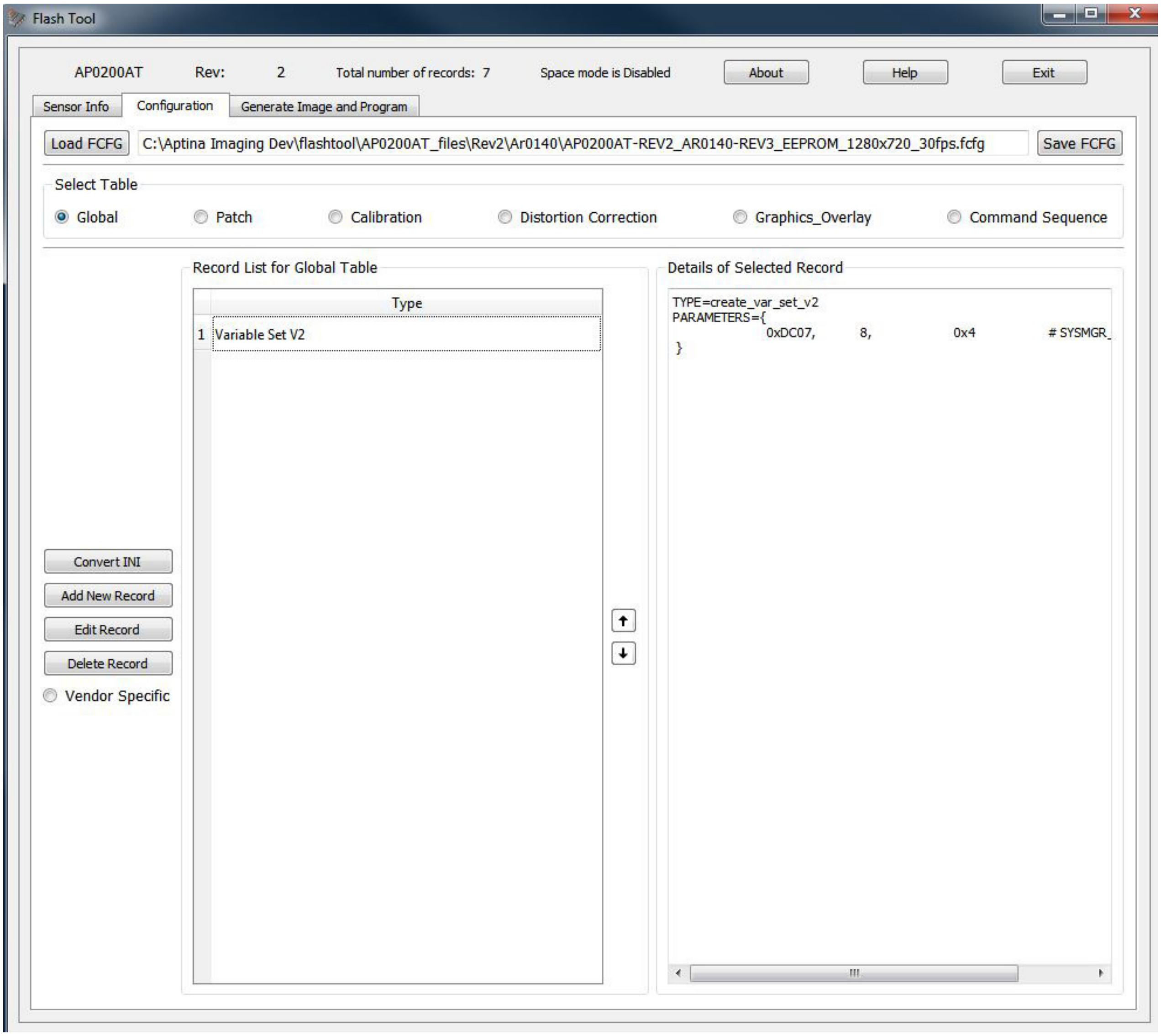
Record List for Global Table

	Type
1	Variable Set V2
2	Register Set
3	Variable Set V2
4	Register Set
5	Variable Set V2
6	Register Set
7	Variable Set V2
8	Register Set
9	Variable Set V2
10	Register Set
11	Variable Set V2
12	Variable Set V2

Details of Selected Record

```
TYPE=create_var_set_v2
PARAMETERS={
               0xBC02,     16,         0x3C5,      # LL_MODE
               0xA802,     16,         0x1C,       # AE_TRAC
               0xA812,     8,          0x8,        # AE_TRAC
               0xA81C,     8,          0x8C,       # AE_TRAC
               0xC8CA,     16,         0x2F,       # CAM_AET
               0xC8CC,     16,         0x180,      # CAM_AET
               0xC8C6,     16,         0x8C,       # CAM_AET
               0xC8C8,     16,         0x280,      # CAM_AET
               0xB00C,     8,          0x20,       # BLACKLE
               0xC8BE,     16,         0x20,       # CAM_AET
               0xA83C,     16,         0x3CD,      # RESERVE
               0xA83E,     16,         0x300,      # RESERVE
               0xA840,     16,         0x180,      # RESERVE
               0xAC3A,     16,         0xA0,       # AWB_DG
               0xC988,     16,         0xE17,      # CAM_ALT
               0xBCB4,     16,         0x3F,       # LL_ALTM_
               0xBCB6,     16,         0x7,        # LL_ALTM_
               0xBCB8,     16,         0x4         # LL_ALTM_
}
```

5. What would be seen is:



AP0200AT Rev: 2 Total number of records: 7 Space mode is Disabled

C:\Aptina Imaging Dev\flashtool\AP0200AT_files\Rev2\Ar0140\AP0200AT-REV2_AR0140-REV3_EEPROM_1280x720_30fps.fcfg

Select Table

● Global ○ Patch ○ Calibration ○ Distortion Correction ○ Graphics_Overlay ○ Command Sequence

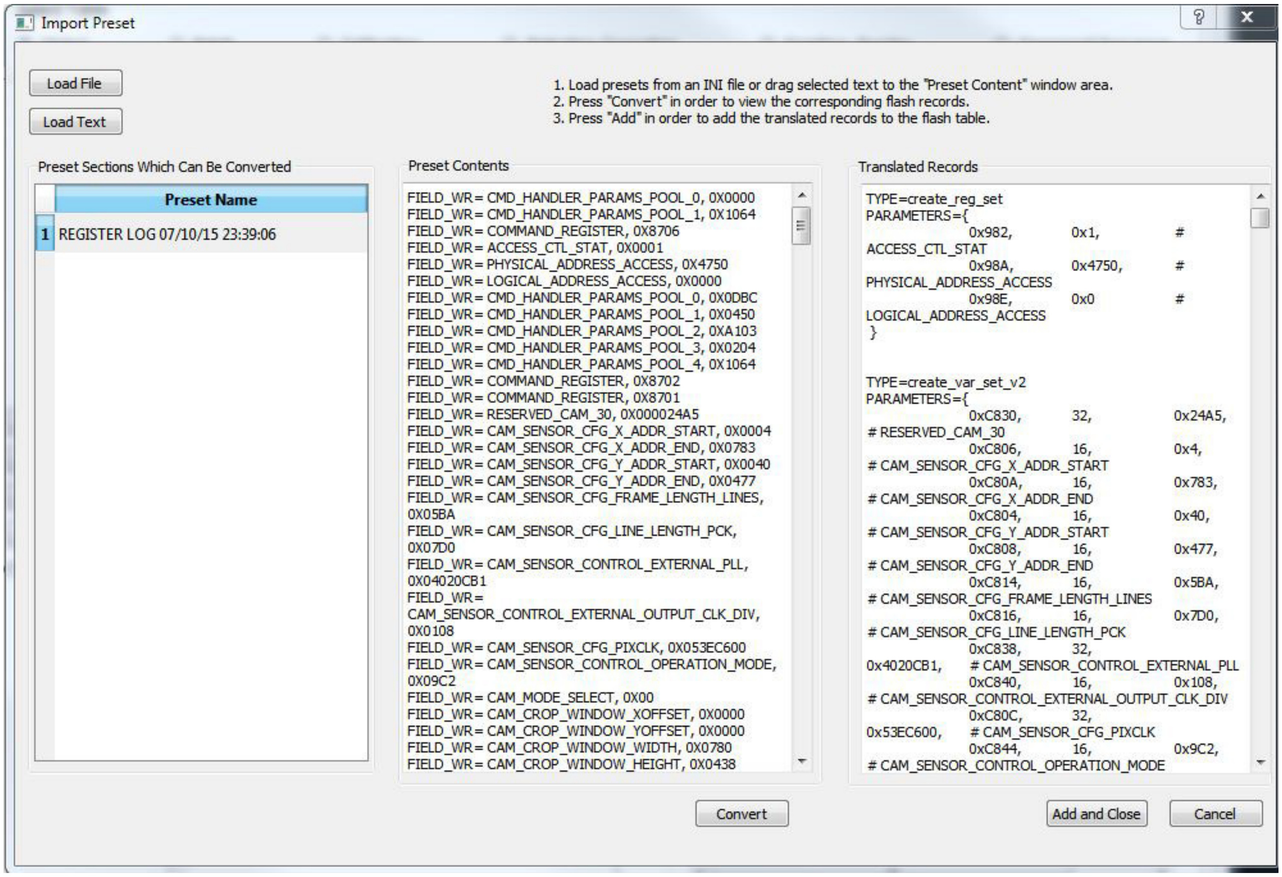
Record List for Global Table

	Type
1	Variable Set V2

Details of Selected Record

```
TYPE=create_var_set_v2
PARAMETERS={
                0xDC07,      8,          0x4          # SYSMGR_
}
```

6. Import this into the Flash Tool:



1. Load presets from an INI file or drag selected text to the "Preset Content" window area.
2. Press "Convert" in order to view the corresponding flash records.
3. Press "Add" in order to add the translated records to the flash table.

Preset Sections Which Can Be Converted

	Preset Name
1	REGISTER LOG 07/10/15 23:39:06

Preset Contents

```
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0X0000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0X1064
FIELD_WR= COMMAND_REGISTER, 0X8706
FIELD_WR= ACCESS_CTL_STAT, 0X0001
FIELD_WR= PHYSICAL_ADDRESS_ACCESS, 0X4750
FIELD_WR= LOGICAL_ADDRESS_ACCESS, 0X0000
FIELD_WR= CMD_HANDLER_PARAMS_POOL_0, 0X0DBC
FIELD_WR= CMD_HANDLER_PARAMS_POOL_1, 0X0450
FIELD_WR= CMD_HANDLER_PARAMS_POOL_2, 0XA103
FIELD_WR= CMD_HANDLER_PARAMS_POOL_3, 0X0204
FIELD_WR= CMD_HANDLER_PARAMS_POOL_4, 0X1064
FIELD_WR= COMMAND_REGISTER, 0X8702
FIELD_WR= COMMAND_REGISTER, 0X8701
FIELD_WR= RESERVED_CAM_30, 0X000024A5
FIELD_WR= CAM_SENSOR_CFG_X_ADDR_START, 0X0004
FIELD_WR= CAM_SENSOR_CFG_X_ADDR_END, 0X0783
FIELD_WR= CAM_SENSOR_CFG_Y_ADDR_START, 0X0040
FIELD_WR= CAM_SENSOR_CFG_Y_ADDR_END, 0X0477
FIELD_WR= CAM_SENSOR_CFG_FRAME_LENGTH_LINES,
0X05BA
FIELD_WR= CAM_SENSOR_CFG_LINE_LENGTH_PCK,
0X07D0
FIELD_WR= CAM_SENSOR_CONTROL_EXTERNAL_PLL,
0X04020CB1
FIELD_WR=
CAM_SENSOR_CONTROL_EXTERNAL_OUTPUT_CLK_DIV,
0X0108
FIELD_WR= CAM_SENSOR_CFG_PIXCLK, 0X053EC600
FIELD_WR= CAM_SENSOR_CONTROL_OPERATION_MODE,
0X09C2
FIELD_WR= CAM_MODE_SELECT, 0X00
FIELD_WR= CAM_CROP_WINDOW_XOFFSET, 0X0000
FIELD_WR= CAM_CROP_WINDOW_YOFFSET, 0X0000
FIELD_WR= CAM_CROP_WINDOW_WIDTH, 0X0780
FIELD_WR= CAM_CROP_WINDOW_HEIGHT, 0X0438
```

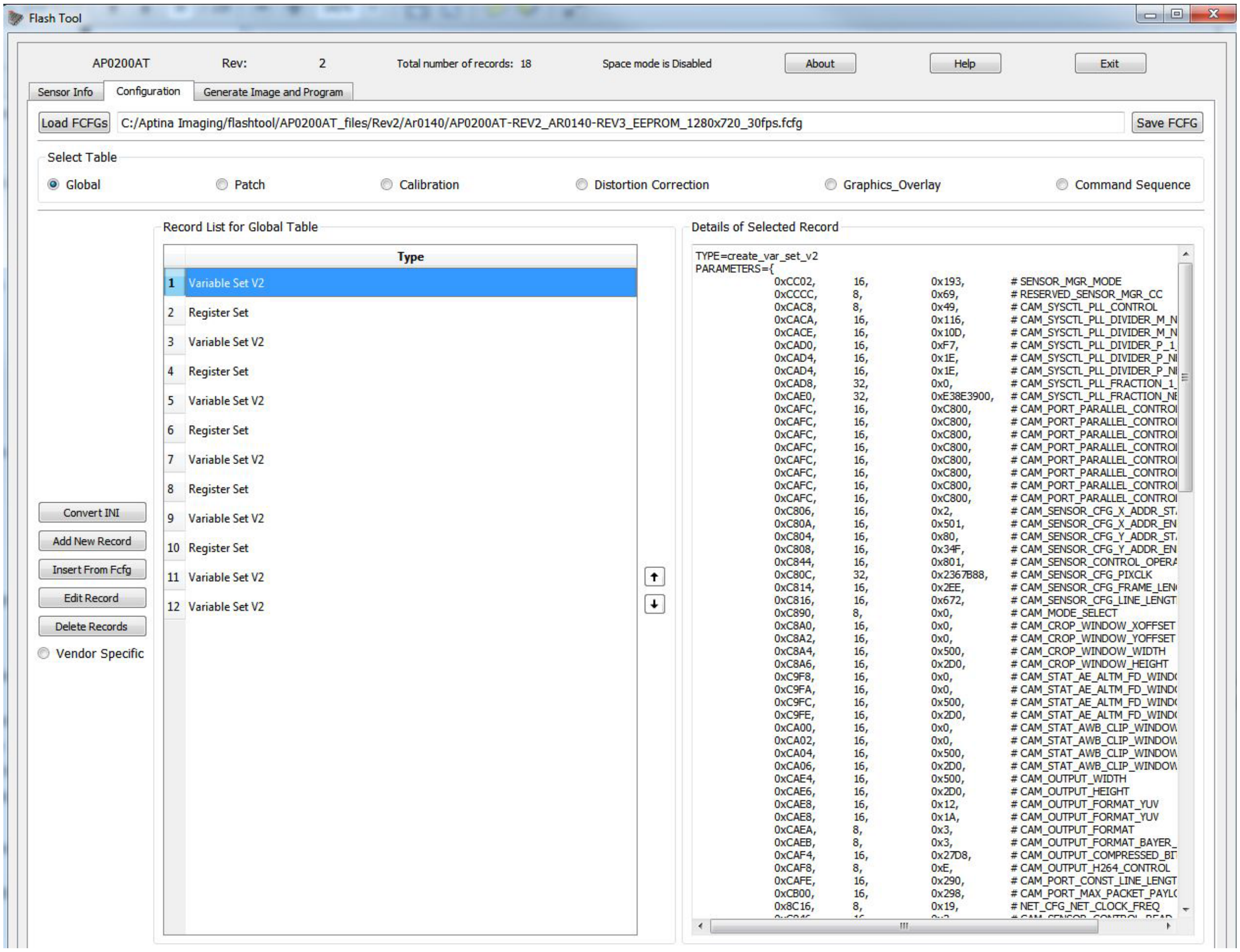
Translated Records

```
TYPE=create_reg_set
PARAMETERS={
                0x982,         0x1,           #
ACCESS_CTL_STAT
                0x98A,         0x4750,        #
PHYSICAL_ADDRESS_ACCESS
                0x98E,         0x0            #
LOGICAL_ADDRESS_ACCESS
 }


TYPE=create_var_set_v2
PARAMETERS={
                0xC830,        32,            0x24A5,
# RESERVED_CAM_30
                0xC806,        16,            0x4,
# CAM_SENSOR_CFG_X_ADDR_START
                0xC80A,        16,            0x783,
# CAM_SENSOR_CFG_X_ADDR_END
                0xC804,        16,            0x40,
# CAM_SENSOR_CFG_Y_ADDR_START
                0xC808,        16,            0x477,
# CAM_SENSOR_CFG_Y_ADDR_END
                0xC814,        16,            0x5BA,
# CAM_SENSOR_CFG_FRAME_LENGTH_LINES
                0xC816,        16,            0x7D0,
# CAM_SENSOR_CFG_LINE_LENGTH_PCK
                0xC838,        32,
0x4020CB1,      # CAM_SENSOR_CONTROL_EXTERNAL_PLL
                0xC840,        16,            0x108,
# CAM_SENSOR_CONTROL_EXTERNAL_OUTPUT_CLK_DIV
                0xC80C,        32,
0x53EC600,      # CAM_SENSOR_CFG_PIXCLK
                0xC844,        16,            0x9C2,
# CAM_SENSOR_CONTROL_OPERATION_MODE
```

NOTES:

1. The import dialog allows the user to create flash records out of DevWare presets. Importing can be done either from an INI file or by directly typing or pasting text into the dialog window using the Load Text option.
2. The sensor specific settings will need to be manually added as they transfer over as command writes and are deleted.

7. Updated settings:



AP0200AT   Rev: 2   Total number of records: 18   Space mode is Disabled

C:/Aptina Imaging/flashtool/AP0200AT_files/Rev2/Ar0140/AP0200AT-REV2_AR0140-REV3_EEPROM_1280x720_30fps.fcfg

Select Table

● Global ○ Patch ○ Calibration ○ Distortion Correction ○ Graphics_Overlay ○ Command Sequence

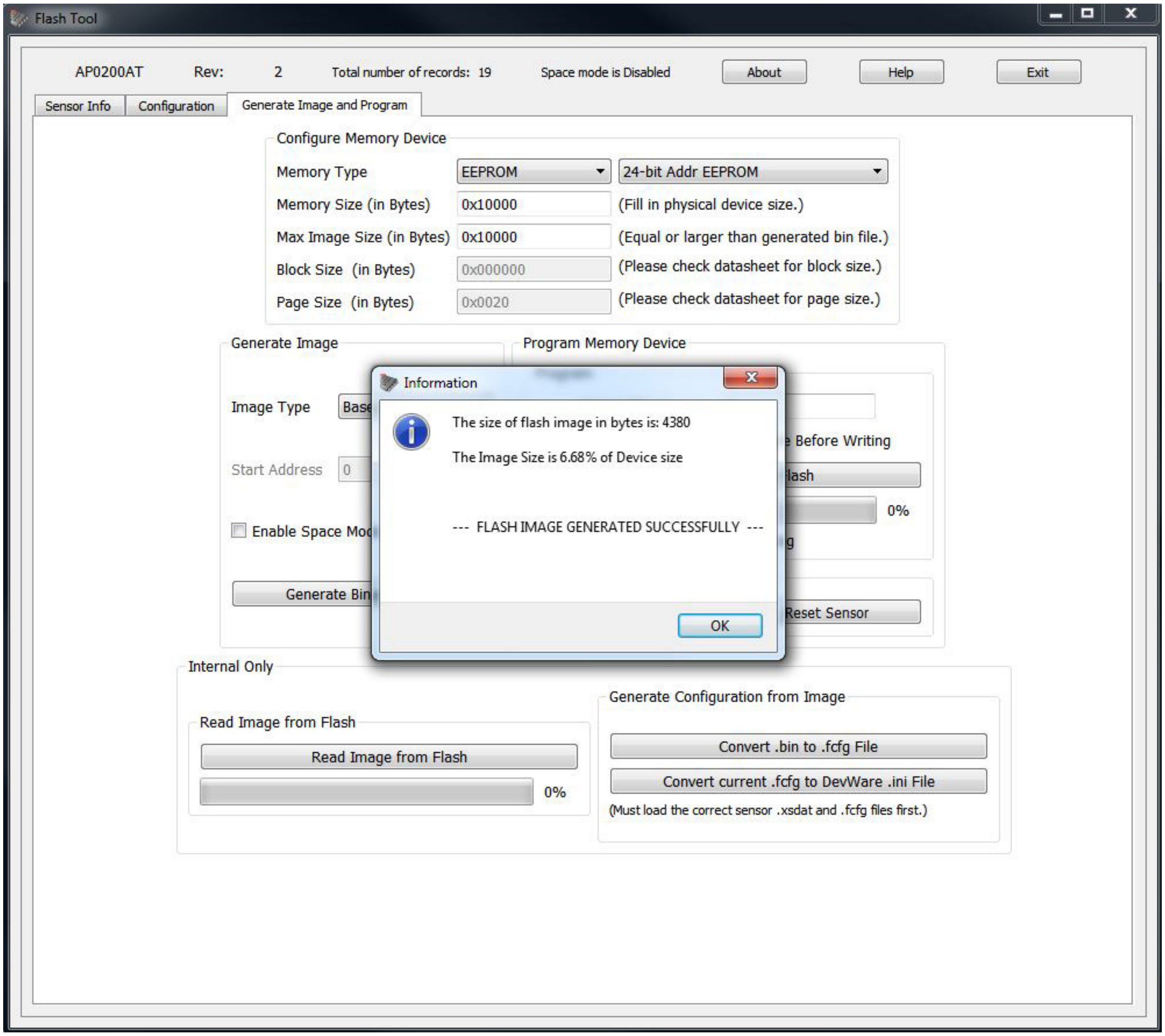
Record List for Global Table

	Type
1	Variable Set V2
2	Register Set
3	Variable Set V2
4	Register Set
5	Variable Set V2
6	Register Set
7	Variable Set V2
8	Register Set
9	Variable Set V2
10	Register Set
11	Variable Set V2
12	Variable Set V2

Details of Selected Record

```
TYPE=create_var_set_v2
PARAMETERS={
            0xCC02,    16,    0x193,        # SENSOR_MGR_MODE
            0xCCCC,    8,     0x69,         # RESERVED_SENSOR_MGR_CC
            0xCAC8,    8,     0x49,         # CAM_SYSCTL_PLL_CONTROL
            0xCACA,    16,    0x116,        # CAM_SYSCTL_PLL_DIVIDER_M_N
            0xCACE,    16,    0x10D,        # CAM_SYSCTL_PLL_DIVIDER_M_N
            0xCAD0,    16,    0xF7,         # CAM_SYSCTL_PLL_DIVIDER_P_1_
            0xCAD4,    16,    0x1E,         # CAM_SYSCTL_PLL_DIVIDER_P_NI
            0xCAD4,    16,    0x1E,         # CAM_SYSCTL_PLL_DIVIDER_P_NI
            0xCAD8,    32,    0x0,          # CAM_SYSCTL_PLL_FRACTION_1_
            0xCAE0,    32,    0xE38E3900,   # CAM_SYSCTL_PLL_FRACTION_NE
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xCAFC,    16,    0xC800,       # CAM_PORT_PARALLEL_CONTROI
            0xC806,    16,    0x2,          # CAM_SENSOR_CFG_X_ADDR_ST
            0xC80A,    16,    0x501,        # CAM_SENSOR_CFG_X_ADDR_EN
            0xC804,    16,    0x80,         # CAM_SENSOR_CFG_Y_ADDR_ST
            0xC808,    16,    0x34F,        # CAM_SENSOR_CFG_Y_ADDR_EN
            0xC844,    16,    0x801,        # CAM_SENSOR_CONTROL_OPERA
            0xC80C,    32,    0x2367B88,    # CAM_SENSOR_CFG_PIXCLK
            0xC814,    16,    0x2EE,        # CAM_SENSOR_CFG_FRAME_LEN
            0xC816,    16,    0x672,        # CAM_SENSOR_CFG_LINE_LENGT
            0xC890,    8,     0x0,          # CAM_MODE_SELECT
            0xC8A0,    16,    0x0,          # CAM_CROP_WINDOW_XOFFSET
            0xC8A2,    16,    0x0,          # CAM_CROP_WINDOW_YOFFSET
            0xC8A4,    16,    0x500,        # CAM_CROP_WINDOW_WIDTH
            0xC8A6,    16,    0x2D0,        # CAM_CROP_WINDOW_HEIGHT
            0xC9F8,    16,    0x0,          # CAM_STAT_AE_ALTM_FD_WIND
            0xC9FA,    16,    0x0,          # CAM_STAT_AE_ALTM_FD_WIND
            0xC9FC,    16,    0x500,        # CAM_STAT_AE_ALTM_FD_WIND
            0xC9FE,    16,    0x2D0,        # CAM_STAT_AE_ALTM_FD_WIND
            0xCA00,    16,    0x0,          # CAM_STAT_AWB_CLIP_WINDOW
            0xCA02,    16,    0x0,          # CAM_STAT_AWB_CLIP_WINDOW
            0xCA04,    16,    0x500,        # CAM_STAT_AWB_CLIP_WINDOW
            0xCA06,    16,    0x2D0,        # CAM_STAT_AWB_CLIP_WINDOW
            0xCAE4,    16,    0x500,        # CAM_OUTPUT_WIDTH
            0xCAE6,    16,    0x2D0,        # CAM_OUTPUT_HEIGHT
            0xCAE8,    16,    0x12,         # CAM_OUTPUT_FORMAT_YUV
            0xCAE8,    16,    0x1A,         # CAM_OUTPUT_FORMAT_YUV
            0xCAEA,    8,     0x3,          # CAM_OUTPUT_FORMAT
            0xCAEB,    8,     0x3,          # CAM_OUTPUT_FORMAT_BAYER_
            0xCAF4,    16,    0x27D8,       # CAM_OUTPUT_COMPRESSED_BI
            0xCAF8,    8,     0xE,          # CAM_OUTPUT_H264_CONTROL
            0xCAFE,    16,    0x290,        # CAM_PORT_CONST_LINE_LENGT
            0xCB00,    16,    0x298,        # CAM_PORT_MAX_PACKET_PAYL
            0x8C16,    8,     0x19,         # NET_CFG_NET_CLOCK_FREQ
```

8. Task completed:



Memory Type: EEPROM; 24-bit Addr EEPROM

Memory Size (in Bytes): 0x10000 (Fill in physical device size.)

Max Image Size (in Bytes): 0x10000 (Equal or larger than generated bin file.)

Block Size (in Bytes): 0x000000 (Please check datasheet for block size.)

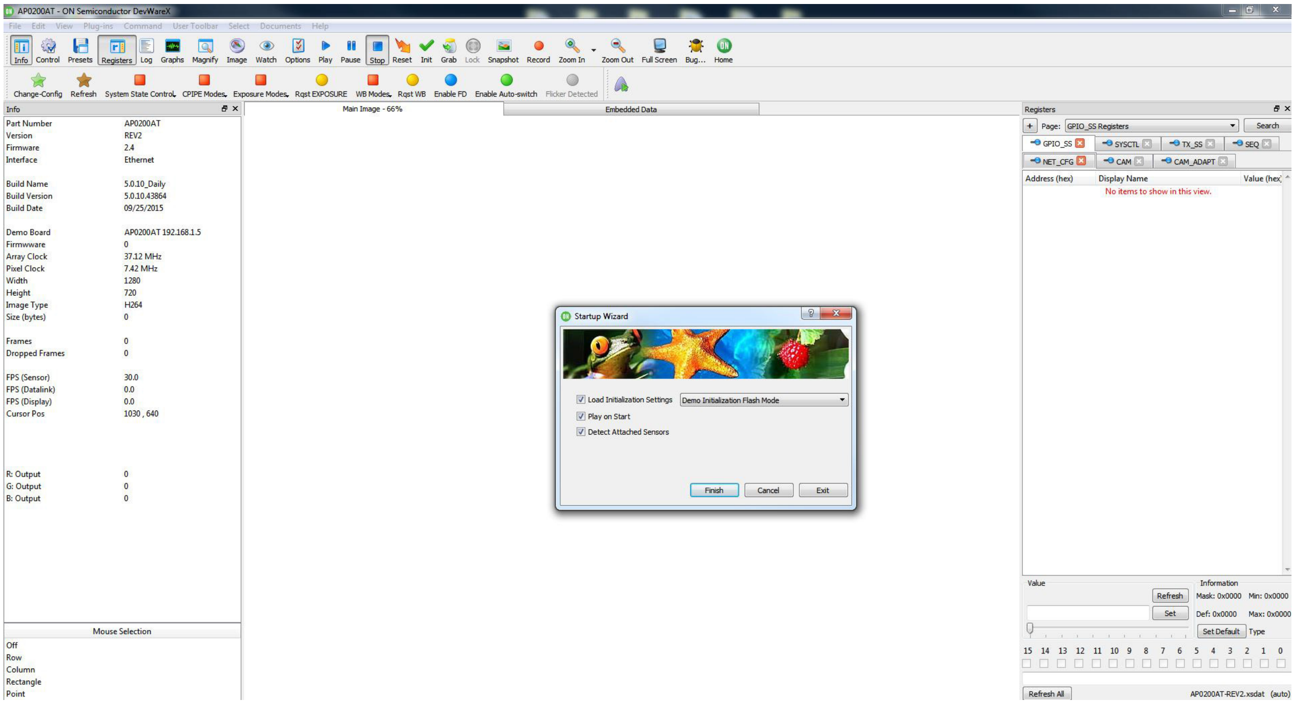
Page Size (in Bytes): 0x0020 (Please check datasheet for page size.)

The size of flash image in bytes is: 4380

The Image Size is 6.68% of Device size

--- FLASH IMAGE GENERATED SUCCESSFULLY ---

9. To load the new configuration, select *Flash−config*, but do not use *DevWare Initialization or Demo Initialization*


AP0200AT - ON Semiconductor DevWareX
File Edit View Plug-ins Command User Toolbar Select Documents Help
Info Control Presets Registers Log Graphs Magnify Image Watch Options Play Pause Stop Reset Init Grab Lock Snapshot Record Zoom In Zoom Out Full Screen Bug... Home
Change-Config Refresh System State Control CPIPE Modes Exposure Modes Rqst EXPOSURE WB Modes Rqst WB Enable FD Enable Auto-switch Flicker Detected
Main Image - 66%
Embedded Data
Registers
Search
GPIO_SS SYSCTL TX_SS SEQ NET_CFG CAM CAM_ADAPT
Refresh
Set
Set Default
Refresh All
Finish
Cancel
Exit

Info

Part Number	AP0200AT
Version	REV2
Firmware	2.4
Interface	Ethernet
Build Name	5.0.10_Daily
Build Version	5.0.10.43864
Build Date	09/25/2015
Demo Board	AP0200AT 192.168.1.5
Firmwware	0
Array Clock	37.12 MHz
Pixel Clock	7.42 MHz
Width	1280
Height	720
Image Type	H264
Size (bytes)	0
Frames	0
Dropped Frames	0
FPS (Sensor)	30.0
FPS (Datalink)	0.0
FPS (Display)	0.0
Cursor Pos	1030 , 640
R: Output	0
G: Output	0
B: Output	0

Mouse Selection

Off
Row
Column
Rectangle
Point

Startup Wizard

☑ Load Initialization Settings: Demo Initialization Flash Mode
☑ Play on Start
☑ Detect Attached Sensors

Page: GPIO_SS Registers

Address (hex)	Display Name	Value (hex)
	No items to show in this view.	

Value

Information
Mask: 0x0000 Min: 0x0000
Def: 0x0000 Max: 0x0000
Type

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

AP0200AT-REV2.xsdat (auto)

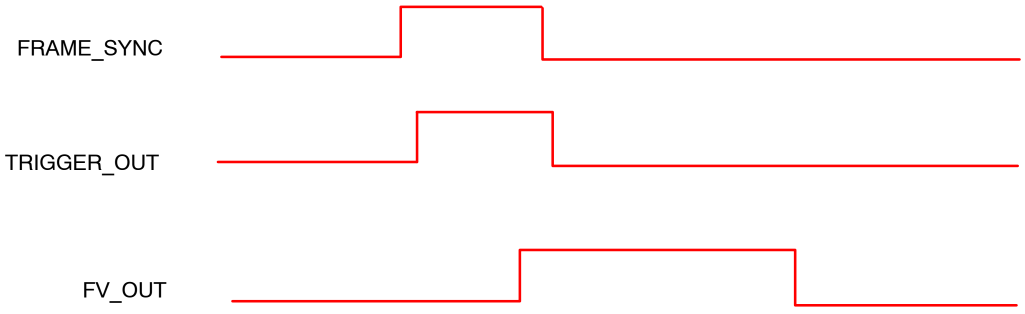
10. The part should begin imaging

MULTI−CAMERA SYNCHRONIZATION SUPPORT

The AP0102AT supports multi−camera synchronization through the FRAME_SYNC pin.

The host (or controlling entity) ‘broadcasts’ a sync−pulse to all cameras within the system that triggers capture. The AP0102AT will propagate the signal to the TRIGGER_OUT pin, and subsequently to the attached sensor’s TRIGGER pin.

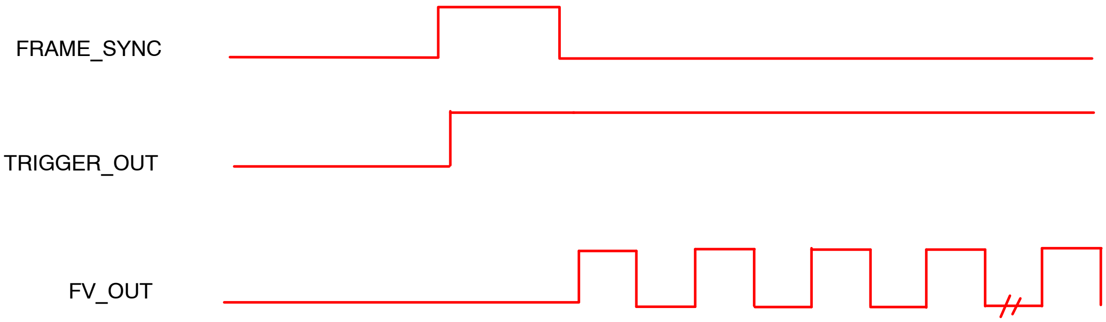
The AP0102AT supports multiple different trigger modes. The first mode supported is ‘single−shot’; this is when the trigger pulse will cause one frame to be output from the image sensor and AP0102AT (see Figure 11).



NOTE: This diagram is not to scale.

Figure 11. Single−Shot Mode

The second mode supported is called ‘continuous’, this is when a trigger pulse will cause the part to continuously output frames, see Figure 12. This mode would be especially useful for applications which have multiple sensors and need to have their video streams synchronized (for example, surround view or panoramic view applications).



NOTE: This diagram is not to scale.

Figure 12. Continuous Mode

When two or more cameras have a signal applied to the FRAME_SYNC input at the same time, the respective FV_OUT signals would be synchronized within 5 PIXCLK_OUT cycles. This assumes that all cameras have the same configuration settings and that the exposure time is the same.

Table 10. TRIGGER MODE VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xC890	[7:0]	Cam_mode_select	Selects camera operating mode: 0: Normal 1: Lens Calibration 2: Test Pattern Generator 3: Synchronized 4: Raw 5: DCNR 6: Reconstruct 7: ALTM
CamControl	R0xC891	[7:0]	Cam_Mode_Sync_Type	Selects type of synchronization: 0: Trigger (standard) 1: Trigger (deterministic) 2: Slave (standard) 3: Slave (shutter−sync)
CamControl	R0xC892	[7:0]	Cam_Mode_Sync_Trigger_mode	Selects trigger mode: 0: One−shot 1: Continuous

3. Cam_Mode_Sync_Type options 1, 2 and 3 are supported by AR0140, AR0230 and AR0231.

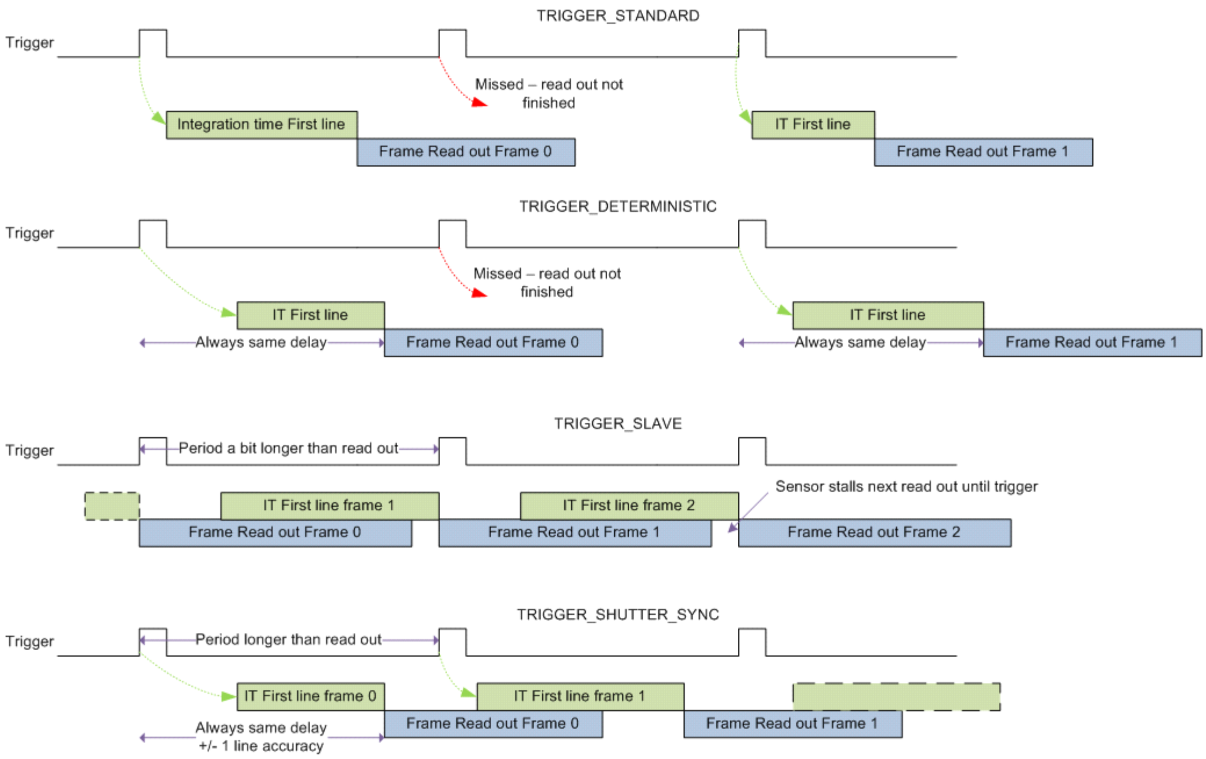


Figure 13. Synchronization Modes

Cam_Mode_Sync_Type selects the various synchronization modes.

- Standard Trigger: The standard ‘trigger’ mode is intended for single−frame read−out. The sensor starts integration when the TRIGGER pin is asserted and continues streaming. It automatically stops streaming if the TRIGGER pin is de−asserted at the time read−out is complete
- Deterministic trigger: Same as the standard trigger mode, except the sensor always delays for frame_length_lines after trigger assertion to ensure the frame rate will be consistent regardless of integration time
- Slave Mode: The slave mode is intended for video applications as the TRIGGER pin will control the FRAME_VALID output. The read out will only start when the TRIGGER pin is asserted. Any rows that start integrating before the sensor start of frame will receive a longer integration time than those after (causing artifacts) unless the TRIGGER occurs co−incident with the correct frame timing
- Shutter Sync Slave Mode: Acts as slave mode, except will delay read out by Frame_length_lines which allows the host to vary the frame rate of the sensor without image artifacts

[Configure Single Shot Mode]

Example Presets for Trigger

FIELD_WR =
CAM_MODE_SYNC_TRIGGER_MODE, 0
FIELD_WR = CAM_STAT_MODE, 0x001F
FIELD_WR = CAM_STAT_CONTROL, 0x0001
FIELD_WR = CAM_MODE_SELECT, 0x00
LOAD = Change−Config

EXPOSURE AND WHITE BALANCE IN MULTI CAMERA SYSTEMS

The AP0102AT supports auto and manual exposure and white balance modes within synchronized multi−camera systems. The Triggered Auto Exposure and Triggered Auto White Balance modes are intended for the multi−camera use cases, where a host is controlling the exposure and white balance of a number of cameras. The model is that one camera is in triggered−auto mode (the master), and the others in host−controlled mode (slaves). The master camera must calculate the exposure and gains, the host then copies this to the slaves, and all changes are then applied at the same time.

The exposure and white−balance flows are as described in Table 11 below and Table 12:

Table 11. TRIGGERED−AUTO MODE EXPOSURE FLOW

Step	Component	When	Action
1	Stats algo (master)	At EOF n	Clears ‘stale’ flags to indicate that new stats are available
2	AE algo (master)	At EOF n	Detects new stats available, uses stats to calculate new log2 exposure
3	AE algo (master)	At EOF n	Calls Sensor Manager API to distribute exposure appropriately. Sensor Manager calculates integration time and gains, updates appropriate CAM_EXP_CTRL_xxx variables
4	AE algo (master)	At EOF n	Sets ‘internal’ exposure request flag to action ‘exposure change’
5	Host	At SOF n	Reads ‘internal’ exposure request flag of master camera to determine if a new exposure is pending (it’s possible that AE is settled so no change is required). Host repeats this each SOF until ‘internal’ exposure request flag is set.
6	Host	At SOF n + m	Detects ‘internal’ exposure request flag is set on the master camera. Retrieves new CAM_EXP_CTRL_xxx exposure/gain variables
7	Host	At SOF n + m	Assuming all cameras are frame−synchronized, the host writes the CAM_EXP_CTRL_xxx variables to all slave devices (all configured in host controlled exposure mode). This change will be committed to the master camera sensor at the next SOF
8	Host	At SOF n + m	Sets ‘host’ exposure request flags on all slaves to instruct Sensor Manager to update exposure
9	Host	At SOF n + m	Sets ‘host’ exposure request flag on master to instruct Sensor Manager to update exposure
10	Sensor Manager (all)	At SOF n + m + 1	Updates sensor and color pipe with new exposure settings
12	Sensor Manager (all)	At EOF n + m + 1	Tracks when exposure change is applied. If it has, clears both ‘internal’ and ’host’ exposure request flags (via CAM Control)
13	AE algo (master)	At EOF n + m + 1	Detects ‘internal’ exposure request flag is clear, can now run again. Repeat from 1

Table 12. TRIGGERED−AUTO MODE WHITE BALANCE FLOW

Step	Component	When	Action
1	Stats algo (master)	At EOF n	Clears ‘stale’ flags to indicate that new stats are available
2	AWB algo (master)	At EOF n	Detects new stats available, uses stats to calculate color temperature and red/blue gains, updates appropriate CAM_EXP_CTRL_xxx variables
3	AWB algo (master)	At EOF n	Sets ’internal’ white−balance request flag to action ‘white−balance’ change
4	Host	At SOF n	Reads ‘internal’ white−balance request flag of master camera to determine if new gains are pending (it’s possible that AWB is settled so no change is required). Host repeats this each SOF until ‘internal’ white−balance request flag is set
5	Host	At SOF n + m	Detects ‘internal’ white−balance request flag is set on the master camera. Retrieves new CAM_EXP_CTRL_xxx gain variables
6	Host	At SOF n + m	Assuming all cameras are frame−synchronized, the host writes the CAM_EXP_CTRL_xxx variables to all slave devices (all configured in host controlled white balance mode). This change will be committed to the sensor at the next SOF
7	Host	At SOF n + m	Sets ‘host’ white−balance request flags on all slaves to instruct Sensor Manager to update white balance gains
8	Host	At SOF n + m	Sets ‘host’ white−balance request flag on master to instruct Sensor Manager to update white balance gains
9	Sensor Manager (master)	At SOF n + m + 1	Calls CAM Control which detects ‘white balance request’ is pending. Sensor Manager obtains CCIM lock, commences CCI transactions to update sensor
10	Sensor Manager (slaves)	At SOF n + m + 1	Calls CAM Control which detects ‘white balance request’ is pending. Sensor Manager obtains CCIM lock, commences CCI transactions to update sensor
11	Sensor Manager (all)	At EOF n + m + 1	Tracks when white−balance change is applied. If it has, clears both ‘internal’ and ‘host’ white−balance request flags (via CAM Control)
12	AE algo (master)	At EOF n + m + 1	Detects ‘internal’ white−balance request flag is clear, can now run again. Repeat from 1

There are a number of constraints when using the triggered−auto exposure/white−balance modes within a multi−camera synchronized system:

1. The CCIM lock MUST be available (otherwise the Sensor Manager will not be able to commit the exposure changes)
2. Both Exposure and White Balance MUST be in auto−triggered mode (otherwise a gain update may be in progress, requested by AWB, delaying the Sensor Manager applying the exposure change)

In triggered−auto mode, the host can operate the Stats in ‘one−shot’ mode if necessary to control the adaptation rate. In all cases, AE and AWB will only run when both new stats are available (the ‘stale’ bit is clear) AND the ‘internal’ exposure/white−balance request flags are clear.

NOTE: Changes to the exposure and white−balance modes can be made by a Refresh command. However, it is strongly recommended that a Change−Config operation is performed when switching from triggered−auto to any other mode. It is possible that an ’internal’ request may be pending which could cause an unsolicited exposure/white−balance change within the new mode. The Change−Config operation stops and restarts the sensor, and ensures that any pending exposure/white−balance changes have been acted on before the new mode is entered.

Table 13. AE AND AWB CONTROLS FOR MULTI−CAMERA MODE

Map	Address	Bits	Name	Description
CamControl	R0xC88C	[6:4]	Cam_aet_mode_exposure	Selects the exposure operation mode: 0: Auto Exposure 1: Triggered Auto Exposure 2: Manual Exposure 3: Host−Controlled
CamControl	R0xC97D	[2:0]	Cam_awb_mode_control	Selects the white−balance operation mode: 0: Auto White Balance 1: Triggered Auto White Balance 2: Manual White Balance 3: Host−Controlled

Example of Multi−Camera Sync

[Multi Camera Support]
FIELD_WR = CAM_AWB_MODE,
CAM_AWB_MODE_CONTROL, 0x00
FIELD_WR = CAM_AET_AEMODE,
CAM_AET_MODE_EXPOSURE, 0x00
[Set Exposure WB in triggered auto mode]
FIELD_WR = CAM_AWB_MODE,
CAM_AWB_MODE_CONTROL, 0x01
FIELD_WR = CAM_AET_AEMODE,
CAM_AET_MODE_EXPOSURE, 0x01

[Set Up Slave Mode]
FIELD_WR = CAM_AWB_MODE,
CAM_AWB_MODE_CONTROL, 0x03
FIELD_WR = CAM_AET_AEMODE,
CAM_AET_MODE_EXPOSURE, 0x03

Device ID

The AP0102AT provides its device ID for identifying the device after power−up by the host processor.

Fuse ID

The Fuse ID for both the AP0102AT and attached image sensor can be read from the part and provided to ON Semiconductor when requested for debugging purposes.

The Fuse ID for the AP0102AT can be read out using the DevWare interface.

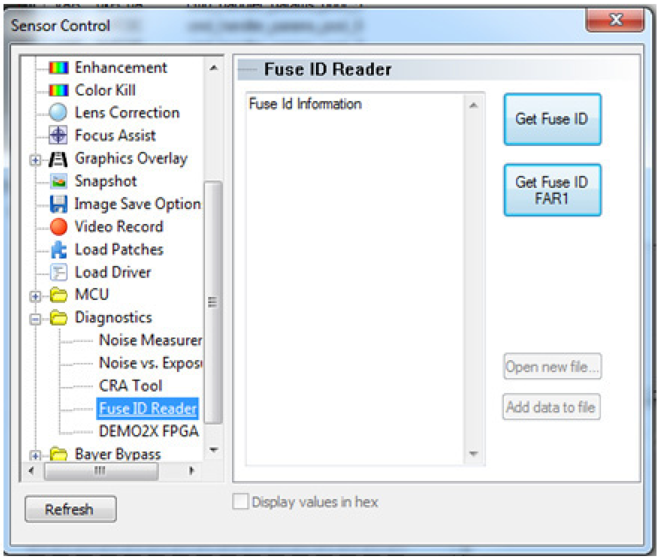


Figure 14. DevWare GUI Showing Fuse ID

CLOCK AND PLL CONTROL

Overview

The AP0102AT is the first product in the automotive SOC and companion chip product line to have two PLLs. The two PLLs are divided in functional area, but can be reconfigured to cross these boundaries. Generally, PLL0 is what is normally thought of as the PLL in single PLL products. PLL0 generates the clocks for the MCU, RX_SS, color pipe, STE, overlay, and TX_SS blocks. Optionally, this can also provide the clock for the external sensor. These portions of the AP0102 are referred to as the ISP (Image Signal Processor) portion of the clock tree. The other PLL is referred to as “PLL1”. PLL1 can be configured to supply the clock to the TX_SS block. The STE plug−in or Register Wizard (must be used to calculate the correct PLL values.

Refer to the register reference for details on the PLL controls.

PLL Bypass Mode

The AP0102AT PLL can be set up to be bypassed and use the external EXTCLK to clock the system directly during initialization. This takes effect from soft reset and can be configured according to the following example:

1. Set SYSCTL 0x0020[4] to 0x1(MCU_BOOT_PLL_BYPASS)
2. Set SYSCTL 0x001A[0] to 0x1 (RESET_SOFT)
3. Set SYSCTL 0x001A[0] to 0x0

NOTES:

1. MCU_BOOT_PLL_BYPASS should not be set if the system is configured for Auto−Config mode
2. If MCU_BOOT_PLL_BYPASS is set, the time to complete the Initialization and System Configuration phases will be extended, dependent upon the supplied external clock speed

System Health

The AP0102AT provides a watchdog monitor as means to support the host in monitoring the operational status of the system for use with ASIL compliant camera systems.

The Watchdog Monitor has a number of variables, including the MON_HEARTBEAT, MON_WATCHDOG_COUNT and MON_WATCHDOG_STATUS variables to allow the host to monitor the health of the system. The host can read these at regular intervals. The MON_HEARTBEAT and MON_WATCHDOG_COUNT will increment at defined intervals, while the MONWATCHDOG_STATUS signals a failure condition if the result is non−zero.

The MON_HEARTBEAT increments with each image frame (so it does not increment when not streaming), and the MON_WATCHDOG_COUNT increments every 200 ms in all modes.

KEEPSYNC AND CROSSBAR

Keepsync

AP0102AT has a keepsync function that will allow the user to generate consistent frame timing required to directly drive an external display.

The Keepsync block provides the following functions.

1. Generate Hsync, Vsync, Data_enab, FRAME_VALID, and LINE_VALID timings that are consistent on every frame
2. During Image pipe reconfigurations, the Keepsync block will maintain the output timing for the display, and "Black" data will be output. Once the configuration change is complete, video streaming will continue without loss of output synchronization

The number of black frames output during configuration changes will typically be 4. Depending on the exact configuration, this number can vary between 3 and 6 frames.

The Keepsync block has a "Virtual" display screen that must be programmed to be greater than or equal to the active video that is to be displayed. Outside of the "Active Video" portion of the display, the output data will be "Black". The pixel value to be used for "Black" is programmable by the host using tx_black_code_msw and tx_black_code_lsw registers.

The size of the virtual window is defined by registers tx_ks_line_length_pck and tx_ks_frame_length_lines. In 2 clocks per pixel output modes, tx_ks_frame_line_length_pck must be set to 2x (or more) the actual number of pixels in the line.

The host control interface is through CAM_PORT_KEEPSYNC and related Variables, including the enable.

The keepsync registers are located on the TX_SS page. Please refer to the AP0102AT Register Reference AND9226/D for details.

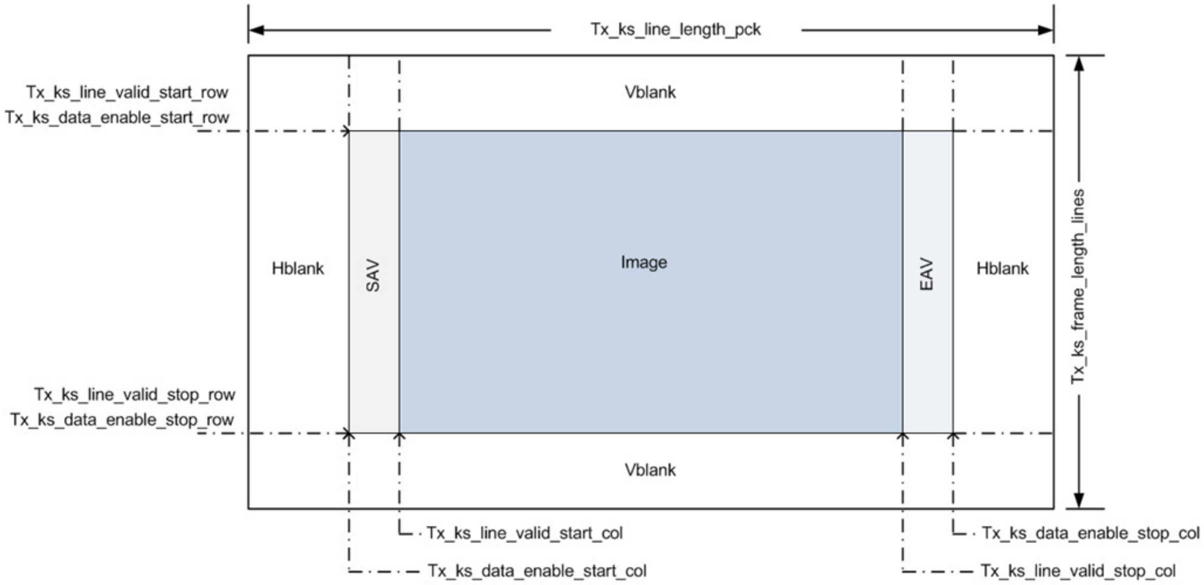

Tx_ks_line_length_pck
Tx_ks_line_valid_start_row
Tx_ks_data_enable_start_row
Vblank
Hblank
SAV
Image
EAV
Hblank
Tx_ks_frame_length_lines
Tx_ks_line_valid_stop_row
Tx_ks_data_enable_stop_row
Vblank
Tx_ks_line_valid_start_col
Tx_ks_data_enable_start_col
Tx_ks_data_enable_stop_col
Tx_ks_line_valid_stop_col

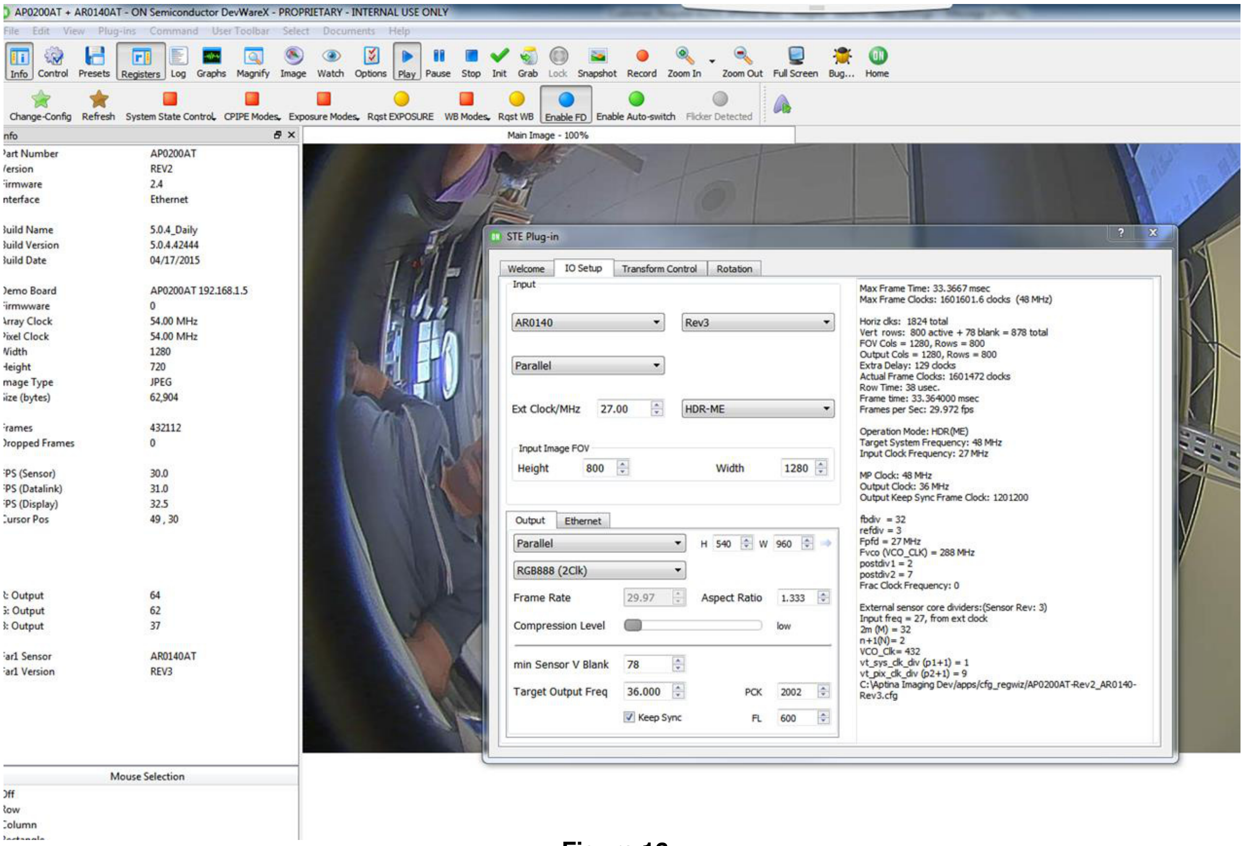
Figure 15. Keepsync

NOTE: Keepsync Constraints

1. Must be configured for constant frame rate when keepsync is enabled (i.e. no variable frame rate and no discrete frame rates)
2. Must not be in triggered frame sync mode (cam_mode_select = 3) when keepsync is enabled because these both attempt to adjust sensor timing and use the same input frame_sync physical pin
3. The existing FV/LV/pixclk output sense CAM_PORT_PARALLEL FW variables will be used to configure the keepsync equivalent controls for FV/LV/pixclk
4. Output pixel clock gating must be disabled when keepsync is enabled

STE Tool usage with Keepsync

The keepsync feature is integrated into the STE tool to create timing and settings. User can check the ‘Keep Sync’ box to define the virtual window size.


AP0200AT + AR0140AT - ON Semiconductor DevWareX - PROPRIETARY - INTERNAL USE ONLY
File Edit View Plug-ins Command User Toolbar Select Documents Help
Info Control Presets Registers Log Graphs Magnify Image Watch Options Play Pause Stop Init Grab Lock Snapshot Record Zoom In Zoom Out Full Screen Bug... Home
Change-Config Refresh System State Control CPIPE Modes Exposure Modes Rqst EXPOSURE WB Modes Rqst WB Enable FD Enable Auto-switch Flicker Detected
Main Image - 100%
STE Plug-in
Welcome IO Setup Transform Control Rotation
Mouse Selection

nfo	
art Number	AP0200AT
ersion	REV2
irmware	2.4
nterface	Ethernet
uild Name	5.0.4_Daily
uild Version	5.0.4.42444
uild Date	04/17/2015
emo Board	AP0200AT 192.168.1.5
irmwware	0
rray Clock	54.00 MHz
ixel Clock	54.00 MHz
Vidth	1280
leight	720
mage Type	JPEG
ize (bytes)	62,904
rames	432112
ropped Frames	0
PS (Sensor)	30.0
PS (Datalink)	31.0
PS (Display)	32.5
ursor Pos	49 , 30
: Output	64
: Output	62
: Output	37
ar1 Sensor	AR0140AT
ar1 Version	REV3

Mouse Selection

ff
ow
olumn

Input

AR0140 | Rev3

Parallel

Ext Clock/MHz 27.00 | HDR-ME

Input Image FOV

Height 800 | Width 1280

Output | Ethernet

Parallel | H 540 W 960

RGB888 (2Clk)

Frame Rate 29.97 | Aspect Ratio 1.333

Compression Level | low

min Sensor V Blank 78

Target Output Freq 36.000 | PCK 2002

☑ Keep Sync | FL 600

```
Max Frame Time: 33.3667 msec
Max Frame Clocks: 1601601.6 clocks (48 MHz)

Horiz clks:  1824 total
Vert  rows:  800 active + 78 blank = 878 total
FOV Cols = 1280, Rows = 800
Output Cols = 1280, Rows = 800
Extra Delay: 129 clocks
Actual Frame Clocks: 1601472 clocks
Row Time: 38 usec.
Frame time: 33.364000 msec
Frames per Sec: 29.972 fps

Operation Mode: HDR(ME)
Target System Frequency: 48 MHz
Input Clock Frequency: 27 MHz

MP Clock: 48 MHz
Output Clock: 36 MHz
Output Keep Sync Frame Clock: 1201200

fbdiv  = 32
refdiv = 3
Fpfd = 27 MHz
Fvco (VCO_CLK) = 288 MHz
postdiv1 = 2
postdiv2 = 7
Frac Clock Frequency: 0

External sensor core dividers:(Sensor Rev: 3)
Input freq = 27, from ext clock
2m (M) = 32
n+1(N)= 2
VCO_Clk= 432
vt_sys_clk_div (p1+1) = 1
vt_pix_clk_div (p2+1) = 9
C:\Aptina Imaging Dev/apps/cfg_regwiz/AP0200AT-Rev2_AR0140-
Rev3.cfg
```

Figure 16.

Then within the transform control window, one must specify the position of the video window and the synchronization signals by using the TX_SS Hardware Registers. These are accessed by using the ‘Overrides’ button in the tool.

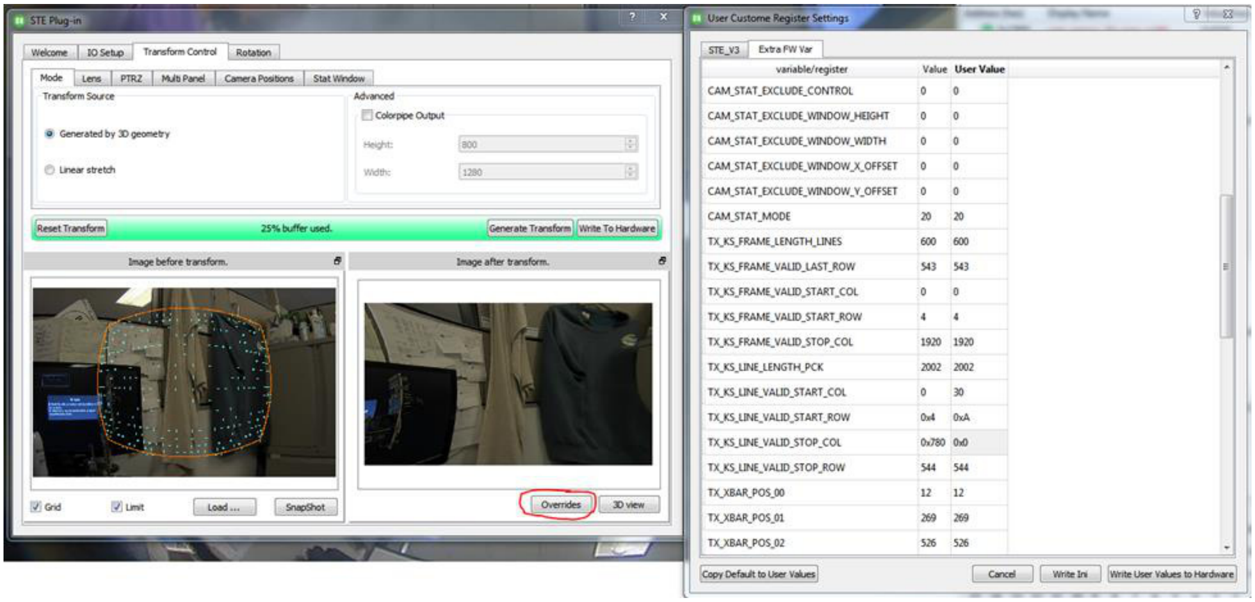
The default numbers in the ‘Value’ fields need to be updated with the customer specific requirements. Clicking on a box will toggle between hex and decimal. When the proper configuration is created, you can test it.

Crossbar

The 'Crossbar' Functionality for AP0102AT allows assigning any data, Vsync, Hsync or line_valid/frame_valid signal to any of the 27 possible parallel output pins. These output pins can be considered as DOUT[26:0] with no special meaning. Crossbar allows any data bit or control signal to be assigned to any output, or even multiple outputs at the same time.

The data sheet and AP0102A show the output pins set to default as 'Legacy Mode' where the pins are assigned per the package pinout table. However, using Crossbar, this can be customized.

Please see the description for the crossbar registers in the register reference document.



Transform Source

● Generated by 3D geometry

○ Linear stretch

Advanced

□ Colorpipe Output

Height: 800

Width: 1280

25% buffer used.

☑ Grid ☑ Limit

variable/register	Value	User Value
CAM_STAT_EXCLUDE_CONTROL	0	0
CAM_STAT_EXCLUDE_WINDOW_HEIGHT	0	0
CAM_STAT_EXCLUDE_WINDOW_WIDTH	0	0
CAM_STAT_EXCLUDE_WINDOW_X_OFFSET	0	0
CAM_STAT_EXCLUDE_WINDOW_Y_OFFSET	0	0
CAM_STAT_MODE	20	20
TX_KS_FRAME_LENGTH_LINES	600	600
TX_KS_FRAME_VALID_LAST_ROW	543	543
TX_KS_FRAME_VALID_START_COL	0	0
TX_KS_FRAME_VALID_START_ROW	4	4
TX_KS_FRAME_VALID_STOP_COL	1920	1920
TX_KS_LINE_LENGTH_PCK	2002	2002
TX_KS_LINE_VALID_START_COL	0	30
TX_KS_LINE_VALID_START_ROW	0x4	0xA
TX_KS_LINE_VALID_STOP_COL	0x780	0x0
TX_KS_LINE_VALID_STOP_ROW	544	544
TX_XBAR_POS_00	12	12
TX_XBAR_POS_01	269	269
TX_XBAR_POS_02	526	526

Figure 17.

IMAGE PIPE AND TUNING

Image and color processing in the AP0102AT is implemented as an image processor coded in hardware logic. During normal operation, the embedded microcontroller will automatically adjust the operating parameters. For normal operation of the AP0200AT, streams of raw image data from the attached image sensor are fed into the color pipeline. The user also has the option to select a number of test patterns to be input instead of sensor data.

This section will describe some of the functionality and tuning methodologies.

METRICS

The AP0102AT has a number of metrics, these metrics will influence the behavior of the algorithms. This gives the user flexibility to tune for their preference.

Brightness and Gain Metrics

Brightness and gain metrics need to be calculated to estimate the current brightness of the scene in order to apply some of the algorithms that help reduce artifacts in low light.

Sensor Gain Metric

Sensor red, green and blue gain metrics indicate the amount of gain per color channel:

$$\text{RedSensorGainMetric} = (\text{currentredanaloggain} \times \text{DCG(dualconversiongain)} \times \text{ColumnGain} \times \text{sensor\_dgain\_red}) \quad \text{(eq. 1)}$$

$$\text{BlueSensorGainMetric} = (\text{currentblueanaloggain} \times \text{DCG} \times \text{ColumnGainxsensor\_dgain\_blue}) \quad \text{(eq. 2)}$$

$$\text{GreenSensorGainMetric} = (\text{currentgreenanaloggain} \times \text{DCG} \times \text{ColumnGainxsensor\_dgain\_green}) \quad \text{(eq. 3)}$$

Gain Metric

Color channel gain metric is the total gain applied to the channel:

$$\text{RedTotalGainMetric} = (\text{currentredanaloggain} \times \text{DCG} \times \text{ColumnGain} \times \text{cpipe\_dgain\_second} \times \text{sensor\_dgain\_red} \times \text{cpipe\_dgain\_red}) \quad \text{(eq. 4)}$$

$$\text{GreenTotalGainMetric} = (\text{currentgreenanaloggain} \times \text{DCG} \times \text{ColumnGain} \times \text{cpipe\_dgain\_second} \times \text{sensor\_dgain\_green} \times \text{cpipe\_dgain\_green}) \quad \text{(eq. 5)}$$

$$\text{BlueTotalGainMetric} = (\text{currentblueanaloggainxDCGxColumnGainxcpipe\_dgain\_secondxsensor\_dgain\_bluexcpipe\_dgain\_blue}) \quad \text{(eq. 6)}$$

Brightness Metric

The Brightness Metric is the total gain applied combined with the integration time and the average luma of the current scene:

$$\text{brightness metric} = \log2\left(\frac{\text{Current Average Luma}}{\text{gainmetric} \times \text{Integration Time (in units of 10 ms)}}\right) + \text{cam\_ll\_bm\_offset} \quad \text{(eq. 7)}$$

Signal to Noise Ratio (SNR) Metric

SNR metric indicates a ratio of signal to noise. The metric is used to control color content of the image:

$$\text{SNR\_metric} = \log(\text{Average luma} / \text{green total gain metric}) \quad \text{(eq. 8)}$$

Table 14. METRIC VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xCA12	[15:0]	CAM_LL _SENSOR_RED_GAIN_METRIC	Gain metric for the sensor's red pixels. This is the product of all analog and digital gains applied to the red pixels within the external sensor. This value is unsigned fixed−point with 5 fractional bits
CamControl	R0xCA14	[15:0]	CAM_LL _SENSOR_GREEN_GAIN_METRIC	Gain metric for the sensor's green pixels. This is the product of all analog and digital gains applied to the green pixels within the external sensor. This value is unsigned fixed−point with 5 fractional bits.
CamControl	R0xCA16	[15:0]	CAM_LL _SENSOR_BLUE_GAIN_METRIC	Gain metric for the sensor's blue pixels. This is the product of all analog and digital gains applied to the blue pixels within the external sensor. This value is unsigned fixed−point with 5 fractional bits
CamControl	R0xCA18	[15:0]	CAM_LL_RED_GAIN_METRIC	This is the red channel total gain metric. It is the product of all analog and digital gains applied to the red pixels. This value is unsigned fixed−point with 5 fractional bits
CamControl	R0xCA1A	[15:0]	CAM_LL_GREEN_GAIN_METRIC	This is the blue channel total gain metric. It is the product of all analog and digital gains applied to the blue pixels. This value is unsigned fixed−point with 5 fractional bits
CamControl	R0xCA1C	[15:0]	CAM_LL_BLUE_GAIN_METRIC	This is the green channel total gain metric. It is the product of all analog and digital gains applied to the green pixels. This value is unsigned fixed−point with 5 fractional bits
CamControl	R0xCA1E	[15:0]	CAM_LL_SNR_METRIC	Signal to noise ratio metric. This is a metric used when interpolating the adaptive noise reduction strength. It is the average of the log of the image luma divided by the gain metric. This value is signed 2's complement fixed−point with 8 fractional bits
CamControl	R0xCA0A	[15:0]	CAM_LL_BRIGHTNESS_METRIC	Brightness Metric in log2 space (higher = brighter)

STATISTICS

The AP0102AT allows the host to access a range of statistics, for example, to support multi−camera use cases. The stats are collected by the Stats algorithm during active frame time or at end of frame (EOF). The AP0102AT will support passing the embedded data from the attached sensor.

The following are the accumulated statistics that are available:

- Average Luma (linear and log2)
- 5 x 5 zone average luma (linear and log2)
- 244−bin histogram
  (Note: one shot mode should be used to reliably read out histogram data)
- ALTM stats
- AWB pixels (number of pixels used for stats)
- AWB red/green/blue weighted sums (log2)
- Number of lowlights (count of pixels below threshold)

These are listed on the stats page of the register reference with the naming “stats_*”.

Statistics Acquisition

The AP0102AT supports control of the statistics acquisition in multiple modes:

- Continuous, where statistics are acquired for every frame. The statistics data is calculated during vertical blanking, and are valid after the next start of active frame. However, this does not apply to the histogram data, which is updated during the active frame and valid during vertical blanking
- One Shot Mode, where statistics are acquired for a single triggered frame
- Sensor Embedded data

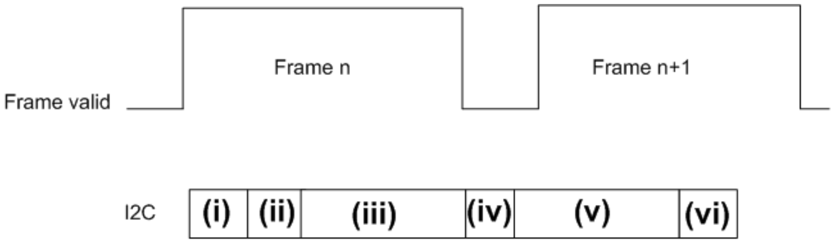
One Shot Mode

In continuous mode, there is a limit to the amount of statistics that can be retrieved by the host within the active frame time. This time is limited by the programmed blanking time and speed the host is able to retrieve the data through the serial interface. The histogram alone has 244 variables and can not be read during the allotted time.

The ‘one−shot’ mode is supported to allow the host to trigger acquisition, wait for the statistics to be acquired, then retrieve the statistics in its own time. This will have the effect of ‘freezing’ the statistics and allowing the host to read out the histogram over a longer period of time by reading out the relevant variables. In one−shot mode, the AE, AWB, Black Level and Flicker Detection functions will only run when new statistics are available. The host can switch between continuous and one−shot modes on a per−frame basis. The statistics will typically be acquired continuously, but the host can switch to one−shot mode, retrieve the statistics data, then return to continuous mode. The device will automatically postpone processing during this time.

Table 15. ONE SHOT CONTROL VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xC9E8	[0]	Cam_Stat_mode_one_shot	0: Continuous, statistics are acquired every frame 1: One−shot, statistics are only acquired after being triggered
CamControl	R0xC9EA	[0]	Cam_Stat_control_trigger	0: No trigger 1: Trigger This bit auto−clears after statistics have been acquired. The host should poll this bit to determine when statistics are available to be retrieved



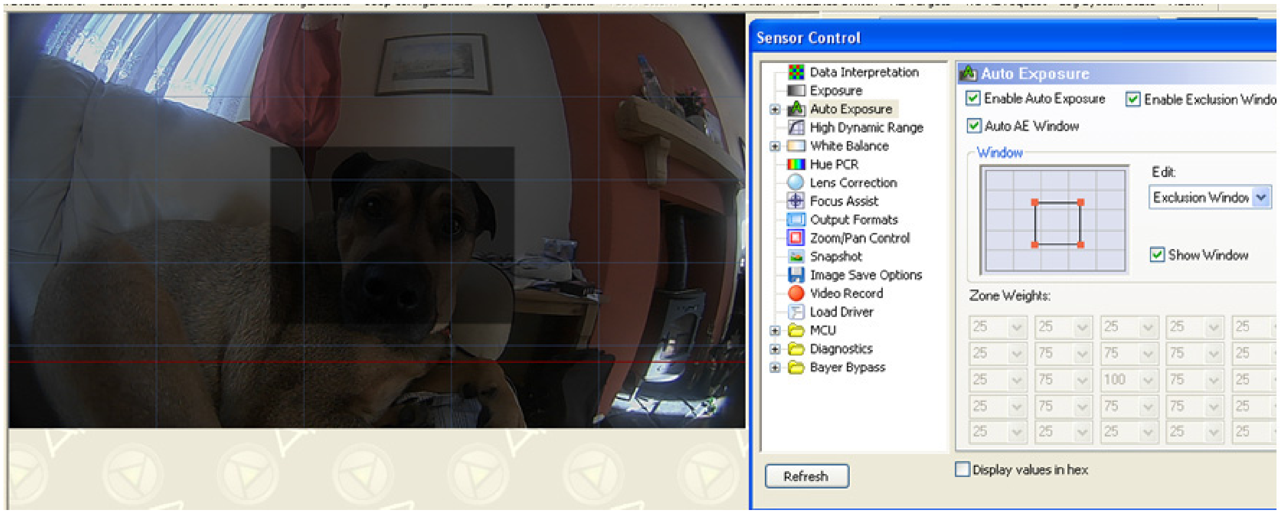
(i). Set CAM_STAT_MODE_ONE_SHOT = 1
(ii). Set CAM_STAT_CONTROL_TRIGGER = 1
(iii). POLL CAM_STAT_CONTROL_TRIGGER = 0
(iv). Read MON_HEARTBEAT (If frame number is required, an optional step)
(v). Read relevant STAT_* variables
(vi). Set CAM_STAT_MODE_ONE_SHOT = 0

Figure 18. Steps to Read Out Statistics Data Using One−Shot Mode

Stats Exclusion Window

The AP0102AT AE/AWB/ALTM statistics collection function has the capability to adjust the statistical information used in the active image area. For instance, it might be useful to exclude an object in a scene (for example, a bumper) that could be very bright and negatively affect the AE result of the overall image. The AP0102AT has the ability to set−up one exclusion window.

ON Semiconductor recommends using the DevWare auto exposure controls to enable and specify the exclusion window as shown in Figure 19.



☑ Enable Auto Exposure ☑ Enable Exclusion Windo
☑ Auto AE Window
Edit: Exclusion Windo
☑ Show Window

25	25	25	25	25
25	75	75	75	25
25	75	100	75	25
25	75	75	75	25
25	25	25	25	25

□ Display values in hex

NOTE: In this example, the dog's face is excluded from the statistics.

Figure 19. Exclusion Window Control in DevWare

Table 16. EXCLUSION WINDOW VARIABLES

Map	Address	Bits	Name	Description
Cam_stat	0xC9EC	[0]	Cam_stat_exclude_ae	Contains the AP0102AT option to exclude AE stats in specified exclusion window. Changes take effect after a Refresh command
Cam_stat	0xC9EC	[1]	Cam_stat_exclude_awb	Contains the AP0102AT option to exclude AWB stats in specified exclusion window. Changes take effect after a Refresh command
Cam_stat	0xC9EC	[2]	Cam_stat_exclude_altm	Contains the AP0102AT option to exclude ALTM stats in specified exclusion window. Changes take effect after a Refresh command
Cam_stat	0xC9F0	[15:0]	Cam_stat_exclude_window_x_offset	The horizontal offset of the first pixel to be excluded, relative to the sensor output window. Changes take effect after a Refresh command
Cam_stat	0xC9F2	[15:0]	Cam_stat_exclude_window_y_offset	The vertical offset of the first pixel to be excluded, relative to the sensor output window. Changes take effect after a Refresh command
Cam_stat	0xC9F4	[15:0]	Cam_stat_exclude_window_width	Width of the exclusion window, in pixels. Changes take effect after a Refresh command
Cam_stat	0xC9F6	[15:0]	Cam_stat_exclude_window_height	Height of the exclusion window, in pixels. Changes take effect after a Refresh command

4. The stat windows can be configured within the STE plug−in.

Embedded Data and Statistics

Some ON Semiconductor sensor's support a feature that, if enabled, inserts two extra lines at the beginning and end of each frame which contain information about that frame. The first two lines contain specific register values that were used to capture that frame. These values allow the host to know certain important things about how the sensor was configured for that frame, e.g. exposure, gain, image size, etc. The last two lines contain statistics about the image that was captured, e.g. mean values, intensity histograms, etc.

The AP0102AT will support this information being available in all modes. This feature is supported on output image sizes from full resolution to VGA.

FRAME COUNTER

The ‘mon_heartbeat’ variable in the AP0102AT will increment with every frame while the device is streaming and can be used as a frame counter. The counter will update during vertical blanking time and will continuously wrap back to zero and continue counting once the 16−bit limit is reached.

Using the frame counter with the ‘Wait for event’ (SOF – Start of Frame) can be used to synchronize when the host reads the frame counter value.

DEFECT CORRECTION

After data decompanding the image stream processing starts with defect correction. To obtain defect free images, the pixels marked defective during sensor readout and the pixels determined defective by the defect correction algorithms are replaced with values derived from the non−defective neighboring pixels. This image processing technique is called defect correction.

AdaCD (Adaptive Color Difference)

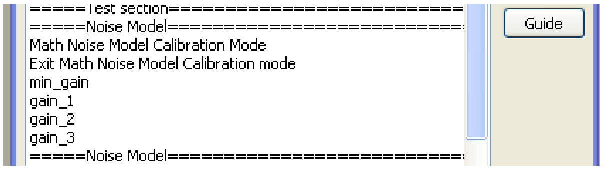
Automotive applications require good performance in extremely low light, even at high temperature conditions. In these stringent conditions the image sensor is prone to higher noise levels, and so efficient noise reduction techniques are required to circumvent this sensor limitation and deliver a high quality image to the user. The AdaCD Noise Reduction Filter is able to adapt its noise filtering process to local image structure and noise level, removing most objectionable color noise while preserving edge details.

Defect Correction and AdaCD Tuning

The settings for both of these algorithms are derived using the same methodology. The process can be broken into two stages. The first stage is to make noise measurement using DevWare and the second stage is to use Sensor Tune to produce values based on the measurements made during the first stage. The test setup for stage 1 requires the module (or demo board) to be shielded from light, the object of the test is to measure the noise when no light is falling on the sensor over different gain settings.

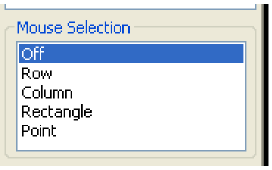
The sequence presented assumes that DevWare is being used. To perform this in a customer system, contact ON Semiconductor.

1. Get the camera imaging (run DevWare Init or boot from flash)
2. Run “Math Noise Model Calibration mode” preset. This will put the camera into the relevant Bayer mode and turn the AE off

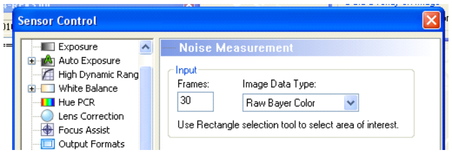


```
=====Test section==========================
=====Noise Model===========================
Math Noise Model Calibration Mode
Exit Math Noise Model Calibration mode
min_gain
gain_1
gain_2
gain_3
=====Noise Model===========================
```

3. In mouse selection select off (this will ensure the whole image is considered)



4. In sensor control select noise measurements to 30 and select Raw Bayer Color



Sensor Control

Noise Measurement

Input

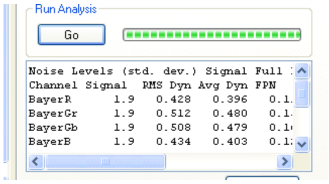
Frames: 30

Image Data Type: Raw Bayer Color

Use Rectangle selection tool to select area of interest.

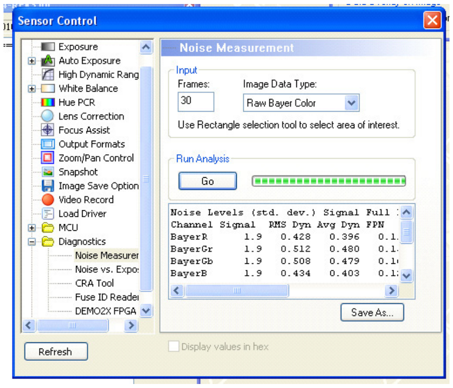
5. Run min_gain preset

6. Hit go in the noise measurement window



```
Noise Levels (std. dev.) Signal Full :
Channel Signal  RMS Dyn  Avg Dyn  FPN
BayerR      1.9    0.428    0.396    0.1.
BayerGr     1.9    0.512    0.480    0.1.
BayerGb     1.9    0.508    0.479    0.1
BayerB      1.9    0.434    0.403    0.1:
```

7. Data for the four color planes can be seen, in the RMS Dyn column note the maximum value



Frames: 30

Image Data Type: Raw Bayer Color

Use Rectangle selection tool to select area of interest.

```
Noise Levels (std. dev.) Signal Full :
Channel Signal  RMS Dyn  Avg Dyn  FPN
BayerR      1.9    0.428    0.396    0.1.
BayerGr     1.9    0.512    0.480    0.1.
BayerGb     1.9    0.508    0.479    0.1
BayerB      1.9    0.434    0.403    0.1:
```

8. Run gain_1 preset
9. Repeat step 6 and 7
10. Run gain_2 preset
11. Repeat step 6 and 7
12. Run gain_3 step preset
13. Repeat step 6 and 7

At the end of this process the user should have 4 values for noise, this will be used as inputs for Sensor Tune.

ALTM − ADAPTIVE LOCAL TONE MAPPING

Real world scenes often have very high dynamic range (HDR) that far exceeds the electrical dynamic range of the imager. Dynamic range is defined as the luminance ratio between the brightest and the darkest object in a scene. In recent years many technologies have been developed to capture the full dynamic range of real world scenes. For example, the multiple exposure method is a widely adopted method for capturing high dynamic range images, which combines a series of low dynamic range images of the same scene taken under different exposure times into a single HDR image.

Even though the new digital imaging technology enables the capture of the full dynamic range, low dynamic range display devices are the limiting factor. Today's typical LCD monitor has contrast ratio around 1,000:1; however, it is not atypical for an HDR image having contrast ratio around 250,000:1. Therefore, in order to reproduce HDR images on a low dynamic range display device, the captured high dynamic range must be compressed to the available range of the display device. This is commonly called tone mapping.

Tone mapping methods can be classified into global tone mapping and local tone mapping. Global tone mapping methods apply the same mapping function to all pixels.

While global tone mapping methods provide computationally simple and easy to use solutions, they often cause loss of contrast and detail. A local tone mapping is thus necessary in addition to global tone mapping for the reproduction of visually more appealing images that also reveal scene details that are important for automotive safety applications. Local tone mapping methods use a spatially varying mapping function determined by the neighborhood of a pixel, which allows it to increase the local contrast and the visibility of some details of the image. Local methods usually yield more pleasing results because they exploit the fact that human vision is more sensitive to local contrast.

ON Semiconductor's ALTM solution significantly improves the performance over global tone mapping. ALTM is directly applied to the Bayer domain to compress the dynamic range from 20−bit to 12−bit. This allows the regular color pipeline to be used for HDR image rendering.

The cam control variables give the user the relevant controls.

Table 17. CAM CONTROL VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xC988	[0]	CAM_ALTM_MODE_ENABLE	When enabled, the dynamic brightness control cam_altm_key_k1 is coupled to ae_rule_avg_log_y_from_stats
CamControl	R0xC988	[1]	CAM_ALTM_SHARPNESS_ENABLE	Enable interpolation of the ALTM 'Sharpening Strength' based on the cam_ll_brightness_metric
CamControl	R0xC988	[2]	CAM_ALTM_DYNAMIC_DAMPING _ENABLE	Enable dynamic damping for ALTM adaptation
CamControl	R0xC98A	[15:0]	CAM_ALTM_KEY_K0	Noise floor used to calculate the key that controls the brightness of the tone mapped image
CamControl	R0xC98C	[31:0]	CAM_ALTM_KEY_K1 (Read only)	This value divided by cam_altm_key_k0 is used to calculate the key that controls the brightness of the tone mapped image. This parameter controls the brightness and is calculated by the firmware when cam_altm_mode_enable = 0x1
Cam Control	R0xC990	[15:0]	CAM_ALTM_LO_GAMMA	Contrast control parameter for the dark regions of an image
Cam Control	R0xC992	[15:0]	CAM_ALTM_HI_GAMMA	Contrast control parameter for the bright regions of an image
Cam Control	R0xC994	[15:0]	CAM_ALTM_K1_SLOPE	K1_slope controls how the ALTM K1 parameter increases in lowlight. If the cam_altm_k1_slope is increased it will decrease the noise and detail in lowlight conditions. If cam_altm_k1_slope is decreased it will increase the noise and detail in lowlight conditions and increase the apparent brightness
Cam Control	R0xC996	[15:0]	CAM_ALTM_K1_MIN	The minimum allowable k1 value
Cam Control	R0xC998	[15:0]	CAM_ALTM_K1_MAX	The maximum allowable k1 value
Cam Control	R0xC99A	[15:0]	CAM_ALTM_DARK_BM	Programmable dark starting brightness value
Cam Control	R0xC99C	[15:0]	CAM_ALTM_BRIGHT_BM	Programmable bright ending brightness value
Cam Control	R0xC99E	[15:0]	CAM_ALTM_K1_DAMPING_SPEED	Programmable k1 damping speed parameter. A lower value means faster adaptation, a higher value means slower adaptation (unity=1)
Cam Control	R0xC9A0	[15:0]	CAM_ALTM_SHARPNESS_DARK_BM	Programmable threshold for sharpness, this is referenced to CAM_LL_BRIGHTNESS_METRIC
Cam Control	R0xC9A2	[15:0]	CAM_ALTM_SHARPNESS_BRIGHT_BM	Programmable threshold for sharpness, this is referenced to CAM_LL_BRIGHTNESS_METRIC
Cam Control	R0xC9A4	[15:0]	CAM_ALTM _SHARPNESS_STRENGTH_DARK	Dark Sharpening Strength value
Cam Control	R0xC9A6	[15:0]	CAM_ALTM _SHARPNESS_STRENGTH_BRIGHT	Bright Sharpening Strength value

ALTM Tuning

The following can be tuned using the ALTM controls:

- Contrast
- Sharpening
- Brightness

The graph below shows the effect of the ALTM variables.

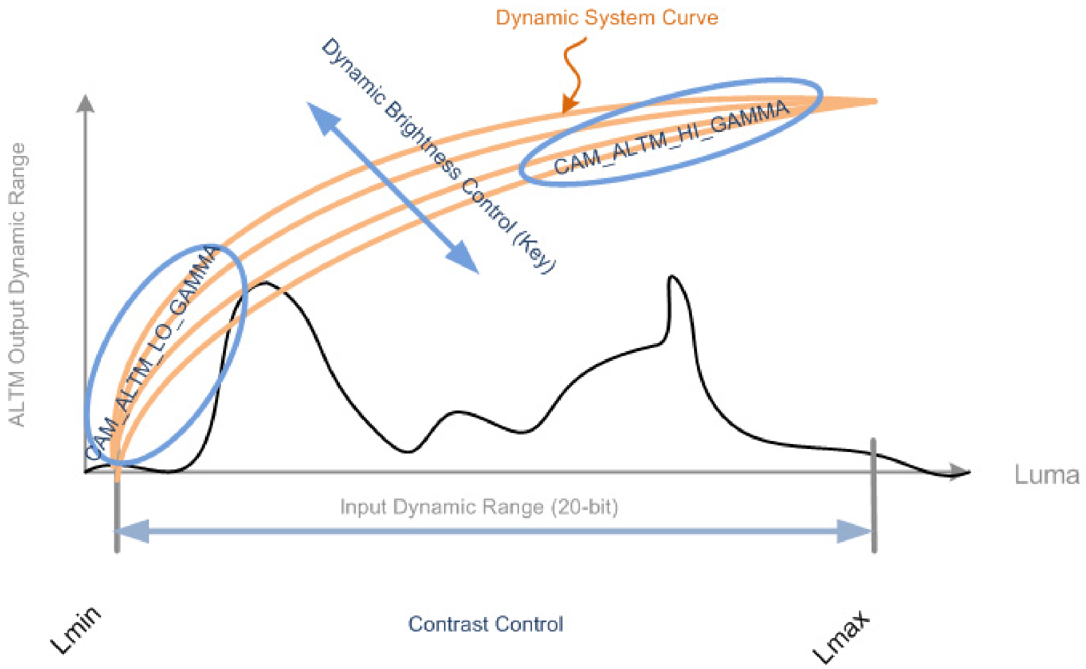

Dynamic System Curve
Dynamic Brightness Control (Key)
CAM_ALTM_HI_GAMMA
CAM_ALTM_LO_GAMMA
ALTM Output Dynamic Range
Luma
Input Dynamic Range (20-bit)
Lmin
Contrast Control
Lmax

Figure 20. Effect of ALTM Variables

ALTM Brightness

cam_altm_k1_min can be used to achieve the overall brightness required. Increasing the value of cam_altm_k1_min will make the image dimmer and decreasing cam_altm_k1_min will increase the overall brightness.

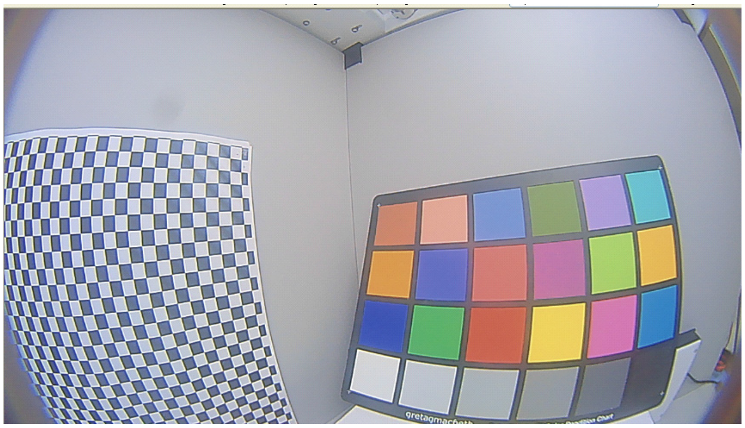


Figure 21. Cam_altm_k1_min = 128

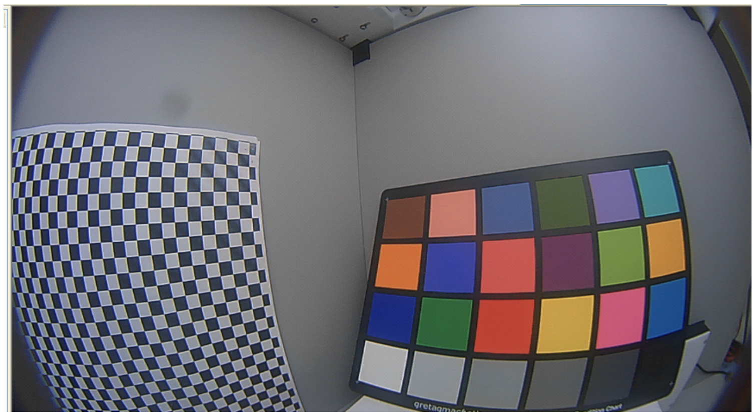


Figure 22. Cam_altm_k1_min = 2048 (default)

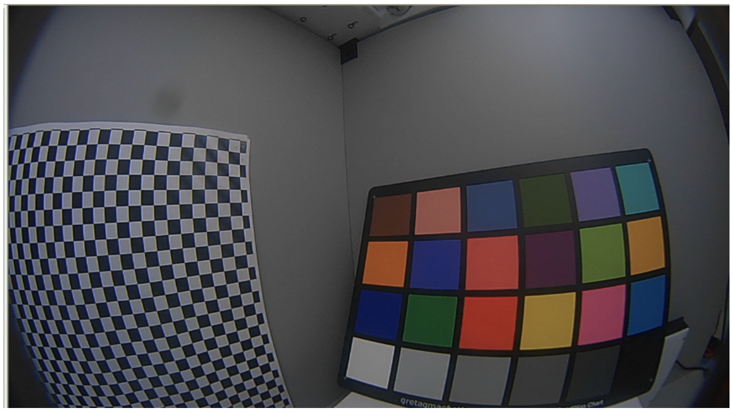


Figure 23. Cam_altm_k1_min = 65535

ALTM Sharpening

The user can vary the amount of sharpness that is applied in the ALTM block, this sharpness is localized (the AP0102AT also has a global sharpening control in the Aperture block).

- Cam_altm_sharpness_dark_bm and cam_altm_sharpness_bright_bm are the reference points on the brightness metric scale
- Cam_altm_sharpness_strength_dark and cam_altm_sharpness_strength_bright will determine how much sharpness will be applied

The graph below show the relationship of the ALTM Sharpness variables in relation to the brightness metric.

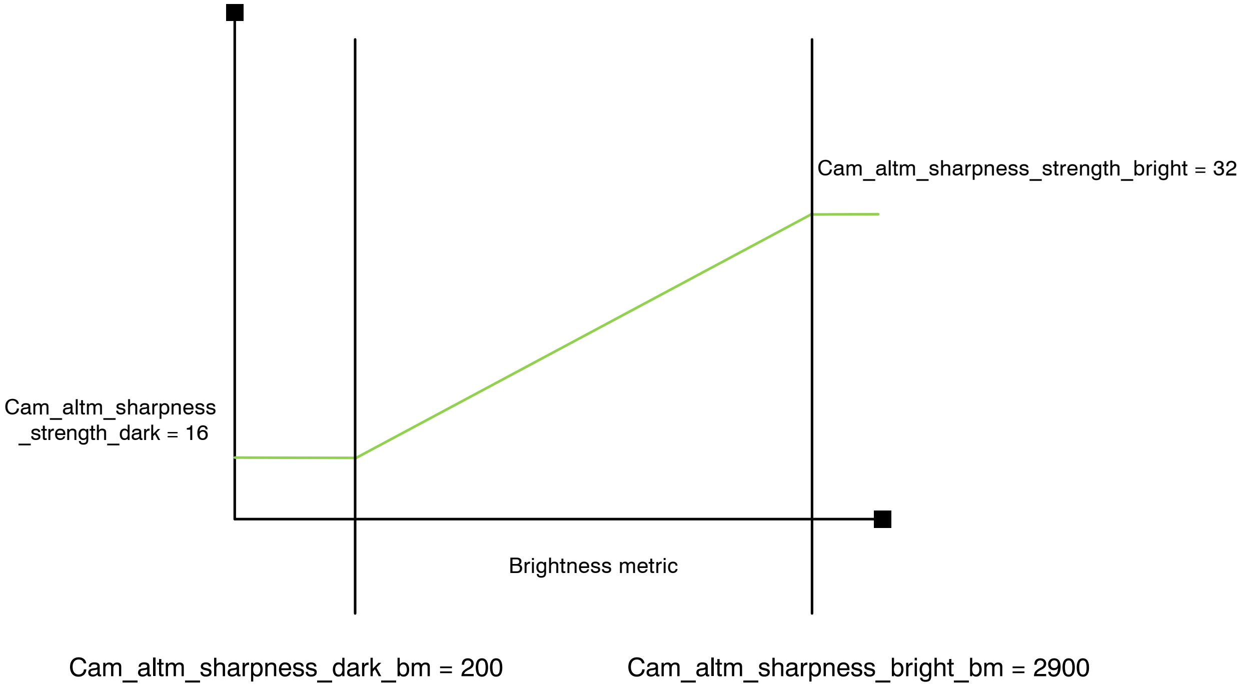


Figure 24. ALTM Localized Sharpening

In Figure 25 the image on the left has a higher value of ALTM sharpness applied.

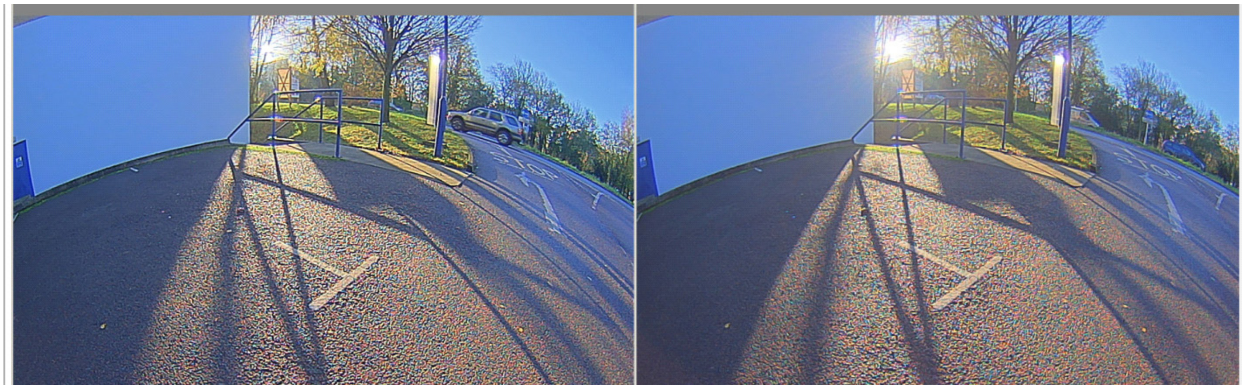
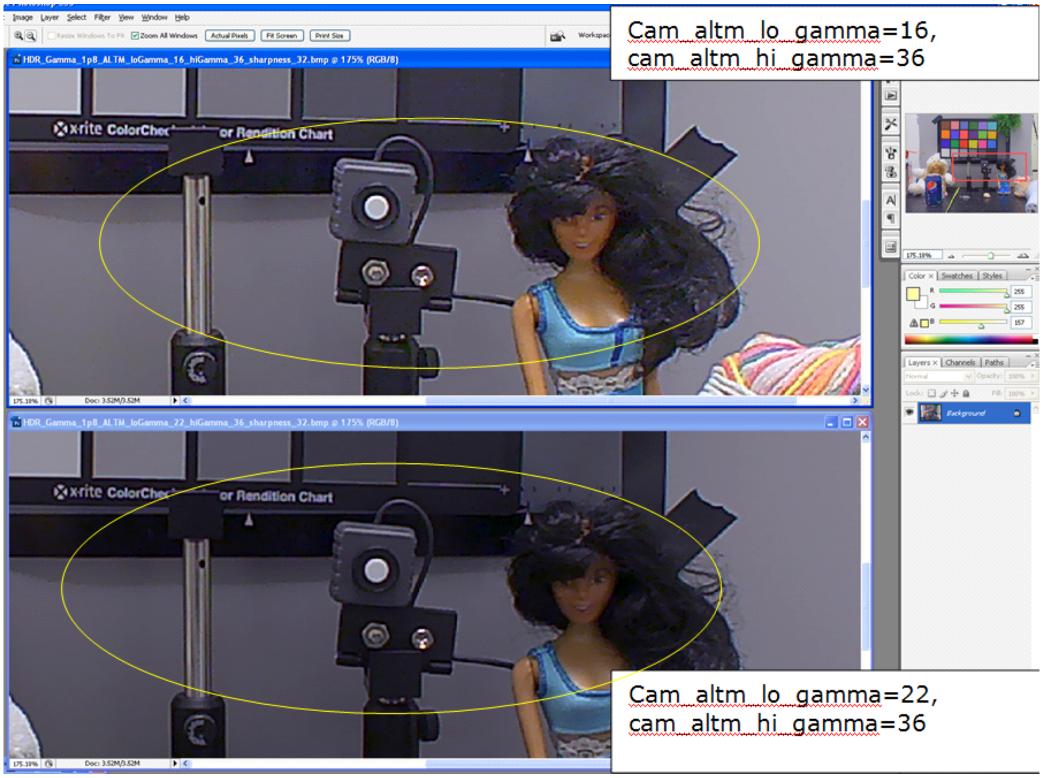


Figure 25. Effect of ALTM Sharpening

ALTM Gamma

The AP0102AT has an ALTM gamma that can be used to apply a gamma correction per pixel in the Bayer domain. Once the user tunes the overall brightness of the scene using the K1 parameter as described above, they could then choose to enhance or compress the data in the dark and bright regions. This is achieved by using the ALTM Gamma parameters.


HDR_Gamma_1p8_ALTM_loGamma_16_hiGamma_36_sharpness_32.bmp @ 175% (RGB/8)
HDR_Gamma_1p8_ALTM_loGamma_22_hiGamma_36_sharpness_32.bmp @ 175% (RGB/8)

Cam_altm_lo_gamma=16,
cam_altm_hi_gamma=36

Cam_altm_lo_gamma=22,
cam_altm_hi_gamma=36

Figure 26. Effect of Gamma Control

Note the relatively higher noise on the doll’s face.

GAMMA CORRECTION

The gamma correction curve is implemented as a piecewise linear function with 33 knee points, taking 12−bit arguments and mapping them to 10−bit output.

The device includes a block for gamma correction that has the capability to adjust its shape, based on brightness, to enhance the performance under certain lighting conditions. Two custom gamma correction tables may be uploaded, one corresponding to a contrast curve for brighter lighting conditions, the other one corresponding to a noise reduction curve for lower lighting conditions. The final gamma correction table used depends on the brightness of the scene and can take the form of either uploaded tables or an interpolated version of the two tables. A single (non−adjusting) table for all conditions can also be used.

Adaptive Gamma Curve Based on Brightness Metric

However instead of programming all the knee points for the curves a small number of variables are available to change the shape of the curve. However due to the interaction with ALTM it is recommended that CAM_LL_GAMMA is set to ~1.0 and the "gamma" would be implemented using the ALTM controls.

The slope of the curves could still be adjusted for different light levels.

Table 18. GAMMA VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xCA0A	[15:0]	CAM_LL_BRIGHTNESS_METRIC	Brightness Metric in log2 space (higher = brighter)
CamControl	R0xCA30	[15:0]	CAM_LL_CONTRAST_BRIGHT_BM	Interpolation start point for first curve
CamControl	R0xCA32	[15:0]	CAM_LL_CONTRAST_DARK_BM	Interpolation stop point for second curve
CamControl	R0xCA34	[15:0]	CAM_LL_GAMMA	The value of the gamma curve, this is applied to both curves. The default is 100; this equates to a gamma of 1.0
CamControl	R0xCA36	[7:0]	CAM_LL_CONTRAST_GRADIENT_BRIGHT	The value of the contrast gradient which would be used for the first curve
CamControl	R0xCA37	[7:0]	CAM_LL_CONTRAST_GRADIENT_DARK	The value of the contrast gradient which would be used for the second curve
CamControl	R0xCA38	[7:0]	CAM_LL_CONTRAST _INTERCEPT_POINT_BRIGHT	Pixel value for the inflection point in the contrast curve in bright conditions
CamControl	R0xCA39	[7:0]	CAM_LL_CONTRAST _INTERCEPT_POINT_DARK	Pixel value for the inflection point in the contrast curve in dark conditions

Fade−to−Black

The device allows image to fade to black under extreme low−light conditions. This feature enables users to optimize the performance of the sensor under these conditions. It minimizes the perception of noise and artifacts while the available illumination is diminishing.

This feature has two user set points that reference the brightness of the scene. When the Fade−to−Black starts, it will interpolate to the end point as the light falls until it gets to the end point. When at the end point, the image will be black.

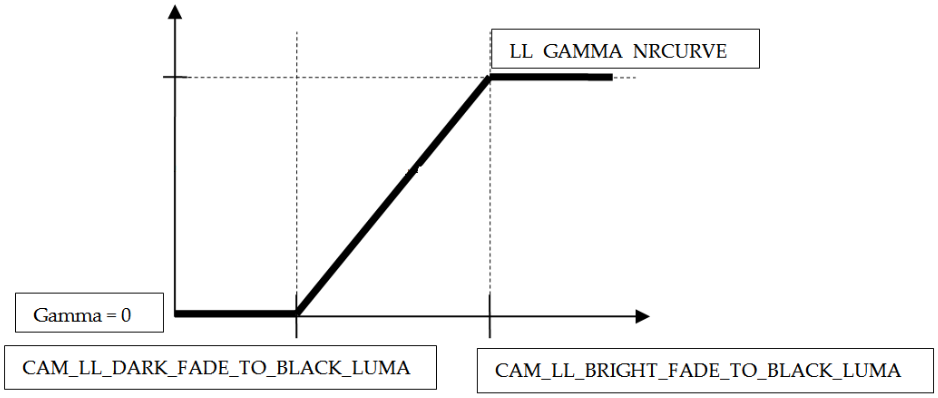


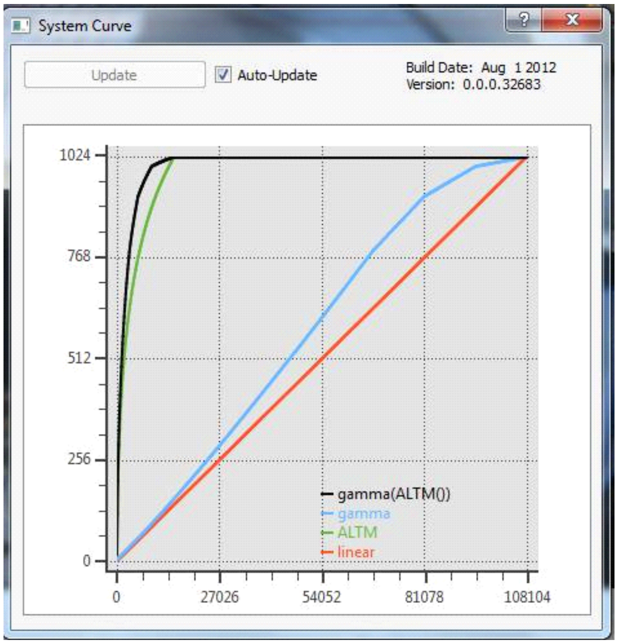
Figure 27. Fade−to−Black Adaptation

Table 19. FADE−TO−BLACK VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xCA3A	[15:0]	CAM_LL_BRIGHT_FADE_TO_BLACK_LUMA	Start value of ll_average_luma_fade_to_black for fade−to−black mode
CamControl	R0xCA3C	[15:0]	CAM_LL_DARK_FADE_TO_BLACK_LUMA	Stop value of ll_average_luma_fade_to_black for fade−to−black mode
Lowlight	R0xBC02[3]	[1]	LL_MODE	Enable Fade−to−Black mode
			LL_ENABLE_FADE_TO_BLACK	
Lowlight	R0xBC8E	[15:0]	LL_AVERAGE_LUMA_FADE_TO_BLACK	Average luma for fade to black (calculated using an adaptive weighting algorithm)

GAMMA SYSTEM CURVE PLUG−IN FOR DEVWARE

DevWare provides a plug−in that will provide real time feedback of changes to registers or variables and the effect on the overall gamma curve. The tool will assist the developer to better understand the system curve and to perform tuning and trade−off between ALTM and Gamma Correction curve to optimize overall system results. The tool is under the *Plug−ins* tab of the AP0102AT DevWare tabs.



System Curve

☑ Auto-Update

Build Date: Aug 1 2012
Version: 0.0.0.32683

Figure 28. System Curve DevWare Plug−in

DEMOSAIC

Color interpolation, or demosaic, is required to reproduce a full color image from the Bayer input image from the sensor. Digital image processing is needed such that the missing color values in the Bayer pattern image is interpolated to produce RGB values for every pixel in the array.

The demosaic algorithm will also determine if a given pixel falls on an edge. The demosaic controls are referenced to the gain metric and the demosaic edge threshold is used to decide if the current pixel is on an edge in the demosaic transform engine. When a pixel is determined to be on an edge, the Demosaic algorithm attempts to maintain sharpness and to limit aliasing.

Table 20. DEMOSAIC CONTROL VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xCA2A	[15:0]	Cam_ll_demosaic_high	Sets the high demosaic edge threshold
CamControl	R0xCA2B	[15:0]	Cam_ll_demosaic_low	Sets the low demosaic edge threshold

AUTO EXPOSURE

The auto exposure algorithm optimizes scene exposure to minimize clipping and saturation in critical areas of the image. This is achieved by controlling exposure time and the analog gains of the sensor as well as digital gains applied to the image.

The auto exposure module analyzes image statistics collected by the exposure measurement engine, makes a decision, and programs the sensor and color pipeline to achieve the desired exposure. The measurement engine subdivides the image into 25 windows organized as a 5 x 5 grid.

AE Modes

The available AE modes are:

- Indoor AE
- Discrete Frame Rate AE

These modes are independent, and can also be enabled concurrently. When all modes are disabled, the AE will simply function based on the available lighting conditions.

Table 21. VARIABLES FOR AE MODE

Map	Address	Bits	Name	Description
CamControl	0xC8BC	[0]	Cam_aet_mode_indoor	Enable 'indoor' mode: 0: Disabled 1: Enabled: limit AE to minimum 1 flicker period of exposure
CamControl	0xC8BC	[1]	Cam_aet_discrete_framerate	Controls variable frame−rate operation: 0: continuously−variable: the frame rate varies in steps of 1 flicker period. 0 = Disable feature 1: Discrete: the frame rate will vary by discrete steps. The discrete frame rates are determined by the cam_aet_frame_rate_0 through cam_aet_frame_rate_2 variables. Note this bit is only supported in SDR mode. Additional Note: HDR mode always operates as fixed frame rate
CamControl	0xC8BC	[6:4]	Cam_aet_mode_exposure	Controls the Exposure operation mode: 0: Auto Exposure 1: Triggered Auto Exposure 2: Manual Exposure 3: Host−Controlled

Indoor AE

Indoor mode has restrictions to ensure that the integration time will always remain a multiple of the flicker frequency. This ensures that the flicker will be avoided at all times. In this mode, the integration time will be limited to at least one flicker period. Due to this limitation, under bright light conditions, images can be overexposed.

When using HDR sensors, flicker in T2 or T3 cannot be avoided.

```
[Enter indoor mode]
  FIELD_WR = CAM_AET_AEMODE,
   CAM_AET_MODE_INDOOR, 0x01
  LOAD = Change−Config
```

```
[Exit indoor mode]
  FIELD_WR  = CAM_AET_AEMODE,
   CAM_AET_MODE_INDOOR, 0x00
  LOAD = Change−Config
```

AE Window Overview

Auto−Exposure operates by:

1. Analyzing image statistics (for example, average luma) collected by the device's internal exposure measurement engine,
2. Calculating the best exposure and gain settings,
3. Programming the sensor and color pipeline based on the calculated exposure and gain.

The AP0102AT subdivides the image into 25 windows organized as a 5 x 5 grid. The AP0102AT can be configured to gather data on a user defined area of the sensor. Two modes of operation are supported:

- Automatic when the AE statistics window is specified relative to the output image dimensions.
- Manual when the statistics window is configured relative to the sensor window.

In both cases the user must configure the variables to define the window. ON Semiconductor recommends that the user use DevWare or the STE GUI to configure the AE Window.

AE Track

Other algorithm features include the rejection of fast fluctuations in illumination (time averaging), control of speed of response, and control of the sensitivity to small changes. While the default settings are adequate in most situations, the user can program target brightness, measurement window, and other parameters described above. The driver changes AE parameters (integration time, gains, and so on) to drive scene brightness to the programmable target.

To avoid unwanted reaction of AE on small fluctuations of scene brightness or momentary scene changes, the AE track driver uses a temporal filter for luma and a threshold around the AE luma target. The driver changes AE parameters only if the difference between the AE luma target and the filtered luma is larger than the AE target step and pushes the luma beyond the threshold.

Depending on the lighting conditions, the AE may settle in either of the two zones

- Zone 0: This zone is for bright conditions. When in this zone, the integration time is less than one flicker period and analog and digital gains are at minimums. Flicker may may be visible in this zone. If using indoor mode, AE will not go into Zone 0
- Zone 1: This zone is for darker lighting conditions. The integration time is kept as a multiple of the flicker period. Analog gain and digital gains are permitted to reach their maximum values

Figure 29 illustrates the various aspects of Zone 0 and Zone 1.

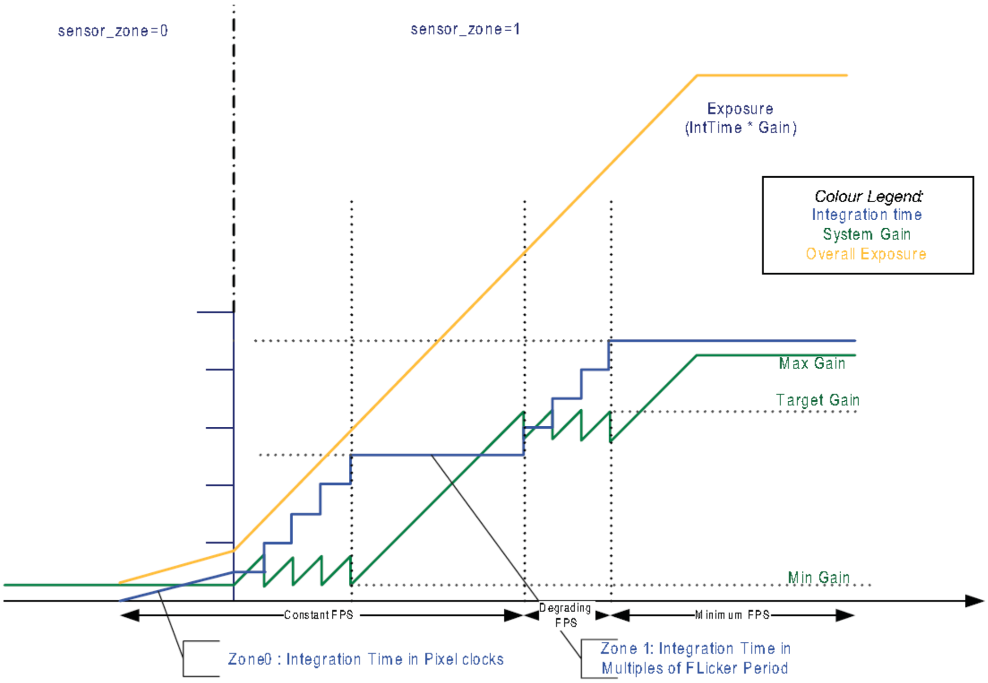


Figure 29. AE Track Zones

Setting the AE

The target average luminance that the AE tries to attain differs based on the brightness value of the scene. The Brightness value can range from 0 to 14; 0 for a very dark scene and 14 for a very bright scene.

The AE may be configured using the AE Base target lookup table (0 to 7). The tables are configured using the AE_TRACK_LOG_Y_TARGET_ variables. Alternatively, the AE can be configured to use only a single brightness target.

Table 22. AE BASE TARGET LOOKUP TABLE

Map	Address	Bits	Name	Description
AE_TRACK	0xA82C to 0xA83A	[15:0]	AE_TRACK_LOG_Y_TARGET(0 to 7)	These variables can be tuned to provide, for example, high noise immunity or high flicker avoidance. This value is unsigned fixed−point with 8 fractional bits

Brightness Metric Offset

The brightness metric offset (cam_ll_bm_offset) is a parameter that is subtracted from the brightness metric (cam_ll_brightness_metric). This calibration would need to be performed once for a camera system, the value would change if the lens f# was changed.

Brightness Metric Offset Calibration

1. The camera should be in the following setup:

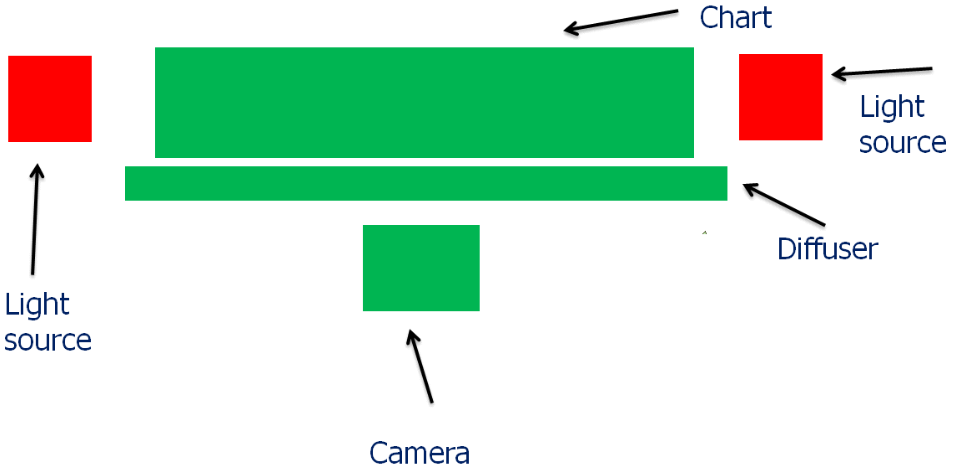


Figure 30. Setup for Measuring BM Offset

2. Configure for the relevant mode
3. Ensure all auto functions are running
4. The light level should be in the range of 700−1000 lux
5. Using a spot meter take a measurement in EV on the diffuser, the spot chosen should be close to the camera
6. Note this value in EV
7. Read cam_ll_brightness_metric
8. Change cam_ll_bm_offset until the value noted in step 6 matches the cam_ll_brightness_metric
9. Change the brightness of the lights and using the spot meter re−measure the EV, it now should track cam_ll_bm_offset

AE Target Tuning (LUT Method)

AE base target tuning is used to tune the target brightness that is desired at various light levels. The light levels are mapped to EV zones.

Equipment

- Studio lights controlled by a dimmer
- Flat field 18% gray target

Tuning Method

1. The brightness Offset needs to be calibrated before the AE Base targets are tuned
2. Run the [AE Base Target Calibration mode] preset
3. Based on Table 23, the user can decide how they wish to tune the AE
4. Note the brightness metric and map it to the EV zone, where EV Zone= (brightness metric/2)
5. Adjust the light level such that the EV zone is 5.0. The point is to have a round number not 5.5 for example
6. Adjust the base target such that you get 18% of full range. For example, on the AR0132, the sensor output is 12−bit, so to get 18% of 2^ 12 = 737 LSB. Monitor stat_average_luma until it reaches 737
7. Increase the light to the next EV zone and repeat step 6
8. Repeat the experiment to cover all EV zone from 0−7

Table 23. AE BASE TARGET TUNING

EV Zone	Low Noise at Low Light	Optimal Noise vs. Detail	More Detail at Low Light
	% of Full Range	% of Full Range	% of Full Range
7	22	22	22
6	20	20	20
5	18	18	18
4	14	14	15
3	10	12	14
2	5	9	13
1	2	7	10
0	1	5	8

AE Target Tuning (Single Target)

If the user prefers to tune AE using a single target method that is also supported.

To configure for this mode then set the following:

- FIELD_WR = AE_TRACK_ALGO, AE_TRACK_EXEC_CALC_TARGET_LUMA, 0
- FIELD_WR = AE_TRACK_MODE, AE_TRACK_AE_MODE_PERCENTILE, 0

The image target would be made ‘brighter’ or ‘darker’ by increasing or decreasing the value of AE_TRACK_AVG_LOG_Y_TARGET.

AE_track_exec_calc_target_luma executes target luma calculation routine. Changes take effect during Vertical Blanking.

When AE_track_ae_mode_percentile bit is enabled AE ensures that high light clipping is within a set tolerance. AE tries to place a histogram high end percentile point below a target value. The amount of highlight clipping permitted varies with the number of pixels in the histogram low end. The more pixels that are in the histogram low end the more important the low end pixels are and thus more clipping is allowed.

AE_track_avg_log_y_target is Luma target in Log2 space.

Variable Frame Rates

AP0102AT supports variable frame rate modes.

Discrete Frame Rate

When this mode is being used, as the scene illumination decreases, the frame rate will decrease to the next programmed value. The variable “Cam_aet_discrete_framerate” is used to enable the discrete frame rate mode.

Table 24. DISCRETE FRAME RATES VARIABLES

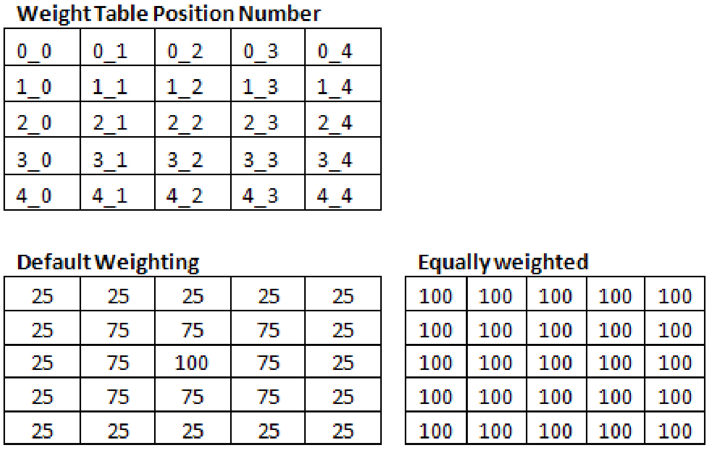
Map	Address	Bits	Name	Description
CamControl	0xC8D4	[15:0]	Cam_aet_frame_rate_0	Discrete mode frame rates in Hertz. Must be less than cam_aet_max_frame_rate and greater than cam_aet_frame_rate_1
CamControl	0xC8D6	[15:0]	Cam_aet_frame_rate_1	Discrete mode frame rates in Hertz. Must be less than cam_aet_frame_rate_0 and greater than cam_aet_frame_rate 2
CamControl	0xC8D8	[15:0]	Cam_aet_frame_rate_2	Must be less than cam_aet_frame_rate_1
CamControl	0xC8D2	[15:0]	Cam_aet_max_frame_rate	The maximum configured frame rate in Hertz (unity = 256). Note this is the firmware−calculated maximum frame−rate as determined by the current sensor configuration

Table 25. EXAMPLE FRAME RATES

Number of Flicker Periods (*8.33 ms)	Frame Period (ms)	Frame Rate (Hz)
1 – 5	33.33	30
5	41.66	24
6	50	20
7	58.33	17.14
8	66.66	15

Zone Weighted AE

The AP0102AT uses the zone weighted algorithm, whereby the frame is divided into multiple zones and the average luminance of each zone is weighted and summed to arrive at the current luminance of the scene. The AE weight table is configured for each position at the ae_rule_ae_weight_table_* (0xA40A to 0xA422).



Weight Table Position Number

0_0	0_1	0_2	0_3	0_4
1_0	1_1	1_2	1_3	1_4
2_0	2_1	2_2	2_3	2_4
3_0	3_1	3_2	3_3	3_4
4_0	4_1	4_2	4_3	4_4

Default Weighting

25	25	25	25	25
25	75	75	75	25
25	75	100	75	25
25	75	75	75	25
25	25	25	25	25

Equally weighted

100	100	100	100	100
100	100	100	100	100
100	100	100	100	100
100	100	100	100	100
100	100	100	100	100

Figure 31. Weight Table Position and Weighting

AE Adaptive Damping

The AE adaptive damping feature adjusts the amount of damping applied to AE while moving towards target luma value. When the difference between target luma and measured luma is large the adaptive damping algorithm tries to settle AE in fewer frames.

The AE damping is controlled with the ae_track variables.

Table 26. AE DAMPING CONTROL

Map	Address	Bits	Name	Description
ae_track	0xA813	[7:0]	ae_track_damp_max	Maximum AE damping. This value is the damping speed when the exposure is near the target (0 is the slowest adaptation). This value is unsigned fixed−point with 5 fractional bits.
ae_track	0xA814	[7:0]	ae_track_damp_slope	Adaptive AE damping slope. This increases the distance between damp_max and damp_min. The smaller the value the larger the distance. This value is unsigned fixed−point with 5 fractional bits. Changes take effect during Vertical Blanking.
ae_track	0xA815	[7:0]	ae_track_damp_min	Minimum AE damping. This value is the damping speed when the exposure is far from the target (1 is the fastest adaptation). This value is unsigned fixed−point with 5 fractional bits. Changes take effect during Vertical Blanking.

MANUAL EXPOSURE AND WHITE BALANCE MODE

The manual exposure and white balance modes are intended to allow simple manual exposure and white balance control by the host. The host sets the CAM_AET_EXPOSURE_TIME_MS, CAM_AET_EXPOSURE_GAIN and CAM_AWB_COLOR_TEMPERATURE controls, the camera will then calculate the appropriate integration times and gains.

There are a number of constraints when using the manual exposure or white−balance modes:

- In manual exposure mode, the auto exposure algorithm does not support flicker avoidance or detection. If flicker is present, then the host is responsible for requesting exposure times that will avoid flicker
- When switching to manual mode from auto mode via a Refresh command it is strongly−recommended that the host waits for two frames after the Refresh completes before changing the exposure or white−balance. It is possible that an internal exposure or white−balance request is in progress when the switch occurs. In this case, the new manual exposure and white−balance settings would not be applied
- The host must always wait for an exposure or white−balance change which was previously initiated to complete, by polling ‘cam_sensor_control_request’, before switching exposure or white−balance mode

Host Controlled Exposure Mode

The Host−Controlled exposure mode is intended to give the host full control over exposure and gains using the CAM_EXP_CTRL_* variables.

In host−controlled exposure mode, the auto exposure algorithm does not support flicker avoidance. If flicker is present, then the host is responsible for requesting exposure times that will avoid flicker.

When switching to host−controlled mode from auto mode via a Refresh command, it is strongly−recommended that the host waits for two frames after the Refresh completes before changing the exposure or white−balance. It is possible that an internal exposure or white−balance request is in progress when the switch occurs such that the new exposure/gain settings will not be applied.

The host must always wait for an exposure or white−balance change it initiated to complete before switching exposure or white−balance mode. Polling must be used to verify completion of the operation.

Example commands for host exposure and white balance

```
[Host Controlled WB]
  FIELD_WR = CAM_AWB_MODE,
  CAM_AWB_MODE_CONTROL, 0x03
  LOAD = Refresh
```

```
[Host Controlled AE]
  FIELD_WR = CAM_AET_AEMODE,
  CAM_AET_MODE_EXPOSURE, 0x03
  LOAD = Refresh
```

```
[Host controlled unity gain white balance example]
  REG = 0xC87E, 0x0080            //
  CAM_EXP_CTRL_CPIPE_DGAIN_RED
  REG = 0xC880, 0x0080            //
  CAM_EXP_CTRL_CPIPE_DGAIN_GREEN1
  REG = 0xC882, 0x0080            //
  CAM_EXP_CTRL_CPIPE_DGAIN_GREEN2
  REG = 0xC884, 0x0080            //
  CAM_EXP_CTRL_CPIPE_DGAIN_BLUE
  REG = 0xC842, 0x02              //
  CAM_SENSOR_CONTROL_REQUEST
  POLL_FIELD =
  CAM_SENSOR_CONTROL_REQUEST, ! = 0,
  DELAY = 50, TIMEOUT = 10
```

```
[Host controlled 2.75x DCG gain exposure mode
example]
  REG = 0xC874, 0x00              //
  CAM_EXP_CTRL_COLUMN_GAIN
  REG = 0xC875, 0x01              //
  CAM_EXP_CTRL_DCG_GAIN
  REG = 0xC842, 0x01              //
  CAM_SENSOR_CONTROL_REQUEST
  POLL_FIELD =
  CAM_SENSOR_CONTROL_REQUEST, ! = 0,
  DELAY = 50, TIMEOUT = 10
```

```
[Host controlled 20 mS exposure mode example]
  REG = 0xC864, 0x05CF            //
  CAM_EXP_CTRL_COARSE_INTEGRATION_TIME
  REG = 0xC842, 0x01              //
  CAM_SENSOR_CONTROL_REQUEST
  POLL_FIELD =
  CAM_SENSOR_CONTROL_REQUEST, ! = 0,
  DELAY = 50, TIMEOUT = 10
```

FLICKER DETECTION AND AVOIDANCE

Most low power fluorescent lights flicker ON and OFF quickly to reduce power consumption. While this fast flickering is marginally detectable by the human eye, it is very noticeable in digital images because the flicker period of the light source is very close to the range of digital images' exposure times. Flicker occurs when the integration time is not an integer multiple of the period of the light intensity. The AP0102AT can be programmed to avoid flicker for 50 or 60 Hertz. For integration times less than the light intensity period (10ms for 50 Hz environment), flicker cannot be avoided. The AP0102AT supports an indoor AE mode, that will ensure flicker−free operation.

In manual exposure mode, the AP0102AT does not provide automatic flicker avoidance. In this case, it is the host that is responsible for controlling exposure times that would avoid flicker.

Flicker avoidance in a HDR sensor relates to T1 only. Flicker in T2 and T3 cannot be avoided.

Table 27. FLICKER AVOIDANCE CONTROL

Map	Address	Bits	Name	Description
CamControl	0xC8D1	[7:0]	CAM_AET_FLICKER_FREQ_HZ	The desired flicker avoidance frequency in Hertz (50Hz or 60Hz)

Flicker Detection

The AP0102AT device has a flicker detection algorithm designed only to detect a 50 Hz or 60 Hz flicker source. The algorithm is based on frame differences, and it is only able to search for and detect the opposite frequency to the one that it is currently configured to avoid. For example, if the device is running with the flicker avoidance set to 60 Hz, the flicker detection will only search for a 50 Hz flicker source.

Table 28. FLICKER DETECTION CONTROL VARIABLES

Map	Address	Bits	Name	Description
CamControl	0xCAB4	[0]	CAM_FLICKER_DETECT_FD_ENABLE	Enable flicker detection: 0: Disable flicker detection 1: Enable flicker detection
CamControl	0xCAB4	[1]	CAM_FLICKER_DETECT_FR_AUTOSWITCH	Auto−switch flicker avoidance period control: 0: Auto switching disabled 1: Enable automatic switching of the flicker period when a flicker source is detected in the scene

COLOR TUNING

This section discusses the color tuning functions: White Balance (WB), Color Correction Matrix (CCM), color saturation, Auto White Balance (AWB), and manual white balance. It also covers the procedure for generating AWB and CCM settings using the SensorTune tool.

White Balance

Color temperature is a way of measuring the characteristic of a light source. It is based on the ratio of the amount of blue light component to the amount of red light component, and the green light component is ignored. A white object may not appear the same "white" under lights with different color temperatures. White balance is the process of removing unrealistic color casts, so that objects which appear white to the human eye are rendered white in the image. White balance is the first step for achieving desired color rendition results.

Color Correction Matrix

To achieve good color rendition and color saturation, interpolated colors of all pixels are subjected to color correction. The color correction is a linear transformation of the image with a 3 x 3 color correction matrix.

The optimal values of the correction matrix elements depend on the spectrum of light incident on the sensor. They can be either programmed by the host or automatically selected by the auto white balance algorithm.

The color correction matrix consists of nine values, each of which represents a digital gain factor on the corresponding color channel with the diagonal elements representing the gain factors on each color channel and the off diagonal terms representing the gain factors to compensate for color crosstalk. The matrix is normalized so that the sum of each row is "1." All the color correction matrix values are stored in the AWB variable map.

Table 29. COLOR CORRECTION MATRIX STRUCTURE

Saturation Type/Gain	Gain R	Gain G	Gain B
Saturation Red	ccmL[0]	ccmL[1]	ccmL[2]
Saturation Green	ccmL[3]	ccmL[4]	ccmL[5]
Saturation Blue	ccmL[6]	ccmL[7]	ccmL[8]

The AP0102AT supports three color correction matrices: one is for the red−rich illumination (normally 2856 K color temperature and named left matrix), the middle one is aimed for fluorescent light (normally 3850 K color temperature), and the third one is for blue rich illumination (normally would be 6500K color temperature and named right matrix).

All matrices have associated R/G and B/G ratios (cam_awb_ccm_[l/m/r]_[r/b]g_gain) describing the gain needed for the red and blue channels on those particular color temperatures. Also, each CCM has a color temperature associated with it (cam_awb_ccm_[l/m/r]_ctemp). Using the color temperature (cam_awb_color_temperature), the left and middle or middle and right matrices are mixed using linear interpolation to generate the CCM for the next frame.

The interpolated RGB values are transformed by the color correction matrix (CCM) into color−corrected R', G', and B' values. The color correction matrix is uploaded by the AWB firmware driver into the corresponding registers in the color pipeline when AWB has settled and the White Balance has adjusted.

Auto White Balance

The AP0102AT has a built−in Auto White Balance (AWB) algorithm designed to compensate for the effects of changing spectra of the scene illumination on the quality of the color rendition When these illuminants are used to light a scene, neutral colors are no longer represented as neutral in a camera system because of the color cast of the reflected light. The image capture device would need to compensate for this color cast in order to more accurately reproduce the original scene. The compensation is called AWB.

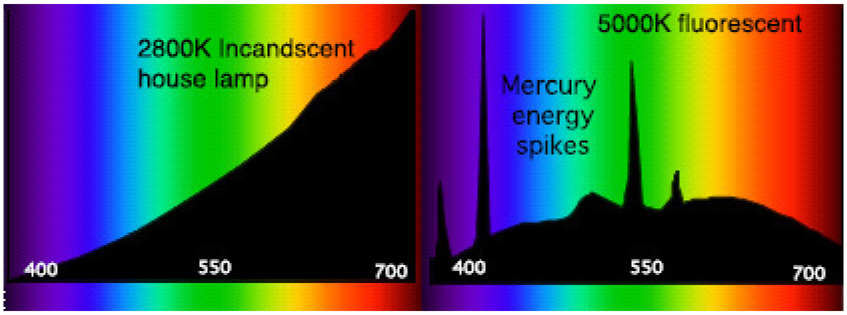


Figure 32. Illuminant Spectral Response Samples

The algorithm consists of two major parts: a measurement engine performing statistical analysis of the image and a driver performing the selection of the optimal color correction matrix and digital gain. While default settings of these algorithms are adequate in most situations, the user can reprogram base color correction matrices; place limits on color channel gains, and control the speed of both matrix and gain adjustments. The AWB displays the current AWB position in color temperature, the range of which will be defined when programming the CCM matrices.

Setting the AWB Mode

The device supports four modes of operation.

In Auto white balance mode, the auto white balance algorithm is responsible for the calculating the color temperature of the scene and applying the correct red and blue gains.

Triggered auto white balance mode is intended for multi−camera usage where a host is controlling the white balance of a number of cameras.

Manual white balance is intended to allow simple manual white balance control by the host.

The host−controlled mode will allow the host to have full control over white balance.

Refer to the section, “Manual Exposure and White Balance Mode” for additional details.

Table 30. AWB Mode Control

Map	Address	Bits	Name	Description
CamControl	R0xC97D]	[2:0]	CAM_AWB_MODE_CONTROL	Selects the white−balance operation mode: 0: Auto White Balance 1: Triggered Auto White Balance 2: Manual White Balance 3: Host−Controlled 4..7: Reserved

How to Speed Up or Slow Down AWB

It is possible to speed up or slow down the AWB, this is achieved by changing the value of awb_pre_awb_ratios_tracking_speed (0xAC16). A value of 32 is the fastest and 1 is the slowest.

Care should be taking when tuning this value, as setting the AWB damping too fast could cause the AWB to be unstable.

Setting of AWB Gains

This is a tuning step that could be performed after using Sensor Tune if fine tuning is required.

Table 31. VARIABLES FOR SETTING AWB GAINS

Variable	Name	Function
R0xAC12	awb_r_gain	Current Red channel gain
R0xAC14	awb_b_gain	Current Blue channel gain
R0xC8C8	cam_awb_ccm_l_rg_gain	R/G gain ratio for Left Matrix
R0xC8CA	cam_awb_ccm_l_bg_gain	B/G gain ratio for Left Matrix
R0xC8CC	cam_awb_ccm_m_rg_gain	R/G gain ratio for Middle Matrix
R0xC8CE	cam_awb_ccm_m_bg_gain	B/G gain ratio for Middle Matrix
R0xC8D0	cam_awb_ccm_r_rg_gain	R/G gain ratio for Right Matrix
R0xC8D2	cam_awb_ccm_r_bg_gain	B/G gain ratio for Right Matrix

In this example it shall be assumed that Alight is being used for Left Matrix, CWF for middle matrix and D65 for Right Matrix.

1. Get demo camera/module imaging
2. Set the scene with ISO chart/ Macbeth chart
3. Ensure the camera is framed up to the relevant ISO markers for the resolution chosen
4. Starting with A light; note the values of awb_r_gain and awb_b_gain
5. Change to CWF; note the values of awb_r_gain and awb_b_gain
6. Change to D65; note the values of awb_r_gain and awb_b_gain
7. The values noted can now be used instead of the values determined by Sensor Tune

How to Produce Color−Tinted Effects

In the AWB driver, there are several variables that will produce global color offsets (tints) to the images. These variables operate on the color correction matrix as a normalization coefficient. For the left (incandescent) matrix, the CCM is computed as shown in Figure 33. Some color tints can be applied to the output of the CCM. However instead of using a new gain module the CCM itself is updated to get extra gain for the Red, Green and Blue channels. There are two sets of tint variables (cam_awb_k_r_l, cam_awb_k_g_l, cam_awb_k_b_l, and cam_awb_k_r_r, cam_awb_k_g_r, cam_awb_k_b_r). The first set (*_l) is for the "left", is for red−rich illumination and the other set (*_r) is for the "right", is for blue−rich. The actual tint applied is interpolated using cam_awb_color_temperature.

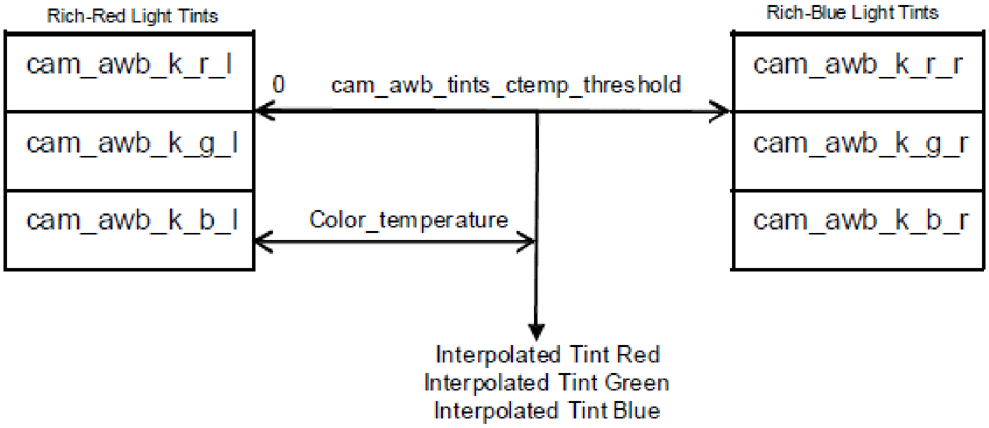


Figure 33. Calculation for Tints Applied in the CCM

Table 32. COLOR−TINT VARIABLES FOR COLOR−TINTED EFFECTS

Map	Address	Name	Function
CamControl	R0xC91E	cam_awb_ccm_l_ctemp	Color temperature for Left Matrix (in Kelvin)
CamControl	R0xC928	cam_awb_color_temperature	Current matrix color temperature (in Kelvin)
CamControl	R0xC980	cam_awb_tints_ctemp_threshold	Color temperature threshold for color tint application
CamControl	R0xC982	cam_awb_k_r_l	Controls the tint for the red channel at the color temperature set by cam_awb_ccm_l_ctemp
CamControl	R0xC983	cam_awb_k_g_l	Controls the tint for the green channel at the color temperature set by cam_awb_ccm_l_ctemp
CamControl	R0xC984	cam_awb_k_b_l	Controls the tint for the blue channel at the color temperature set by cam_awb_ccm_l_ctemp
CamControl	R0xC985	cam_awb_k_r_r	Controls the tint for the red channel at the color temperature set by cam_awb_tints_ctemp_threshold
CamControl	R0xC986	cam_awb_k_g_r	Controls the tint for the green channel at the color temperature set by cam_awb_tints_ctemp_threshold
CamControl	R0xC987	cam_awb_k_b_r	Controls the tint for the blue channel at the color temperature set by cam_awb_tints_ctemp_threshold

The functionality is summarized below:

- When cam_awb_color_temperature is equal to cam_awb_ccm_l_ctemp, only the red light tint matrix (*_l) has effects
- When cam_awb_color_temperature is larger than cam_awb_ccm_l_ctemp, but smaller than cam_awb_tints_ctemp_threshold, both red−light and blue−light tint matrices are used for interpolation to generate the current color tint effect;
- When cam_awb_color_temperature is larger than cam_awb_tints_ctemp_threshold, only the blue−light tint matrix (*_r) has effects

Hue PCR Tuning

Even after calibrating the CCM to convert the device color space into device independent color space, there could be specific colors that do not represent an individual's preference for memory colors such as green trees or grass, blue sky, etc. Certain hue ranges are preferred to be rendered slightly differently to better represent the preferred color rendition. The Preferred Color Reproduction (PCR) feature allows the user to saturate and or change the hue of a color range in Cr−Cb space.

Table 33. HUE PCR REGISTER DESCRIPTION

Page	Register	Name	Notes
CPIPE YUV PIPE REGS	0x3400 to 0x3422	HUE1_Q1Q2 to HUE18_Q3Q4	Hue Rotation angle regions for Q1, Q2, Q3, and Q4
CPIPE YUV PIPE REGS	0x3424 to 0x346A	PCR_COLOR_GAIN1_REGION_1 to PCR_COLOR_GAIN9_REGION_36	PCR saturation gain region 1 to 36.Not recommends changing

Equipment

- Color Checker Chart
- Dimmable lamps
- Pastel and saturated props
- DevWare

Test Cases

- Color Checker Chart evenly illuminated
- Illumination levels: ~1500 lux down to ~10 lux

Tuning Procedure

It is recommended that the user to verify all colors after final tuning of CCM and AWB and then decide if there are additional PCR regions that could benefit from manual tuning as per their preferences.

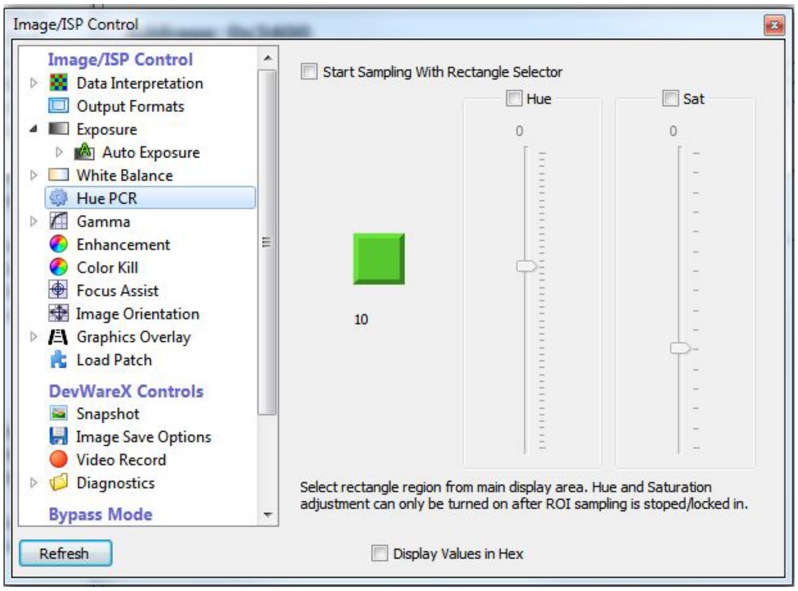
Manual Tuning

The PCR tuning tool provided in DevWare's Sensor Control panel is an easy way to manually manipulate the PCR color gains. This method assumes that the customer has not yet defined their desired output color preference. Given that color saturation and hue change is preference based, there is no absolute method for tuning.

1. Bring up the sensor in DevWare, using all the calibrated settings
2. Set up a scene using a color checker and other colorful objects. Sample shown in the following image

3. Launch DevWare

4. In the Sensor Control Panel, click on Hue PCR



□ Start Sampling With Rectangle Selector

□ Hue

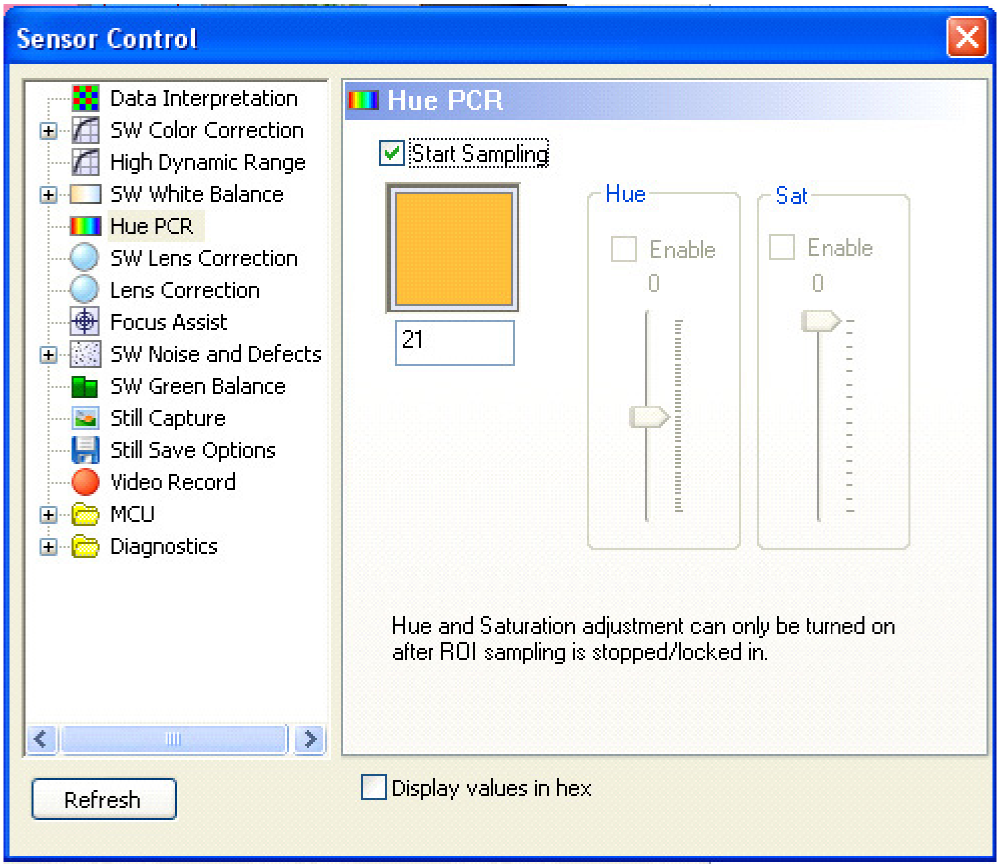
□ Sat

Select rectangle region from main display area. Hue and Saturation adjustment can only be turned on after ROI sampling is stoped/locked in.

□ Display Values in Hex

5. Check the ‘Start Sampling’ box to determine the subsequent PCR region for the selected region of interest. This will disable the Hue PCR feature while you select the region of interest

6. Using the rectangle Mouse Selection feature, select a region of interest that represents the color that you would like to manipulate



☑ Start Sampling

21

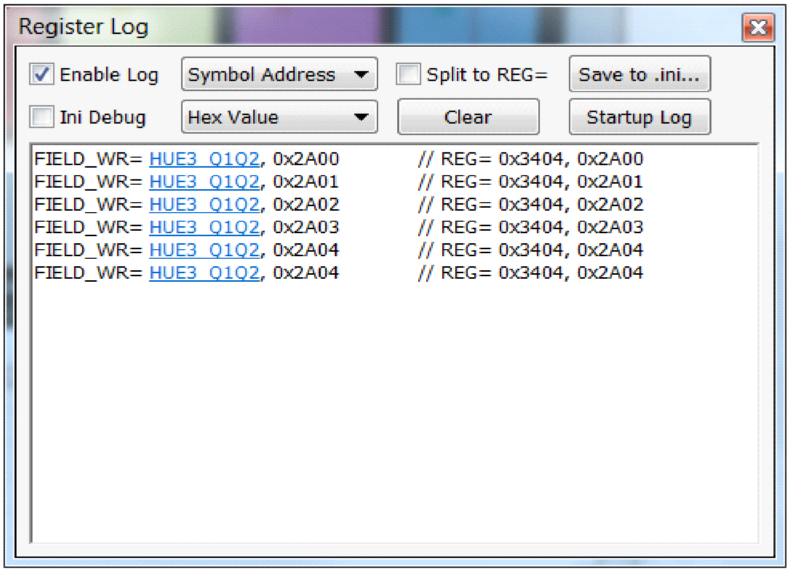
□ Enable

□ Enable

Hue and Saturation adjustment can only be turned on after ROI sampling is stopped/locked in.

□ Display values in hex

7. Once you have finished selecting the region of interest, click on the ‘Start Sampling’ box to disable it
8. Using the sliders, find settings that are pleasing. Be sure to take care that the color gain is not so high as to cause undesirable noise in any of the patches and that the hue change is not too drastic to cause chromatic contouring. It is recommended that the Saturation slider is not used
9. Launch ‘Register Log’ window and tick ‘Enable Log’ option as shown below. Copy the final Hue register value and paste in PRESET of INI file



☑ Enable Log Symbol Address □ Split to REG=

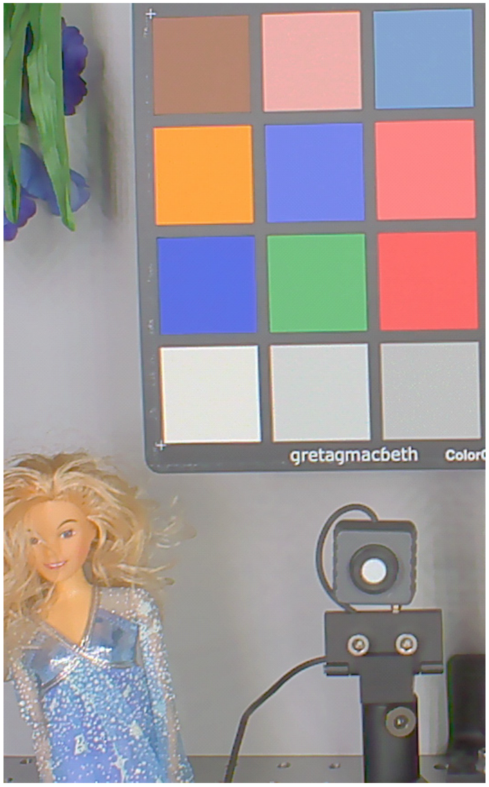
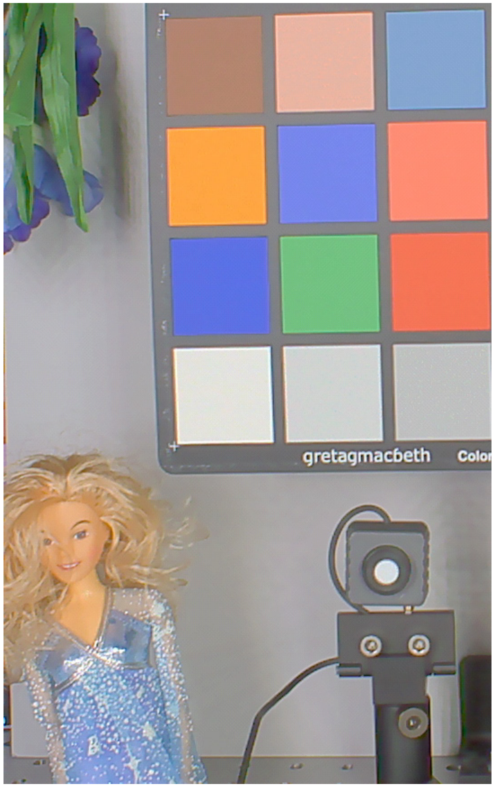
□ Ini Debug Hex Value

```
FIELD_WR= HUE3_Q1Q2, 0x2A00          // REG= 0x3404, 0x2A00
FIELD_WR= HUE3_Q1Q2, 0x2A01          // REG= 0x3404, 0x2A01
FIELD_WR= HUE3_Q1Q2, 0x2A02          // REG= 0x3404, 0x2A02
FIELD_WR= HUE3_Q1Q2, 0x2A03          // REG= 0x3404, 0x2A03
FIELD_WR= HUE3_Q1Q2, 0x2A04          // REG= 0x3404, 0x2A04
FIELD_WR= HUE3_Q1Q2, 0x2A04          // REG= 0x3404, 0x2A04
```

10. Take multiple scene captures and verify that the settings are appropriate

Hue PCR Trade−offs

With the HUE PCR controls it is possible to optimize the hue within one PCR region for a particular color (in the example below the red primary patch was optimized) while negatively impacting the hue of another particular color within the same PCR region (in the example below the lighter flesh tone patch is in the same PCR region as the red primary patch).

Hue Optimized for red primary patch	Hue Optimized for light flesh tone patch
	

Color Kill

To remove high−or low−light color artifacts in low light, a color kill circuit operates in YUV color space to de−saturate, and therefore suppress, chromatic noise and can be used for fading the image to gray at low light.

It affects only pixels whose luminance exceeds a certain preprogrammed threshold. The U and V values of those pixels are attenuated proportionally to the difference between their luminance and the threshold.

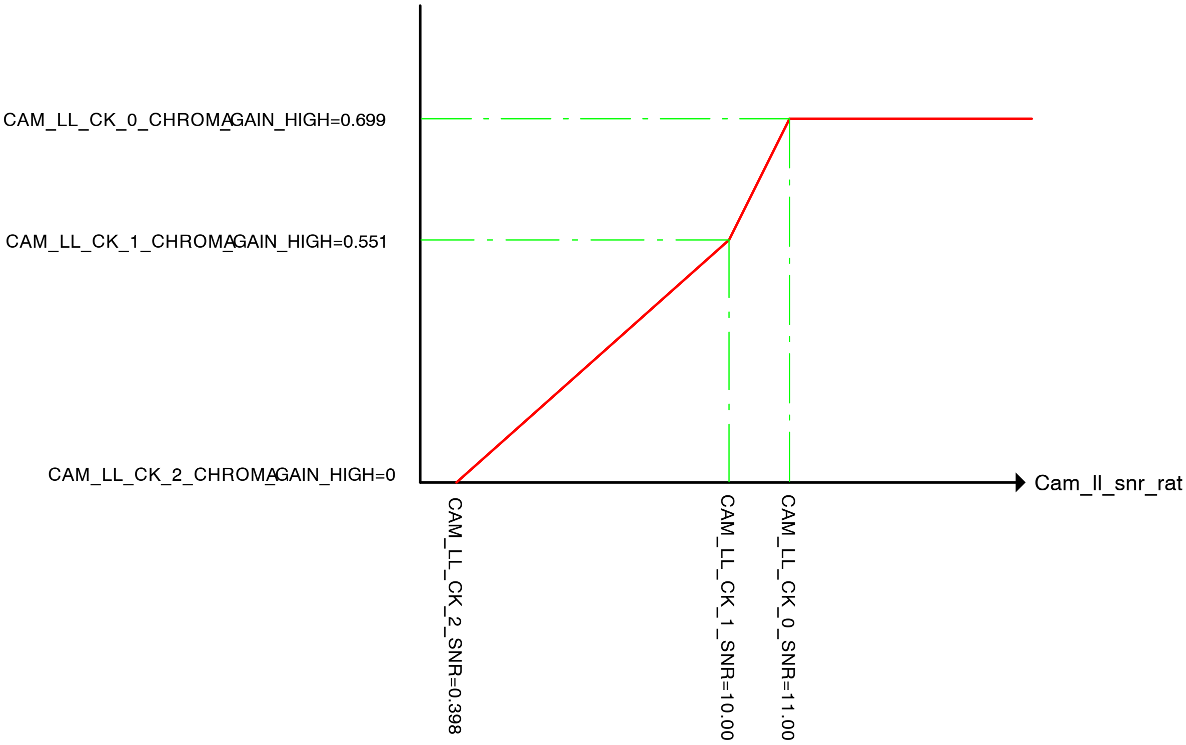


Figure 34. Color Kill Operation

Table 34. COLOR KILL VARIABLES

Map	Address	Bits	Name	Description
CamControl	R0xCA9C	[15:0]	CAM_LL_CK_0_SNR	Low SNR colorkill solution. This is the SNR metric (cam_ll_snr_metric) value used to generate the current colorkill solution. The current colorkill solution is interpolated from the table of colorkill solutions (cam_ll_ck_N*) in the CAM page
CamControl	R0xCAA4	[15:0]	CAM_LL_CK_0 _CHROMA_GAIN_HIGH	Low SNR colorkill solution. The chroma gain applied to a pixel is determined from that pixels colorkill metric value
CamControl	R0xCAA8	[15:0]	CAM_LL_CK_1_SNR	Mid SNR colorkill solution. This is the SNR metric (cam_ll_snr_metric) value used to generate the current colorkill solution. The current colorkill solution is interpolated from the table of colorkill solutions (cam_ll_ck_N*) in the CAM page
CamControl	R0xCAB0	[15:0]	CAM_LL_CK_1 _CHROMA_GAIN_HIGH	Mid SNR colorkill solution. The chroma gain applied to a pixel is determined from that pixels colorkill metric value
CamControl	R0xCAB4	[15:0]	CAM_LL_CK_2_SNR	High SNR colorkill solution. This is the SNR metric (cam_ll_snr_metric) value used to generate the current colorkill solution. This would be used to determine when the image would fade to gray

The impact of color kill on the image is shown below:

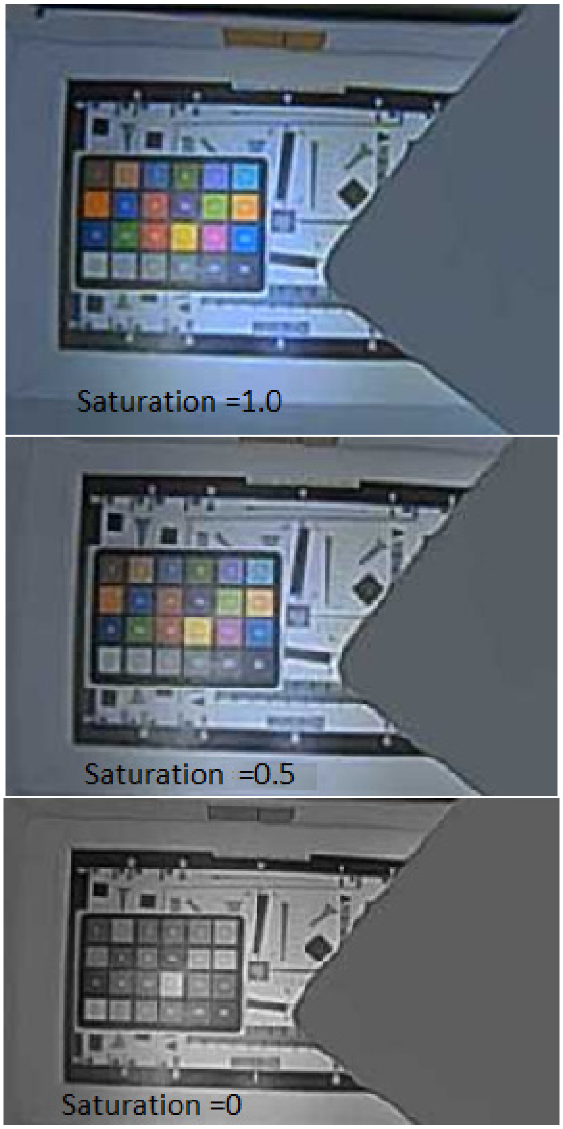


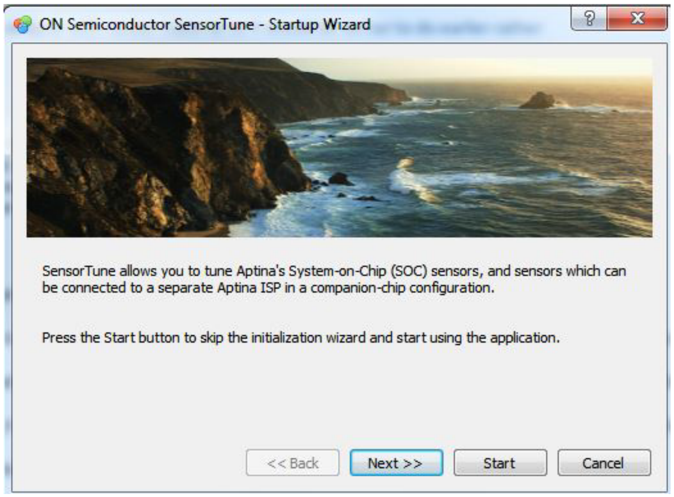
Figure 35. Color Kill

Generating AWB and CCM Using Sensor Tune

NOTE: This section explains the tuning flow for performing AWB tuning in the field; at this point in time, sensor tune only gives a starting point. The FAE team will need to perform some level of tuning for the customer (the level of tuning will depend on customer, market and application).

This section explains how to use the SensorTune tool to produce AWB and CCM settings for AP0102AT. The procedure is as follows:

1. Go to *Start* → *All Programs* → *ON Semiconductor* → *Tools* → *Sensor Tune (AWB and CCM)*, and click on *Next >>*.

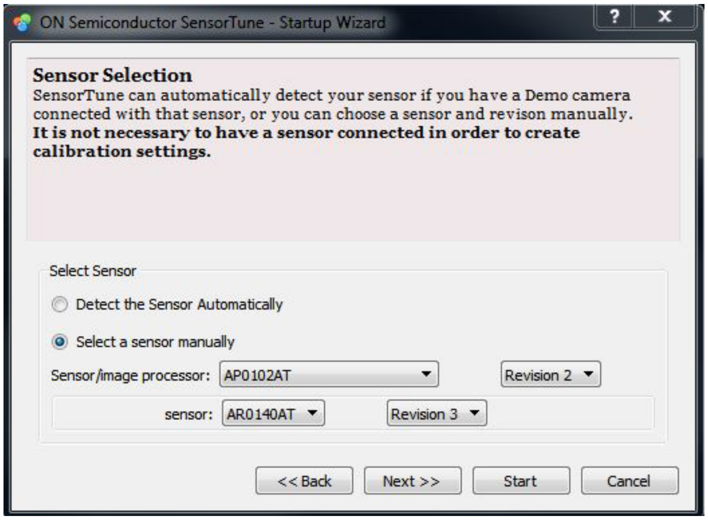


ON Semiconductor SensorTune - Startup Wizard

SensorTune allows you to tune Aptina's System-on-Chip (SOC) sensors, and sensors which can be connected to a separate Aptina ISP in a companion-chip configuration.

Press the Start button to skip the initialization wizard and start using the application.

2. Choose either *Detect the Sensor Automatically* or *Select a sensor manually.*



ON Semiconductor SensorTune - Startup Wizard

Sensor Selection

SensorTune can automatically detect your sensor if you have a Demo camera connected with that sensor, or you can choose a sensor and revison manually.
It is not necessary to have a sensor connected in order to create calibration settings.

Select Sensor

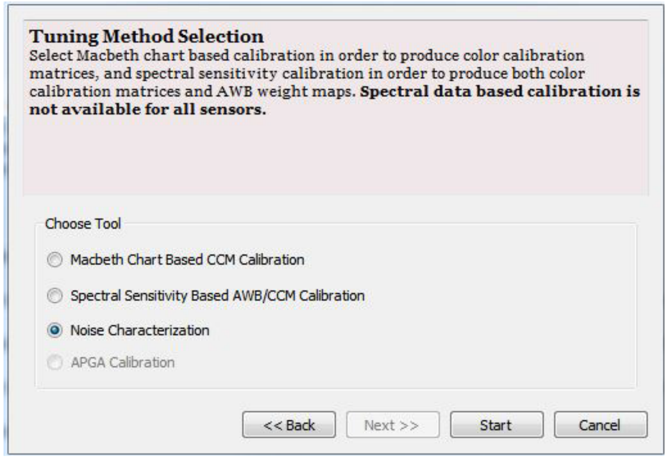
○ Detect the Sensor Automatically

● Select a sensor manually

Sensor/image processor: AP0102AT   Revision 2

sensor: AR0140AT   Revision 3

If selecting manually then the user needs to select the "image processor", "image sensor" and "Revision".



Tuning Method Selection

Select Macbeth chart based calibration in order to produce color calibration matrices, and spectral sensitivity calibration in order to produce both color calibration matrices and AWB weight maps. **Spectral data based calibration is not available for all sensors.**

Choose Tool

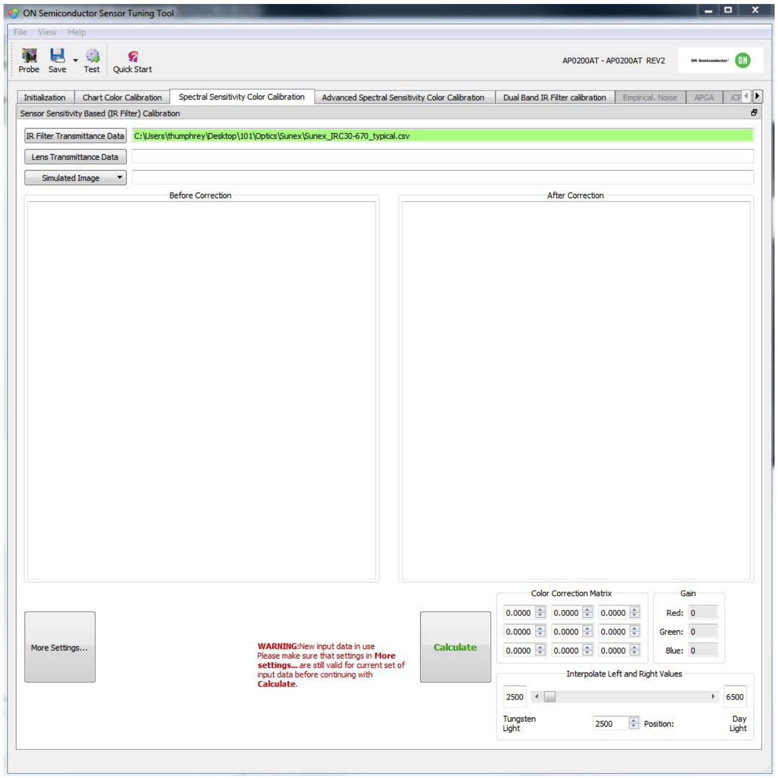
○ Macbeth Chart Based CCM Calibration

○ Spectral Sensitivity Based AWB/CCM Calibration

● Noise Characterization

○ APGA Calibration

3. Select *Spectral Sensitivity based AWB Calibration*, and then click *Start*
4. Load the IR Filter Transmittance Data (mandatory) and Lens Transmittance Data (optional). The IR filter data is typically saved as a text or csv file. There are two columns in the data:
   - The first column is the wavelength in nm
   - The second column is the IR filter value (normalized or in percentage) at the corresponding wavelength
   - An example of the correct csv format is in the 'quick start' tab of SensorTune. When the IRCF has been added successfully the Calculate button should turn green



ON Semiconductor Sensor Tuning Tool
File
View
Help
Probe
Save
Test
Quick Start
AP0200AT - AP0200AT REV2
ON Semiconductor
Initialization
Chart Color Calibration
Spectral Sensitivity Color Calibration
Advanced Spectral Sensitivity Color Calibration
Dual Band IR Filter calibration
Empirical. Noise
APGA
iCP
Sensor Sensitivity Based (IR Filter) Calibration
Before Correction
After Correction
More Settings...
Calculate

IR Filter Transmittance Data: C:\Users\thumphrey\Desktop\101\Optics\Sunex\Sunex_IRC30-670_typical.csv

Lens Transmittance Data:

Simulated Image:

WARNING: New input data in use Please make sure that settings in **More settings...** are still valid for current set of input data before continuing with **Calculate**.

Color Correction Matrix

0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000

Gain

Red: 0

Green: 0

Blue: 0

Interpolate Left and Right Values

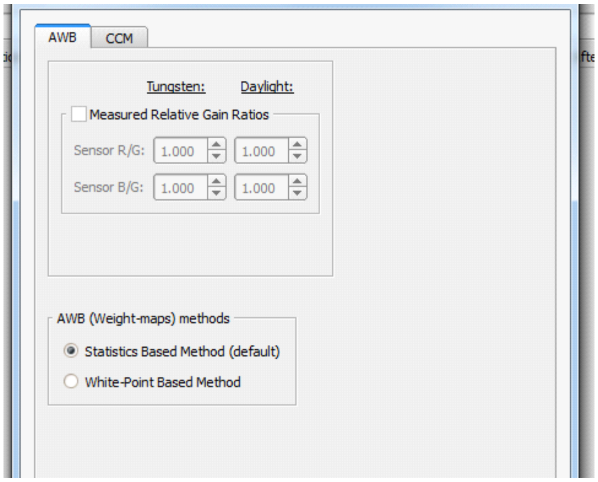
2500 — 6500

Tungsten Light

2500 Position:

Day Light

5. Click on *More Settings* to bring up a new menu



Tungsten: Daylight:

□ Measured Relative Gain Ratios

Sensor R/G: 1.000 1.000

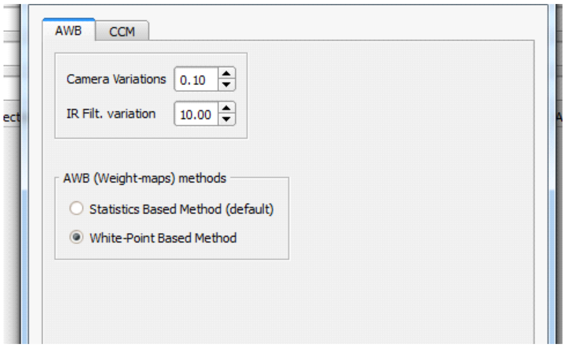
Sensor B/G: 1.000 1.000

AWB (Weight-maps) methods

● Statistics Based Method (default)

○ White-Point Based Method

6. Select *White−Point Based Method*



Camera Variations 0.10

IR Filt. variation 10.00

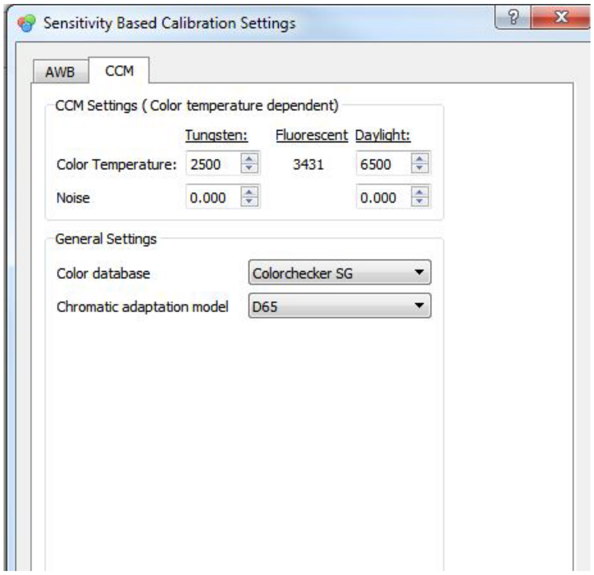
AWB (Weight-maps) methods

○ Statistics Based Method (default)

● White-Point Based Method

a. Camera variation− This is the variation in percentage (0.10 = 10%) of the image sensor color ratios. The user should not need to change this value

b. IR filter variation− This is the variation in nm for the IRCF filter used



CCM Settings (Color temperature dependent)

	Tungsten:	Fluorescent	Daylight:
Color Temperature:	2500	3431	6500
Noise	0.000		0.000

General Settings

Color database: Colorchecker SG

Chromatic adaptation model: D65

c. Color Temperature
These fields specify the color temperatures of illuminants to use for the two color correction matrices. Default illuminant for the Daylight matrix is 6500K CIE D65, and default for the Tungsten matrix is 2500 K black body

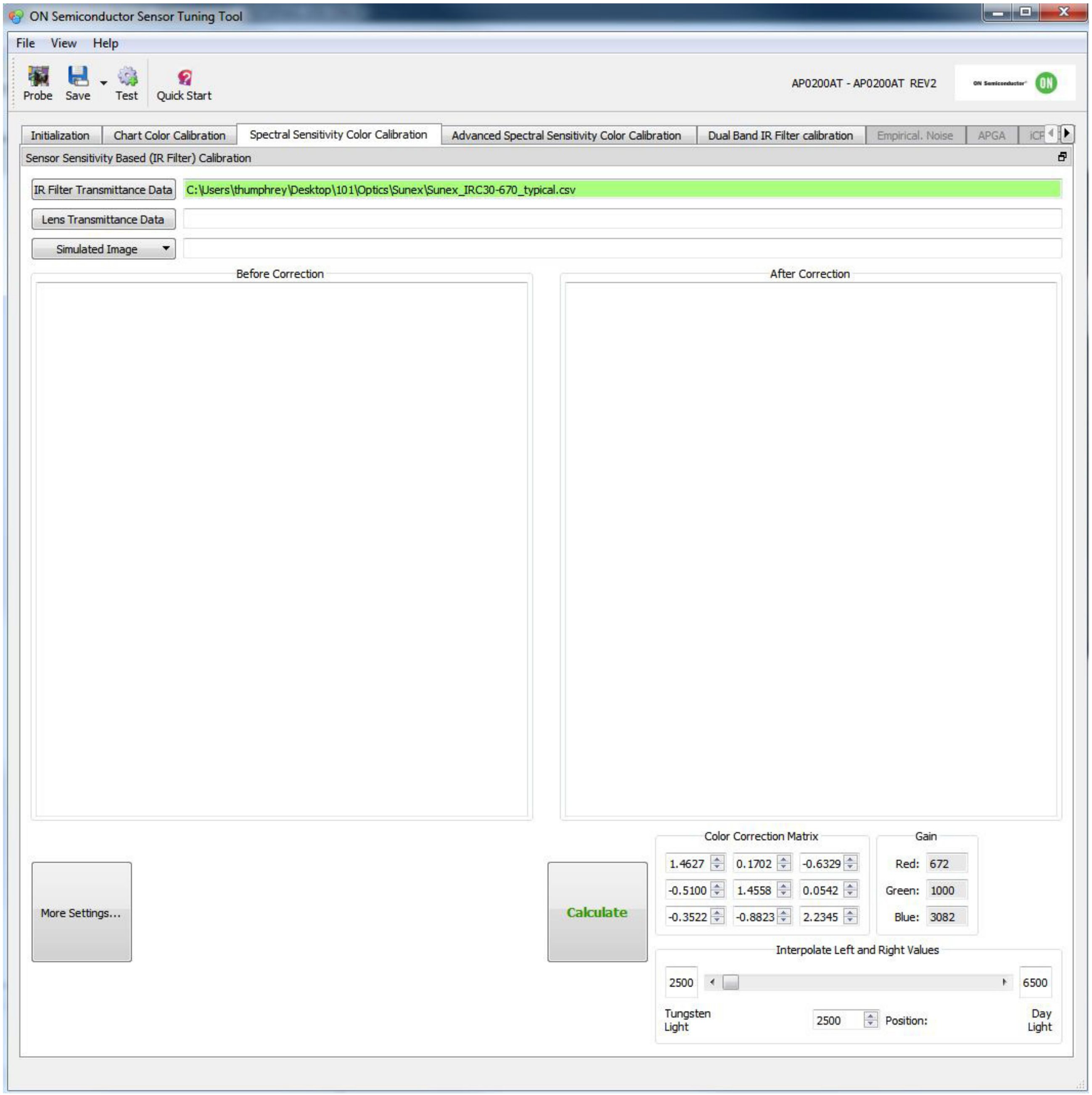
d. Lum. Level
These fields specify the estimated scene luminance level, in units of cd/m2 (candelas per square meter). Specifying a scene light level of 100 cd/m2 will result in the "normal" saturation, that is, adjustment for illuminant chromaticity only, but no saturation adjustment. Using a higher scene luminance level will result in saturation enhancement, and lower will result in de−saturation. The default luminance level for the daylight matrix is 100, that is, no saturation adjustment, assuming that daylight images are likely to have decent SNR. The default saturation level for tungsten is 25, which results in some de−saturation, to account for the typically lower lighting level associated with tungsten illumination. This can be changed easily by editing the luminance level field. The saturation range can be controlled to as low as 30−40% with luminance level of <10 and as high as 120% with luminance level >200. Note that excessive saturation enhancement and de−saturation is not recommended for color matrix generation.

e. Noise
The default noise parameters are 0 for Tungsten and D65, respectively. These are set to give best color reproduction

f. Color database−
This option is to select which color database is used for CCM generator. The default is ON Semiconductor object dataset which includes 177 natural colors and color patches. ON Semiconductor objects may provide better color reproduction for memory colors, such as sky, foliage, and skin tones. This is recommended for ON Semiconductor products. The other option is the Macbeth ColorChecker SG color database in which color samples are more uniformly distributed. The Macbeth ColorChecker SG database may provide slightly better color reproduction for color patches when using a lighting booth (for example, Judge II)

7. Click on *Calculate* to generate the AWB and CCM matrices

NOTE: Recommended settings are shown as above for best starting point for CCM.



IR Filter Transmittance Data: C:\Users\thumphrey\Desktop\101\Optics\Sunex\Sunex_IRC30-670_typical.csv

Color Correction Matrix

1.4627	0.1702	-0.6329
-0.5100	1.4558	0.0542
-0.3522	-0.8823	2.2345

Gain

Red:	672
Green:	1000
Blue:	3082

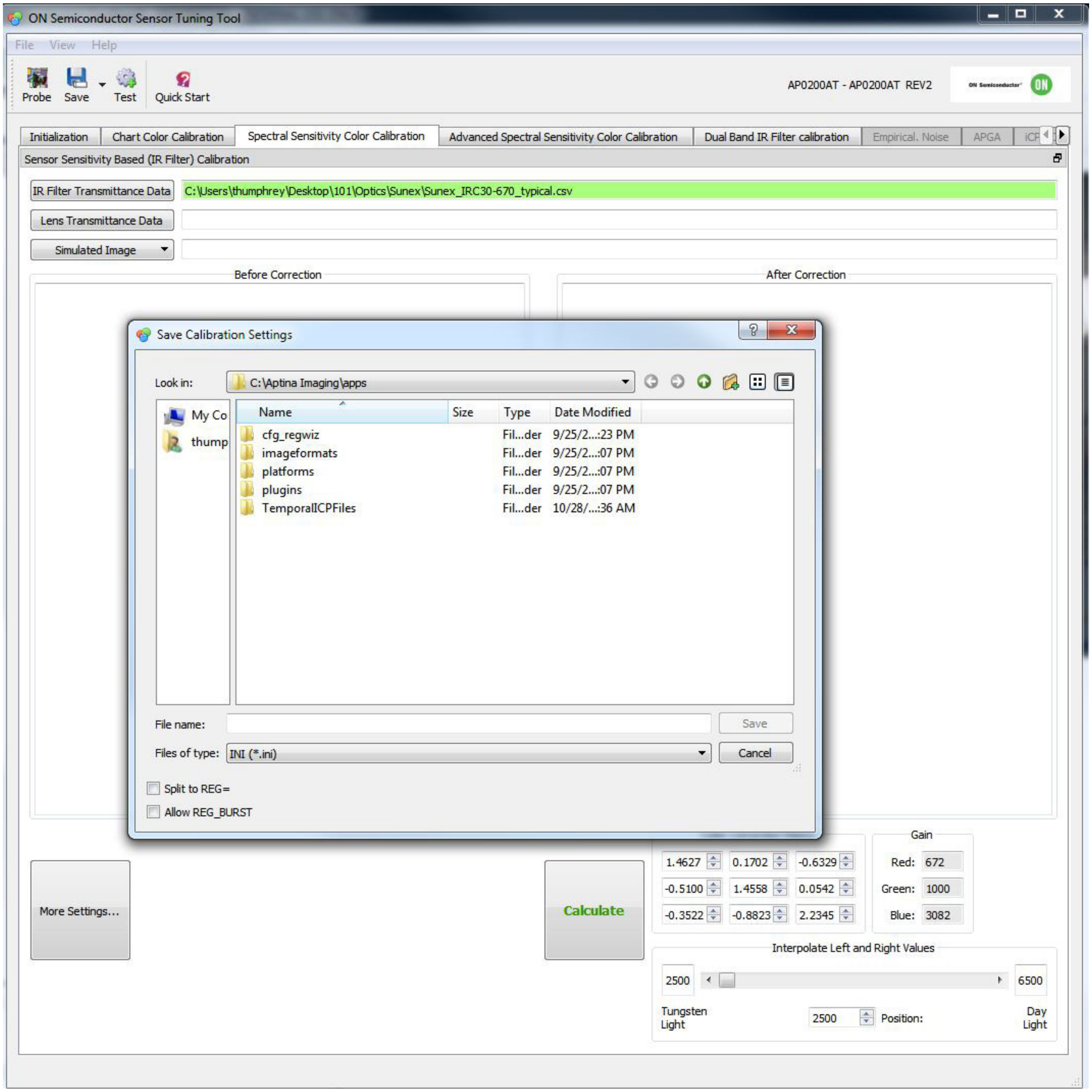
Interpolate Left and Right Values

2500 Tungsten Light

6500 Day Light

Position: 2500

8. Save the settings to an ini file



IR Filter Transmittance Data: C:\Users\thumphrey\Desktop\101\Optics\Sunex\Sunex_IRC30-670_typical.csv

Look in: C:\Aptina Imaging\apps

Name	Size	Type	Date Modified
cfg_regwiz		Fil...der	9/25/2...:23 PM
imageformats		Fil...der	9/25/2...:07 PM
platforms		Fil...der	9/25/2...:07 PM
plugins		Fil...der	9/25/2...:07 PM
TemporalICPFiles		Fil...der	10/28/...:36 AM

File name:

Files of type: INI (*.ini)

□ Split to REG=

□ Allow REG_BURST

1.4627	0.1702	-0.6329
-0.5100	1.4558	0.0542
-0.3522	-0.8823	2.2345

Gain

Red:	672
Green:	1000
Blue:	3082

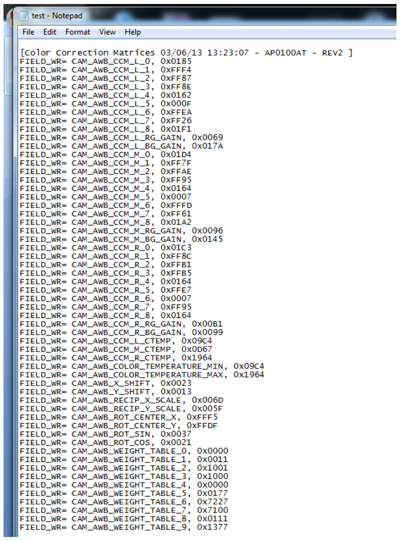
Interpolate Left and Right Values: 2500 – 6500

Tungsten Light 2500 Position: Day Light

9. Sensor tune creates two presets in the ini file, we are only interested in the first preset

VALIDATION OF AWB

1. Get the relevant sensor and image co−processor imaging, run DevWare or Demo Init. Run the first preset in the INI file, which was saved separately



```
[Color Correction Matrices 03/06/13 13:23:07 - AP0100AT - REV2 ]
FIELD_WR= CAM_AWB_CCM_L_0, 0x0185
FIELD_WR= CAM_AWB_CCM_L_1, 0xFFF4
FIELD_WR= CAM_AWB_CCM_L_2, 0xFF87
FIELD_WR= CAM_AWB_CCM_L_3, 0xFF8E
FIELD_WR= CAM_AWB_CCM_L_4, 0x0162
FIELD_WR= CAM_AWB_CCM_L_5, 0x000F
FIELD_WR= CAM_AWB_CCM_L_6, 0xFFEA
FIELD_WR= CAM_AWB_CCM_L_7, 0xFF26
FIELD_WR= CAM_AWB_CCM_L_8, 0x01F1
FIELD_WR= CAM_AWB_CCM_L_RG_GAIN, 0x0069
FIELD_WR= CAM_AWB_CCM_L_BG_GAIN, 0x017A
FIELD_WR= CAM_AWB_CCM_M_0, 0x01D4
FIELD_WR= CAM_AWB_CCM_M_1, 0xFF7F
FIELD_WR= CAM_AWB_CCM_M_2, 0xFFAE
FIELD_WR= CAM_AWB_CCM_M_3, 0xFF95
FIELD_WR= CAM_AWB_CCM_M_4, 0x0164
FIELD_WR= CAM_AWB_CCM_M_5, 0x0007
FIELD_WR= CAM_AWB_CCM_M_6, 0xFFFD
FIELD_WR= CAM_AWB_CCM_M_7, 0xFF61
FIELD_WR= CAM_AWB_CCM_M_8, 0x01A2
FIELD_WR= CAM_AWB_CCM_M_RG_GAIN, 0x0096
FIELD_WR= CAM_AWB_CCM_M_BG_GAIN, 0x0145
FIELD_WR= CAM_AWB_CCM_R_0, 0x01C3
FIELD_WR= CAM_AWB_CCM_R_1, 0xFF8C
FIELD_WR= CAM_AWB_CCM_R_2, 0xFFB1
FIELD_WR= CAM_AWB_CCM_R_3, 0xFFB5
FIELD_WR= CAM_AWB_CCM_R_4, 0x0164
FIELD_WR= CAM_AWB_CCM_R_5, 0xFFE7
FIELD_WR= CAM_AWB_CCM_R_6, 0x0007
FIELD_WR= CAM_AWB_CCM_R_7, 0xFF95
FIELD_WR= CAM_AWB_CCM_R_8, 0x0164
FIELD_WR= CAM_AWB_CCM_R_RG_GAIN, 0x00B1
FIELD_WR= CAM_AWB_CCM_R_BG_GAIN, 0x0099
FIELD_WR= CAM_AWB_CCM_L_CTEMP, 0x09C4
FIELD_WR= CAM_AWB_CCM_M_CTEMP, 0x0D67
FIELD_WR= CAM_AWB_CCM_R_CTEMP, 0x1964
FIELD_WR= CAM_AWB_COLOR_TEMPERATURE_MIN, 0x09C4
FIELD_WR= CAM_AWB_COLOR_TEMPERATURE_MAX, 0x1964
FIELD_WR= CAM_AWB_X_SHIFT, 0x0023
FIELD_WR= CAM_AWB_Y_SHIFT, 0x0013
FIELD_WR= CAM_AWB_RECIP_X_SCALE, 0x006D
FIELD_WR= CAM_AWB_RECIP_Y_SCALE, 0x005F
FIELD_WR= CAM_AWB_ROT_CENTER_X, 0xFFF5
FIELD_WR= CAM_AWB_ROT_CENTER_Y, 0xFFDF
FIELD_WR= CAM_AWB_ROT_SIN, 0x0037
FIELD_WR= CAM_AWB_ROT_COS, 0x0021
FIELD_WR= CAM_AWB_WEIGHT_TABLE_0, 0x0000
FIELD_WR= CAM_AWB_WEIGHT_TABLE_1, 0x0011
FIELD_WR= CAM_AWB_WEIGHT_TABLE_2, 0x1001
FIELD_WR= CAM_AWB_WEIGHT_TABLE_3, 0x1000
FIELD_WR= CAM_AWB_WEIGHT_TABLE_4, 0x0000
FIELD_WR= CAM_AWB_WEIGHT_TABLE_5, 0x0177
FIELD_WR= CAM_AWB_WEIGHT_TABLE_6, 0x7227
FIELD_WR= CAM_AWB_WEIGHT_TABLE_7, 0x7100
FIELD_WR= CAM_AWB_WEIGHT_TABLE_8, 0x0111
FIELD_WR= CAM_AWB_WEIGHT_TABLE_9, 0x1377
```

Test Cases

1. Gretag Macbeth Color Checker chart illuminated with a light box under the following color temperatures with a Neutral background:
   - A light (tungsten ~ 2800 K)
   - D65 (daylight ~6500 K)
   - CWF (cool white fluorescent ~4200 K)
2. AWB failure tests: Lab Scene: Single color card of the following colors are used under the above conditions to test for color tracking (or Background immunity)
   - Neutral (nominal condition)
   - Sky Blue
   - Bright Yellow
   - Green
   - Red
   - Orange
   - Browns
3. AWB failure tests: Natural Scene
   - Tree Shaded Portrait
   - Blue sky dominant scene
   - Grass dominant scene
   - Indoor Office Lighting with Cream/Yellow/Blue Walls
   - Skin tones

These are just suggestions and for each customer application there could be different criteria.

Lab Testing with Macbeth Chart in Light Box

The camera being tested should be placed in a light box (for example, Judge II) with Macbeth chart in the scene (when using a light box, ensure other light sources do not impact the lighting). Then the color temperature of the light box should be changed and the bottom row monitored to measure the AWB convergence. The aim is to ensure the red and blue channels are within a few percent of the green channel.

Figure 36 and Figure 37 show good and bad AWB convergence.

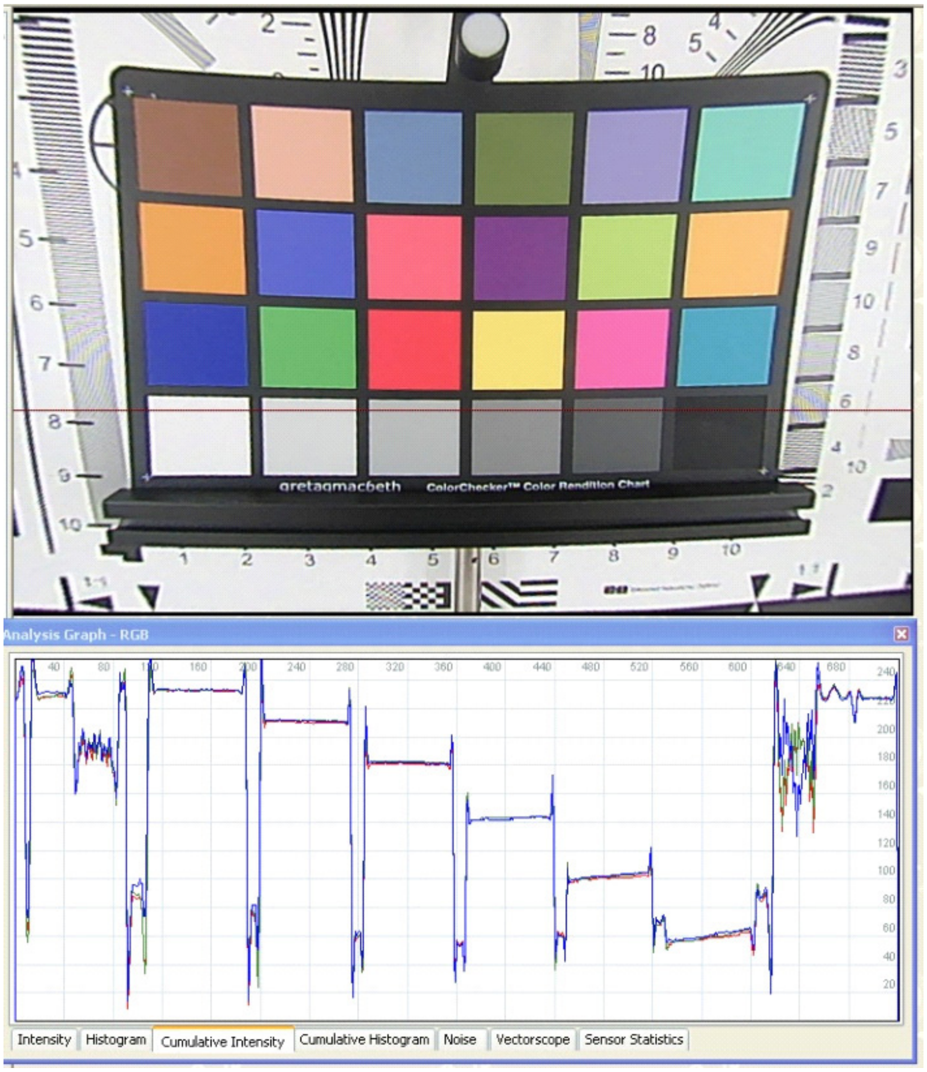


Figure 36. Good AWB Convergence

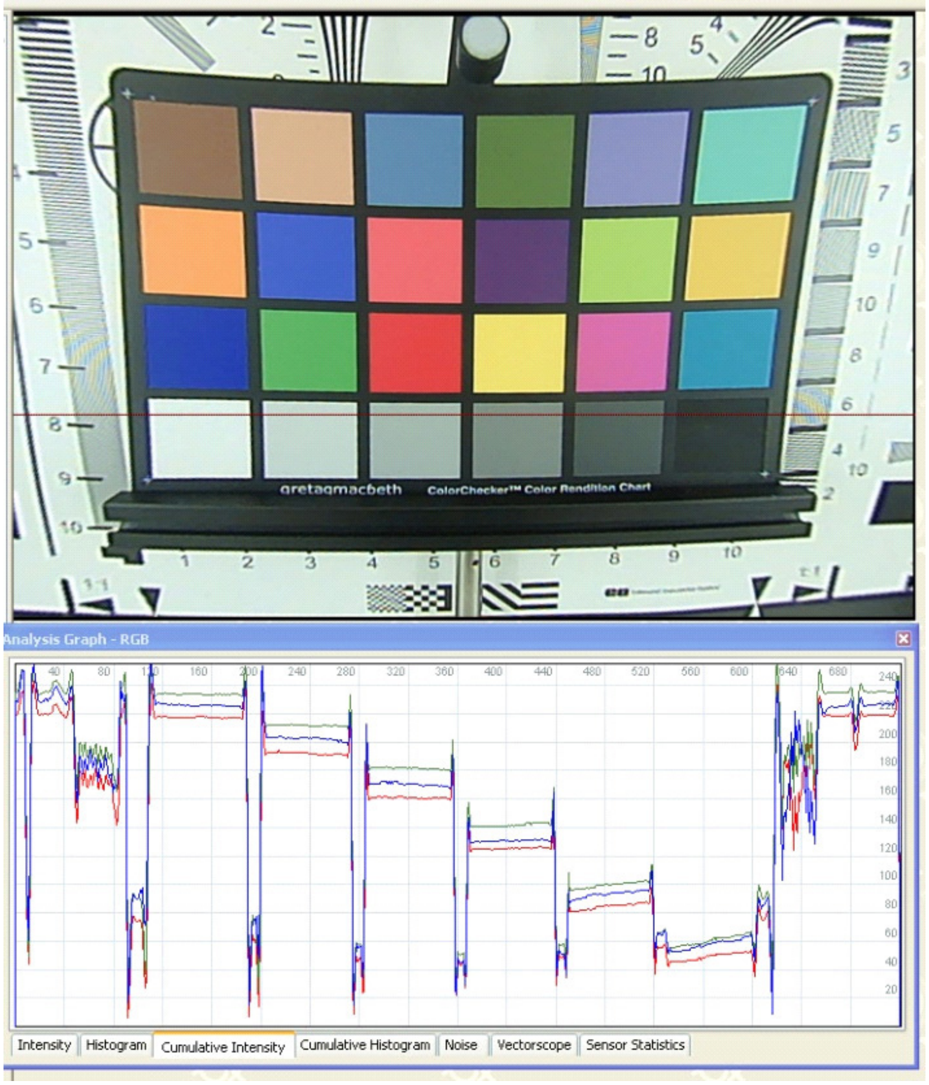


Figure 37. Bad AWB Convergence

How to Improve AWB Convergence

NOTE: This guide assumes that D65, CWF, and A light are being used. The customer may have some different light sources, but the principle is the same.

1. Setting of AWB gain ratios
   a. Turn on the D65 lights and monitor the awb_r_gain and awb_b_gain variables
   b. Write these into thecam_awb_ccm_r_rg_gain and cam_awb_ccm_r_bg_gain variables
   c. Check that cam_awb_ccm_r_ctemp is set to 6500
   d. Change to the CWF lights and monitor the awb_r_gain and awb_b_gain variables
   e. Write these into the cam_awb_ccm_m_rg_gain and cam_awb_ccm_m_bg_gain variables
   f. Check that cam_awb_ccm_r_ctemp is set to your CWF color temperature (for example, 4000)
   g. Change to the A lights and monitor the awb_r_gain and awb_b_gain variables
   h. Write these into the cam_awb_ccm_l_rg_gain and cam_awb_ccm_l_bg_gain variables
   i. Ensure cam_awb_ccm_r_ctemp is set to your A light color temperature (for example, 2700)

Non−Neutral Colors Impacting the White Balance Performance

If the neutrals aren't producing a white balanced response using the color checker chart, switch to using a pure gray checker chart to eliminate the possible influence of colored patches on the color checker.

Alternatively, obscure the colors on the color checker chart with a gray sheet.

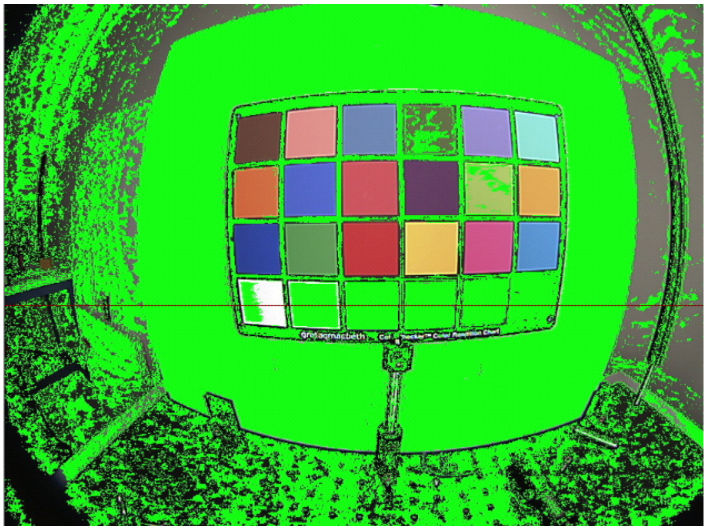
If the neutrals still aren't providing a good white balance then try small adjustments (increase/decrease by 1−5 decimal values) of the following variables (followed by a Refresh). Note that it may be necessary to pass an object such as your hand in between the camera and chart to trigger the AWB algorithm.

- cam_awb_x and y_shift
- cam_awb_rot_center_x and y

Weight Map Tuning (Lab)

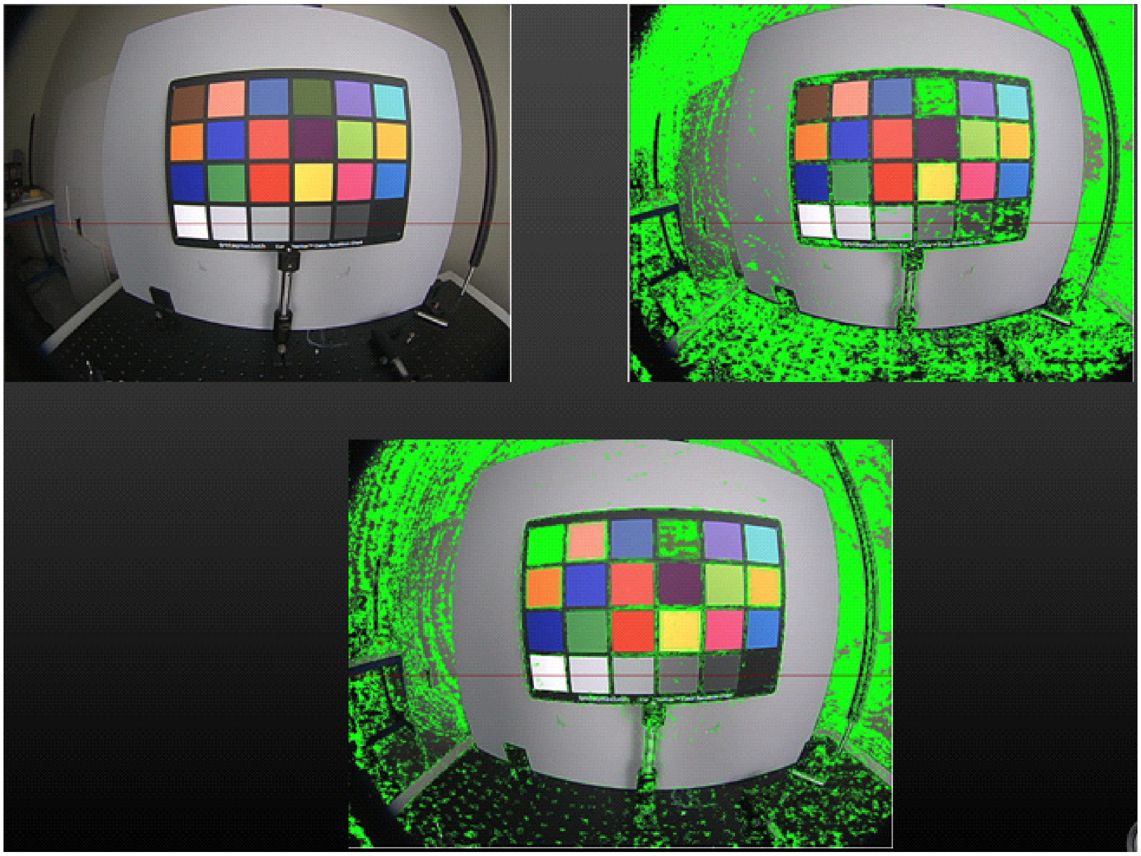
If the image is still not producing a white balanced image, then the AWB weight map will need to be tuned.

Ideally, all the neutrals are being highlighted.



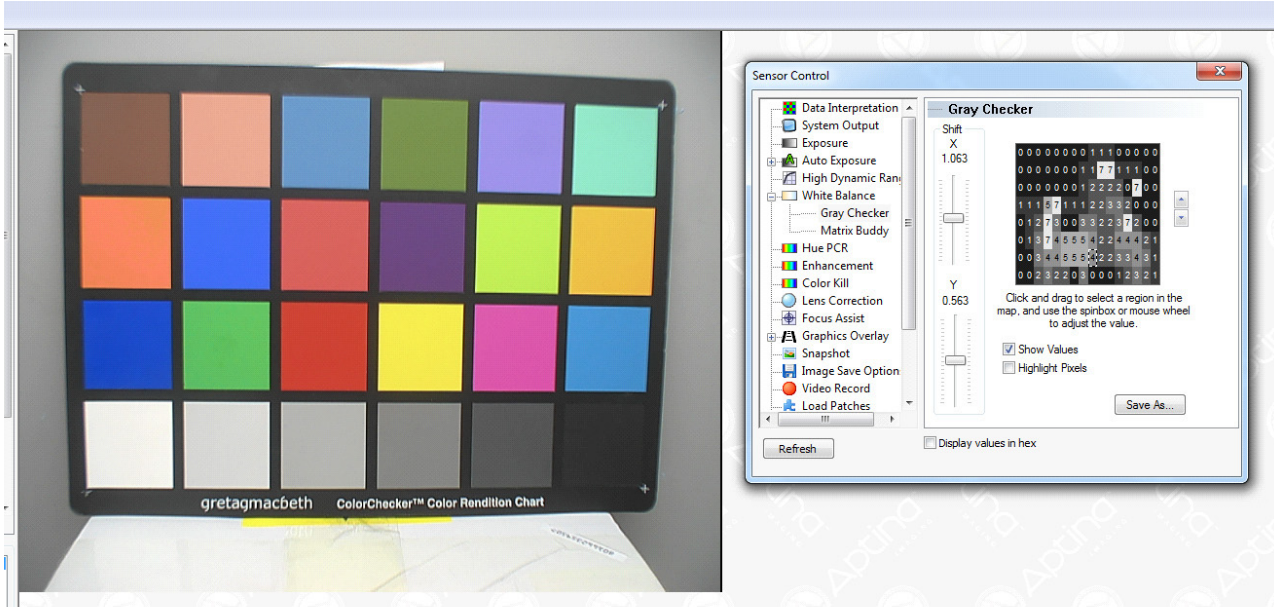
The AWB algorithm will not select clipped patches, hence on the above figure the white patch is not selected.

Few examples of bad weight tables are shown below, where either no neutrals being selected or a very small percentage of them that get selected.



The process for selecting a weight is described below.

1. Open the Gray Checker in the DevWare GUI:



X 1.063

Y 0.563

```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

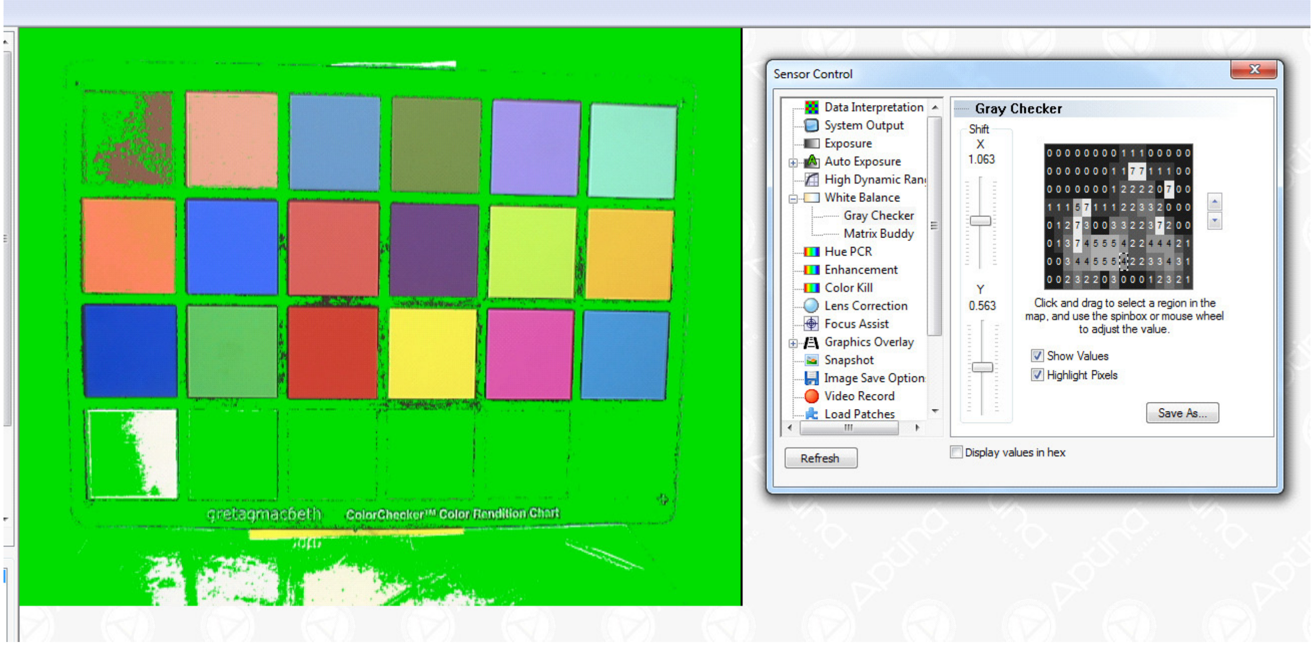
☑ Show Values

□ Highlight Pixels

□ Display values in hex

2. Tick *Highlight Pixels*. This will now show which pixels are being selected for AWB in this scene

This example is not bad, but let’s try and improve the top left patch.



Sensor Control

Gray Checker

Shift
X 1.063
Y 0.563

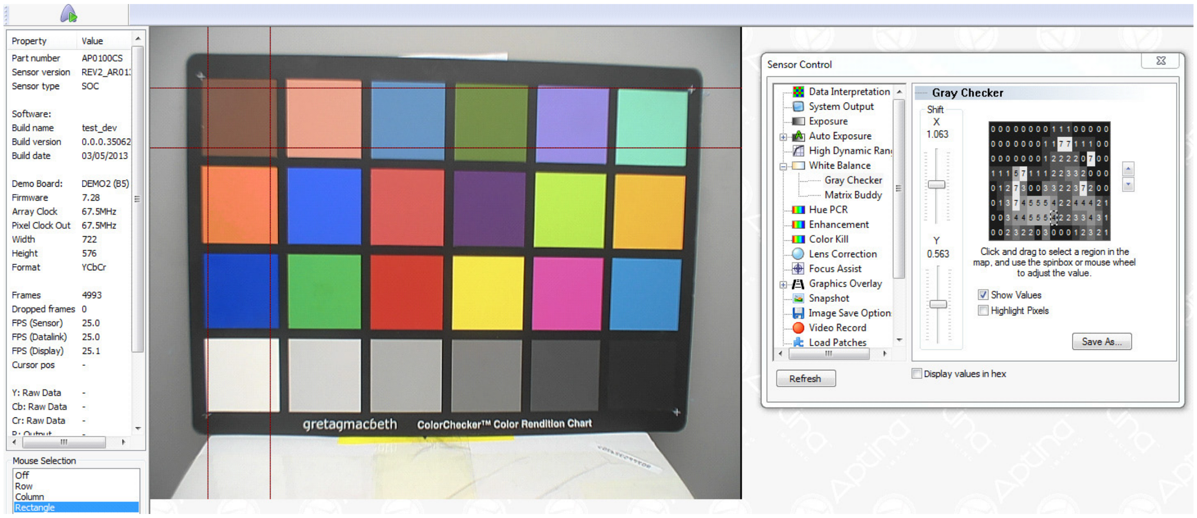
```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values
☑ Highlight Pixels

☐ Display values in hex

3. Uncheck *Highlight Pixels* and using the rectangle select the top left patch



Property	Value
Part number	AP0100CS
Sensor version	REV2_AR01
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.35062
Build date	03/05/2013
Demo Board:	DEMO2 (B5)
Firmware	7.28
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Frames	4993
Dropped frames	0
FPS (Sensor)	25.0
FPS (Datalink)	25.0
FPS (Display)	25.1
Cursor pos	-
Y: Raw Data	-
Cb: Raw Data	-
Cr: Raw Data	-

Mouse Selection
Off
Row
Column
Rectangle

Sensor Control

Gray Checker

Shift
X 1.063
Y 0.563

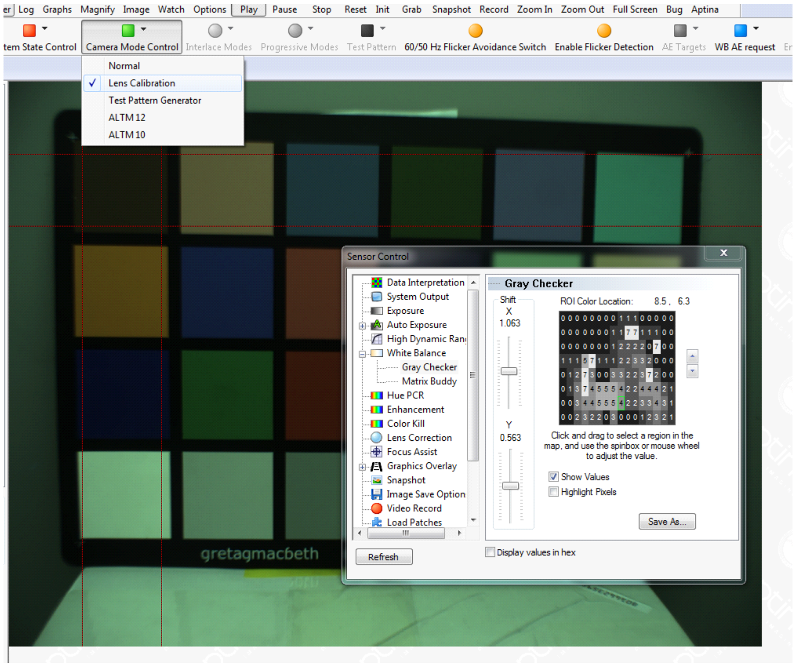
```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values
☐ Highlight Pixels

☐ Display values in hex

4. Select Lens Calibration mode, the camera should now be in Bayer.
   In the Gray Checker you should notice and X and Y coordinate and also in the map a green outline. This green outline tells the user which weight to tune



Shift X 1.063

Y 0.563

ROI Color Location: 8.5 , 6.3

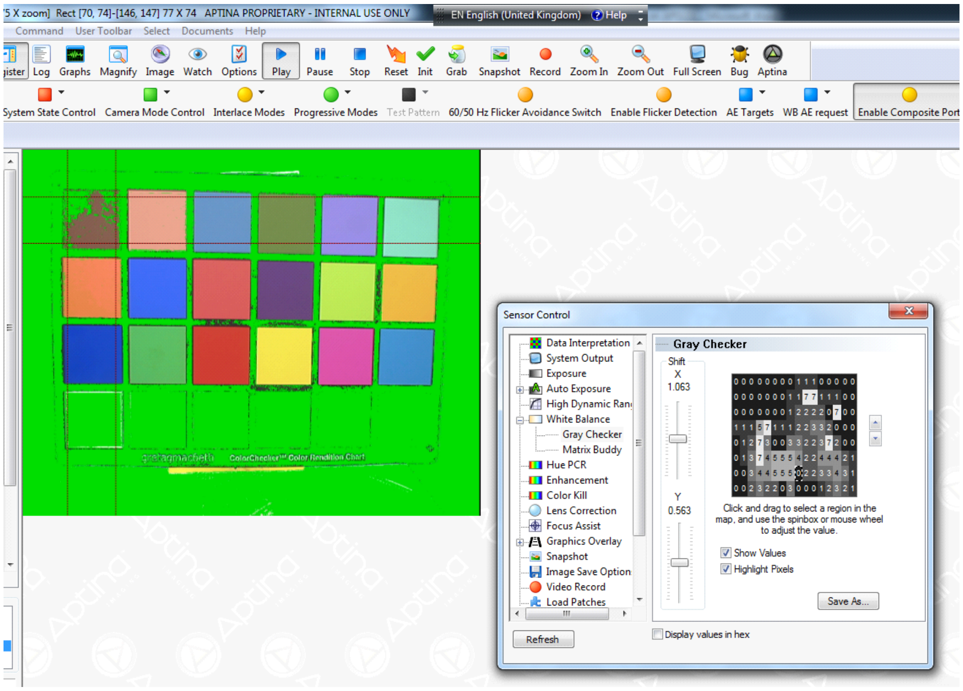
```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values
☐ Highlight Pixels

☐ Display values in hex

5. Switch back to normal mode and turn highlight pixels on. Now reduce the weight which was selected until it has an effect (this example only had a small effect)



Shift X 1.063

Y 0.563

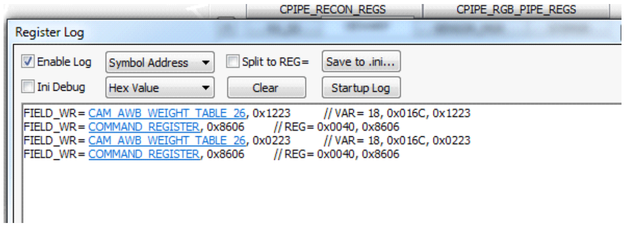
```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 0 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values
☑ Highlight Pixels

☐ Display values in hex

6. It is useful to have the log window open to keep a note of the weight changes



```
FIELD_WR= CAM_AWB_WEIGHT_TABLE_26, 0x1223      // VAR= 18, 0x016C, 0x1223
FIELD_WR= COMMAND_REGISTER, 0x8606      // REG= 0x0040, 0x8606
FIELD_WR= CAM_AWB_WEIGHT_TABLE_26, 0x0223      // VAR= 18, 0x016C, 0x0223
FIELD_WR= COMMAND_REGISTER, 0x8606      // REG= 0x0040, 0x8606
```

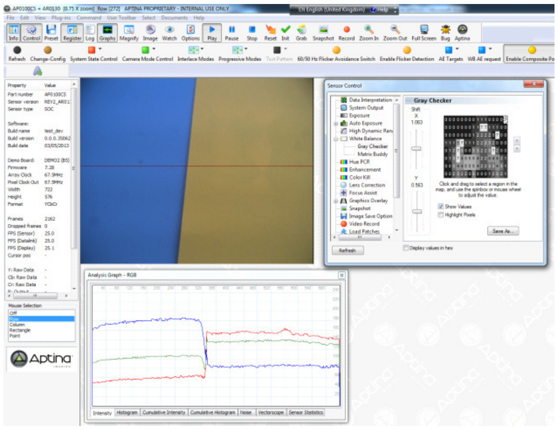
7. Repeat this process until the image provides acceptable AWB convergence.Once the part is correctly white balancing and color correcting in the lab, real scene and videos should be captured to verify the tuning

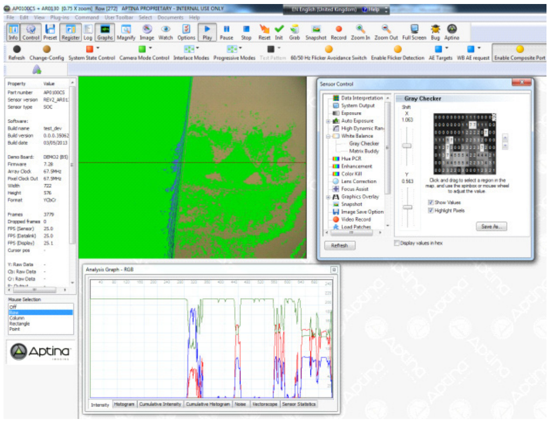
Lab Testing for Single Color Tracking (or Background Immunity)

This is considered as a stress test for the AWB; the scene should have a dominant color added to it to provoke failure cases.

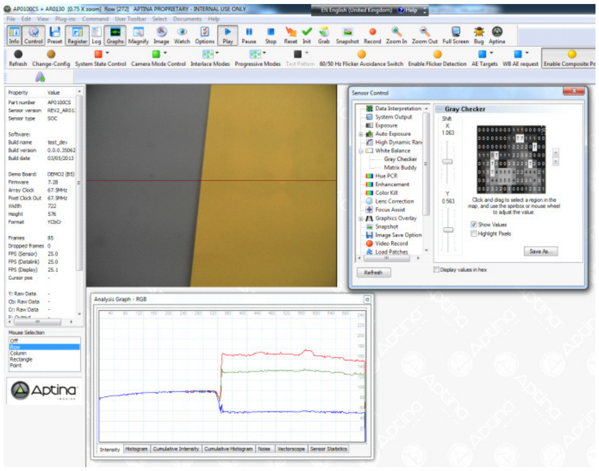
There are certain colors under specific illuminants that may appear as gray. Examples are yellow under D65 and blue under A light. Hence if there is yellow background, chances are that more of the yellow pixels get selected for white balance and we end up with severe color tracking issues.

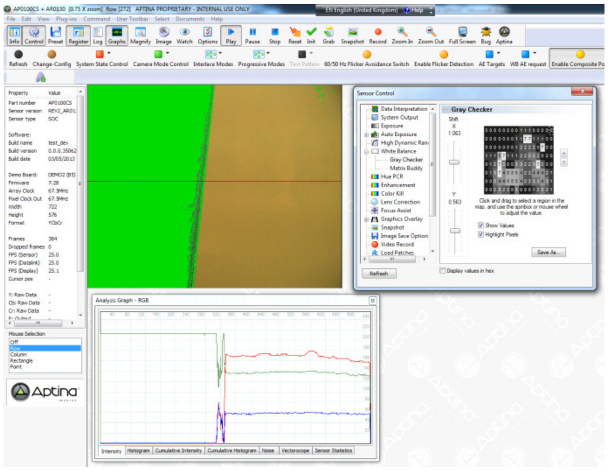
A yellow card is added to the scene, and the left hand side of the image (gray) turns blue. This is because the AWB is considering some of the yellow card for its AWB solution.





Using the same technique as in section “Weight map tuning (lab)” the weight is identified and changed so that it is no longer considered as part of the AWB solution.





This process will continue for all color charts that are to be tested and over the range of color temperatures. It is possible that a scenario will occur that one weight fixes something in one scene and looks not as good in another; in that scenario a compromise might be required.

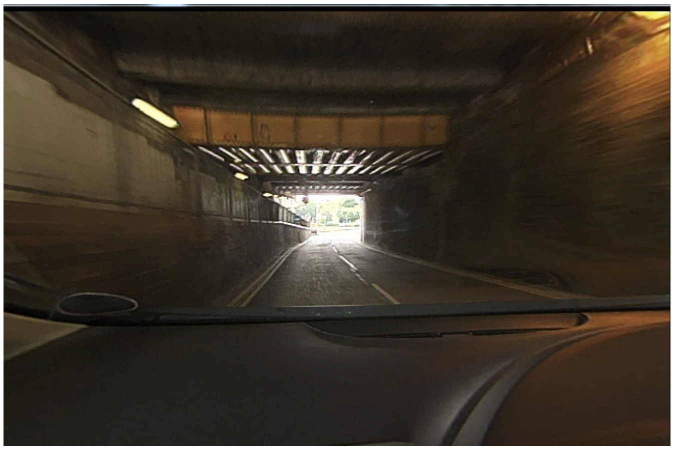
Real−Life Validation

Real−life validation is a critical part of testing the AWB solution. The real life scenes used will depend on the customer application but should include things like.

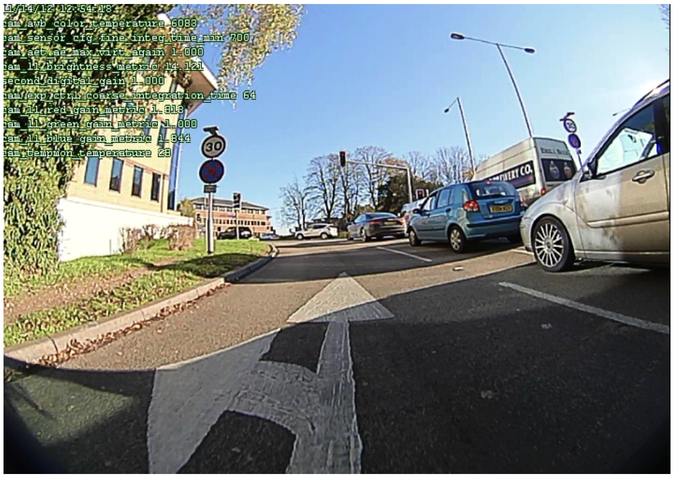
- Indoor mixed lighting

- Underground car park with mixed lighting

- Car driving through the tunnel during day

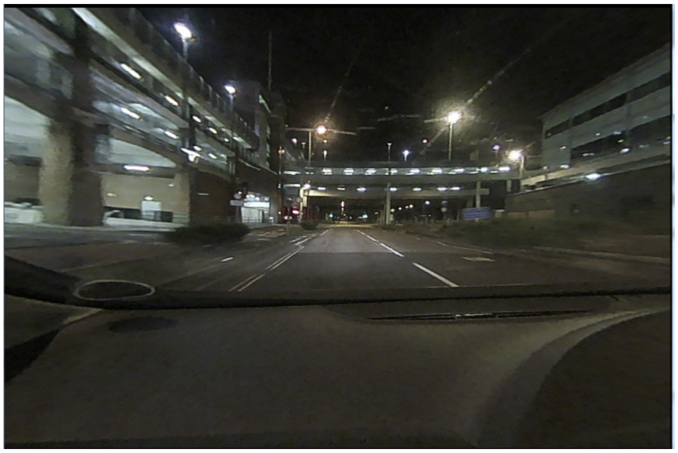


- Sunny outdoor car video



```
11/14/12 12:54:18
cam_awb_color_temperature 6083
cam_sensor_cfg_fine_integ_time_min 700
cam_aet_ae_max_virt_again 1.000
cam_ll_brightness_metric 14.121
second_digital_gain 1.000
cam_exp_ctrl_coarse_integration_time 64
cam_ll_red_gain_metric 1.813
cam_ll_green_gain_metric 1.000
cam_ll_blue_gain_metric 1.844
cam_tempmon_temperature 28
```

- Car driving in the city center with mixed lighting at night

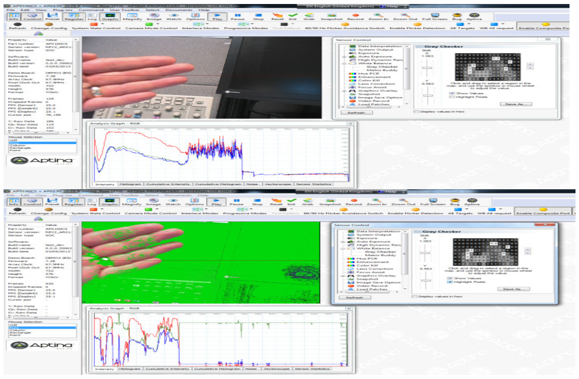


- Car driving through the tunnel at night

- Underground garage

- Skin tones are very important for AWB tuning

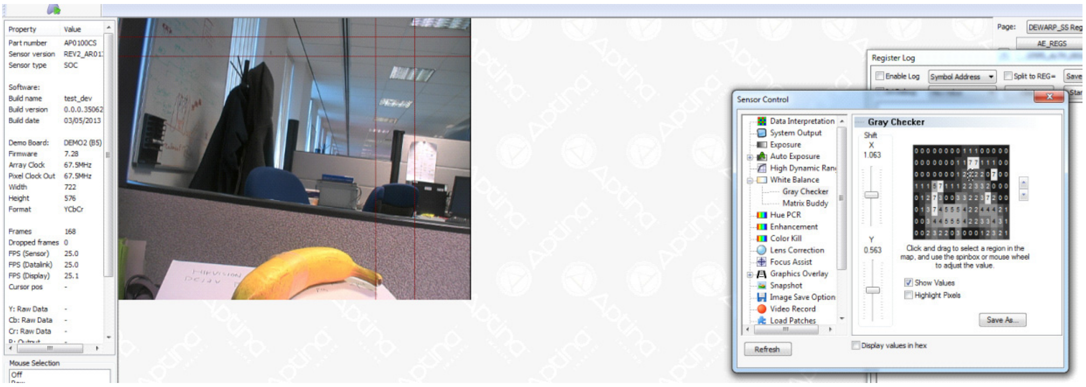
- When skin tone is in scene it should not be selected



Weight Map Tuning (Real Life)

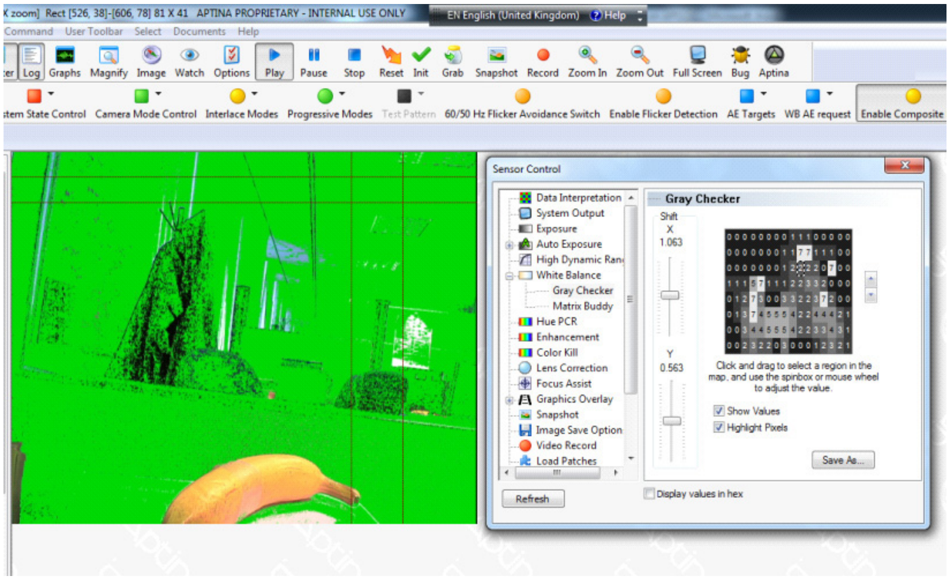
The principle of tuning the weight map for real life scenarios is the same as lab scenes.

In this example we will attempt to reduce the weight of the ceiling (this is just an exercise and not something that would normally be done).



Property	Value
Part number	AP0100CS
Sensor version	REV2_AR01
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.35062
Build date	03/05/2013
Demo Board:	DEMO2 (B5)
Firmware	7.28
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Frames	168
Dropped frames	0
FPS (Sensor)	25.0
FPS (Datalink)	25.0
FPS (Display)	25.1
Cursor pos	-
Y: Raw Data	-
Cb: Raw Data	-
Cr: Raw Data	-

1. Turn on highlights and add rectangle to the area



X zoom] Rect [526, 38]-[606, 78] 81 X 41 APTINA PROPRIETARY - INTERNAL USE ONLY

```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

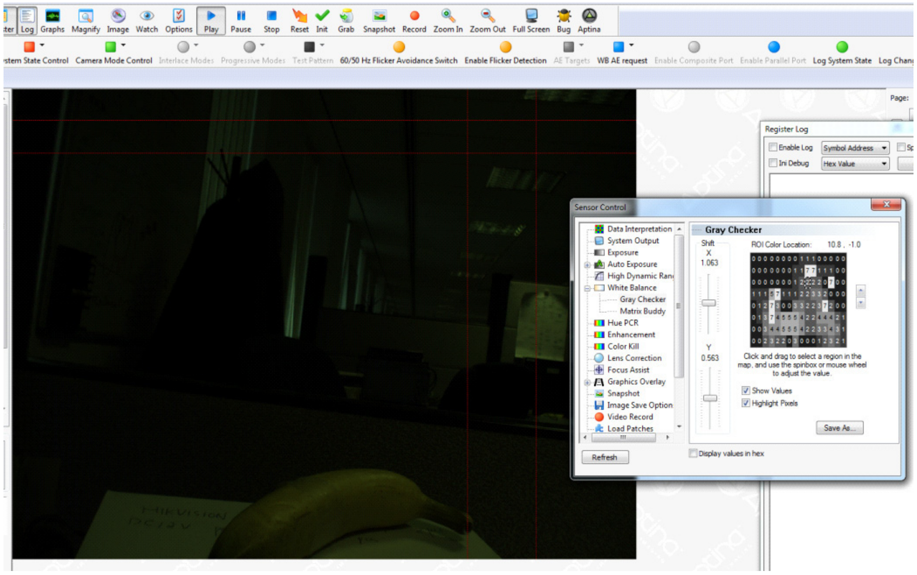
Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values

☑ Highlight Pixels

□ Display values in hex

2. Switch to Bayer, in this example nothing is being flagged up. So increase the exposure time (this may happen more in real scenes than the lab)



Register Log: □ Enable Log, □ Ini Debug

Gray Checker

Shift X 1.063, Y 0.563

ROI Color Location: 10.8 , -1.0

```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

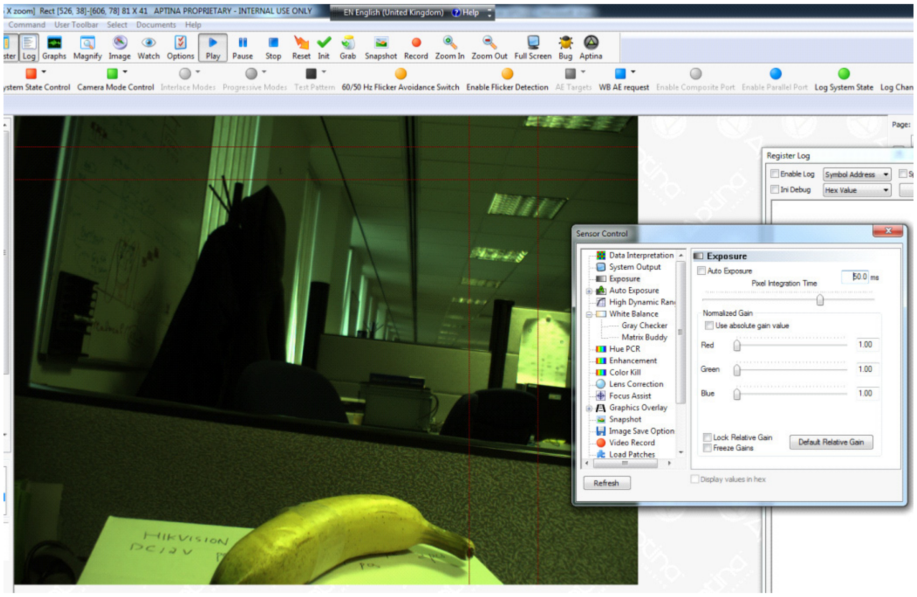
Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values

☑ Highlight Pixels

□ Display values in hex

3. Increase exposure time



Register Log: □ Enable Log, □ Ini Debug

Exposure

□ Auto Exposure

Pixel Integration Time 50.0 ms

Normalized Gain

□ Use absolute gain value

Red 1.00

Green 1.00

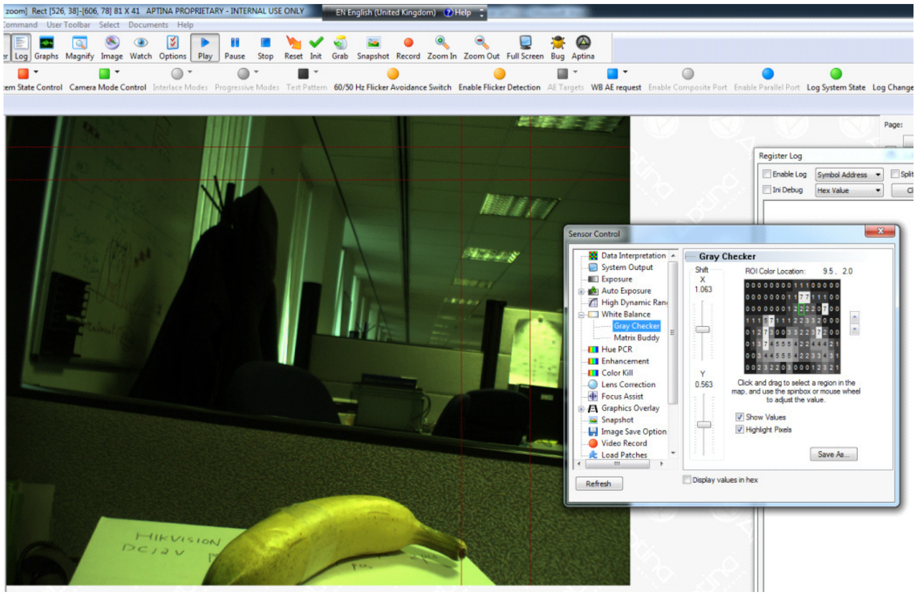
Blue 1.00

□ Lock Relative Gain

□ Freeze Gains

□ Display values in hex

With the exposure time increased a weight is highlighted.



Sensor Control

Gray Checker

Shift X 1.063 Y 0.563

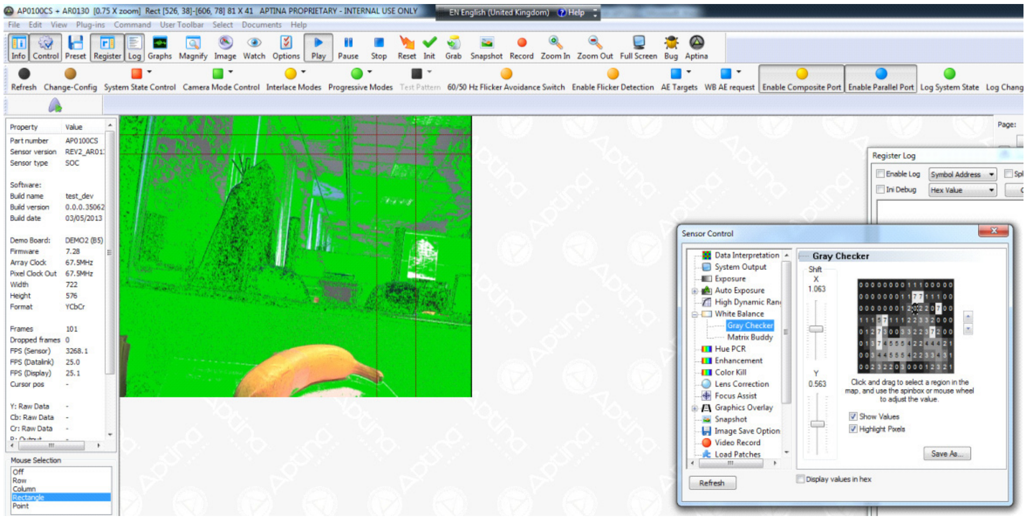
ROI Color Location: 9.5 , 2.0

```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 2 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values
☑ Highlight Pixels
☐ Display values in hex

4. Switch back to normal mode and reduce the weight, the ceiling is now being used less in AWB stats gathering



Property	Value
Part number	AP0100CS
Sensor version	REV2_AR01
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.35062
Build date	03/05/2013
Demo Board:	DEMO2 (B5)
Firmware	7.28
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Frames	101
Dropped frames	0
FPS (Sensor)	3268.1
FPS (Datalink)	25.0
FPS (Display)	25.1
Cursor pos	-
Y: Raw Data	-
Cb: Raw Data	-
Cr: Raw Data	-

Mouse Selection: Rectangle

Gray Checker

Shift X 1.063 Y 0.563

```
0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 1 1 7 7 1 1 1 0 0
0 0 0 0 0 0 0 1 2 0 2 2 0 7 0 0
1 1 1 5 7 1 1 1 2 2 3 3 2 0 0 0
0 1 2 7 3 0 0 3 3 2 2 3 7 2 0 0
0 1 3 7 4 5 5 5 4 2 2 4 4 4 2 1
0 0 3 4 4 5 5 5 4 2 2 3 3 4 3 1
0 0 2 3 2 2 0 3 0 0 0 1 2 3 2 1
```

Click and drag to select a region in the map, and use the spinbox or mouse wheel to adjust the value.

☑ Show Values
☑ Highlight Pixels
☐ Display values in hex

After all the real life tuning, repeat “Lab Testing with Macbeth Chart in Light Box” on page 93; it might be the case that the lab tuning is not perfect. This could be the compromise that is required to be made.

Manual Adjustment to CCM Colors

If any customer prefers different colors performance as per their application, CCM could be manually adjusted to some extent. This function is recommended for advanced user only.

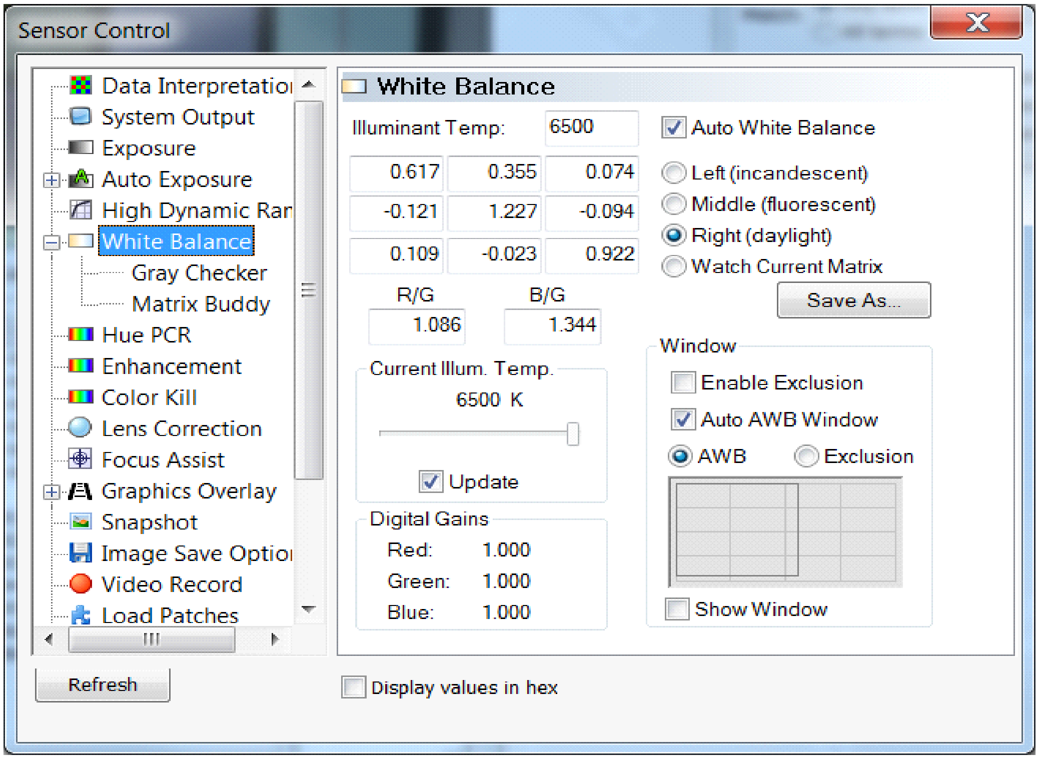
Manual Tuning

1. After tuning for best white balance performance apply the CCM derived from sensor tune and check its performance as per user preference
2. If changes required, power up the sensor and turn off the color kill, hue and PCR. Enable CCM

Table 35. VARIABLES

Map Address	Name	Description
REG 0x3210[5]	5: en_ccm	0x01
REG 0x3210[9]	9: pcr_enable	0x00
REG 0x3210[10]	10: hue_enable	0x00
VAR 0xCA54	cam_ll_ck_control	0x0000

3. Use the Macbeth color checker with a solid gray background as the scene and illuminate it with D65 light or set the scene as per user requirement
4. Ensure the scene is correctly exposed
5. Using the DevWare White Balance control page adjust the matrix to achieve the preferred colors
6. This adjustment is possible on all there CCMs (D65 (Right), CWF (Middle), and A (Left)) independently. Images below show example of accessing and changing D65 CCM



White Balance

Illuminant Temp: 6500

0.617	0.355	0.074
-0.121	1.227	-0.094
0.109	-0.023	0.922

R/G 1.086 B/G 1.344

Current Illum. Temp. 6500 K

☑ Update

Digital Gains
Red: 1.000
Green: 1.000
Blue: 1.000

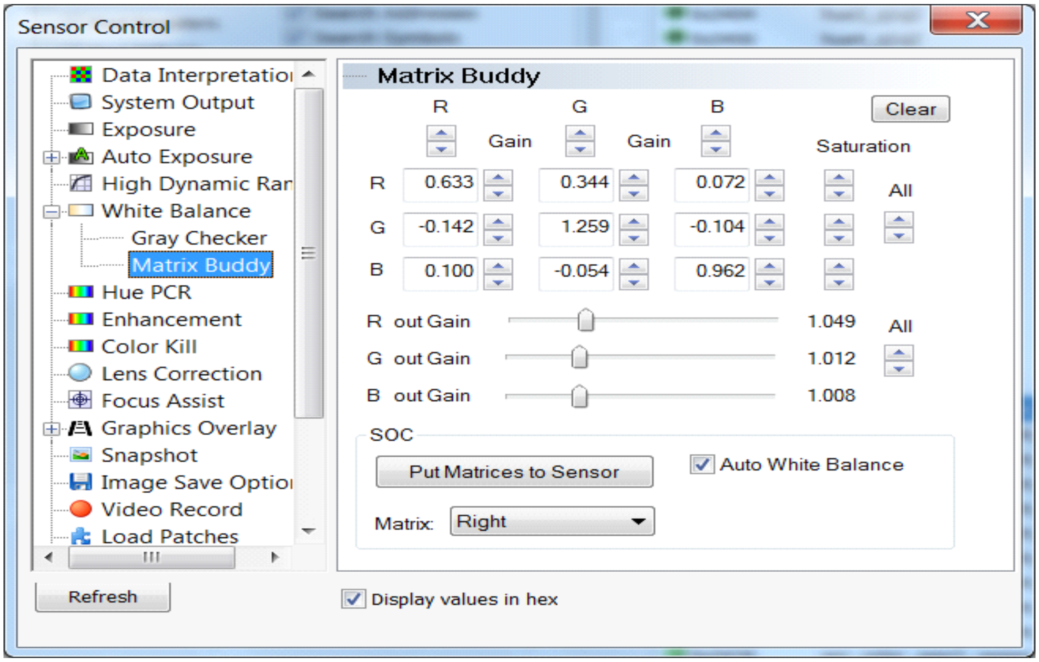
☑ Auto White Balance

○ Left (incandescent)
○ Middle (fluorescent)
● Right (daylight)
○ Watch Current Matrix

Window
☐ Enable Exclusion
☑ Auto AWB Window
● AWB ○ Exclusion
☐ Show Window

☐ Display values in hex

7. Select correct CCM option from ‘White Balance’ window



Matrix Buddy

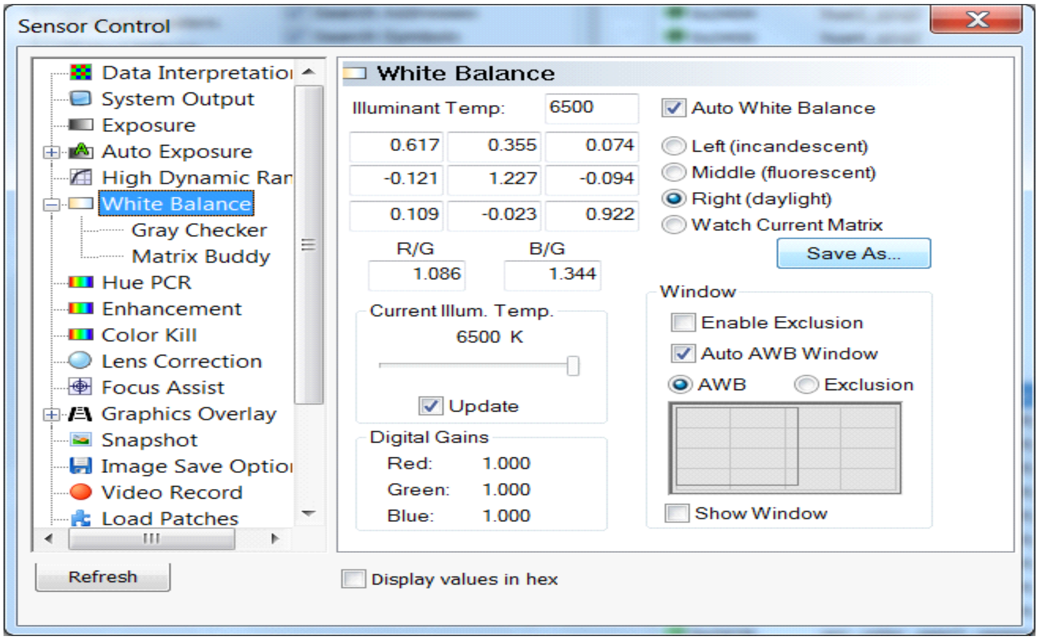
	R	G	B
R	0.633	0.344	0.072
G	-0.142	1.259	-0.104
B	0.100	-0.054	0.962

R out Gain 1.049
G out Gain 1.012
B out Gain 1.008

SOC
☑ Auto White Balance
Matrix: Right

☑ Display values in hex

8. Select Right(D65) CCM from ’Matrix’ option from ‘Matrix Buddy’ window
9. The columns of the matrix are the RGB channels and the rows are the color components within each channel. The rows should add up 1
10. Adjust the colors as required which enable ‘Put Matrices to Sensor’ option to use new CCM values in the sensor



White Balance

Illuminant Temp: 6500

0.617	0.355	0.074
-0.121	1.227	-0.094
0.109	-0.023	0.922

R/G 1.086
B/G 1.344

Current Illum. Temp.
6500 K
☑ Update

Digital Gains
Red: 1.000
Green: 1.000
Blue: 1.000

☑ Auto White Balance
○ Left (incandescent)
○ Middle (fluorescent)
● Right (daylight)
○ Watch Current Matrix

Window
☐ Enable Exclusion
☑ Auto AWB Window
● AWB ○ Exclusion
☐ Show Window

☐ Display values in hex

11. Once satisfied with the colors, save INI file from option ‘Save As’
12. Verify new CCM with range of different scenes to make sure manual adjustment in one color might have negative effect on other colors
13. Same procedure applies to CWF and A light CCM matrices

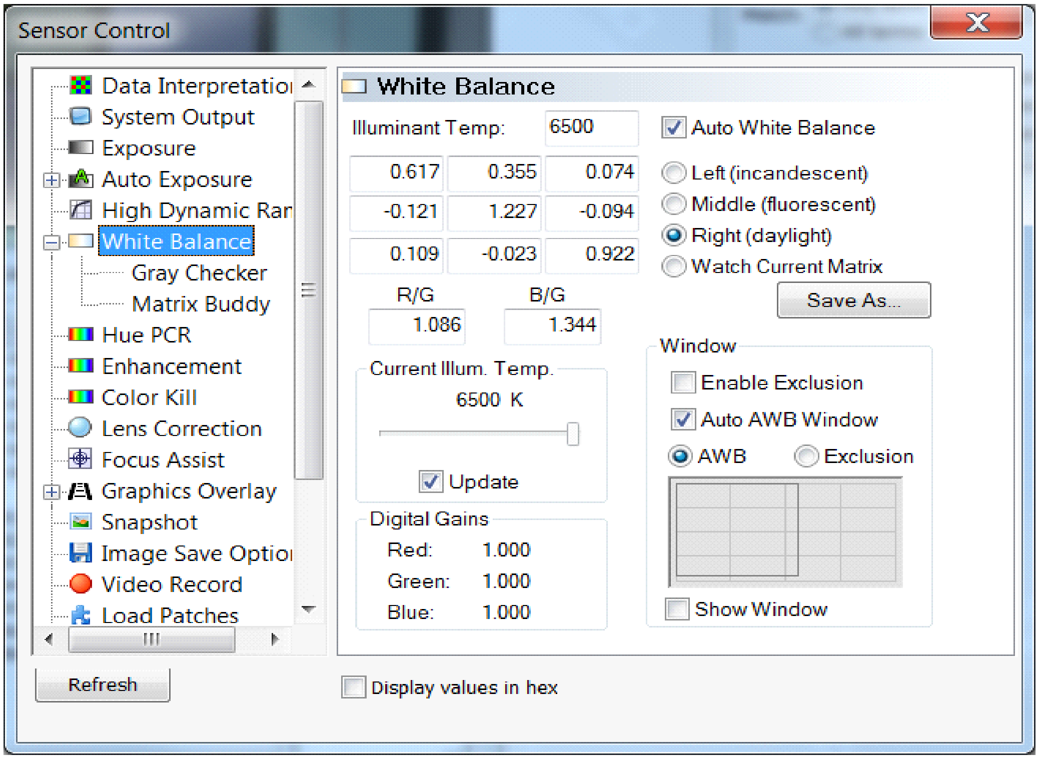
Manual Tuning

1. After tuning for best white balance performance apply the CCM derived from sensor tune and check its performance as per user preference
2. If changes required, power up the sensor and turn off the color kill, hue and PCR. Enable CCM

Table 36. VARIABLES

Map Address	Name	Description
REG 0x3210[5]	5: en_ccm	0x01
REG 0x3210[9]	9: pcr_enable	0x00
REG 0x3210[10]	10: hue_enable	0x00
VAR 0xCA54	cam_ll_ck_control	0x0000

3. Use the Macbeth color checker with a solid gray background as the scene and illuminate it with D65 light or set the scene as per user requirement.
4. Ensure the scene is correctly exposed
5. Using the DevWare White Balance control page adjust the matrix to achieve the preferred colors
6. This adjustment is possible on all there CCMs (D65 (Right), CWF (Middle), and A (Left)) independently. Images below show example of accessing and changing D65 CCM



White Balance

Illuminant Temp: 6500

0.617	0.355	0.074
-0.121	1.227	-0.094
0.109	-0.023	0.922

R/G 1.086 B/G 1.344

Current Illum. Temp. 6500 K

☑ Update

Digital Gains
Red: 1.000
Green: 1.000
Blue: 1.000

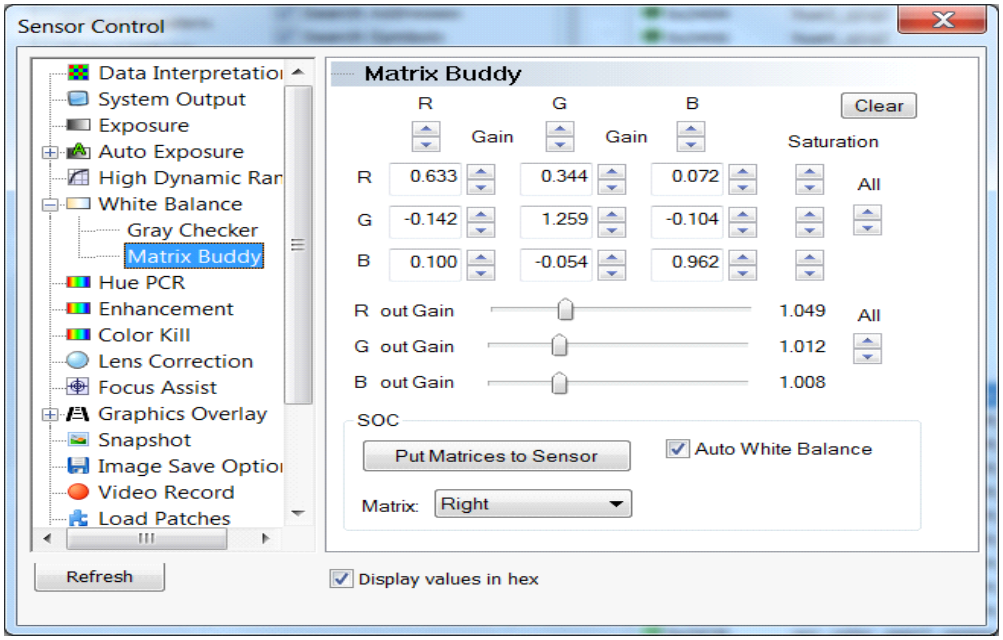
☑ Auto White Balance

○ Left (incandescent)
○ Middle (fluorescent)
● Right (daylight)
○ Watch Current Matrix

Window
□ Enable Exclusion
☑ Auto AWB Window
● AWB ○ Exclusion
□ Show Window

□ Display values in hex

7. Select correct CCM option from ‘White Balance ’window



	R	G	B
R	0.633	0.344	0.072
G	-0.142	1.259	-0.104
B	0.100	-0.054	0.962

R out Gain 1.049

G out Gain 1.012

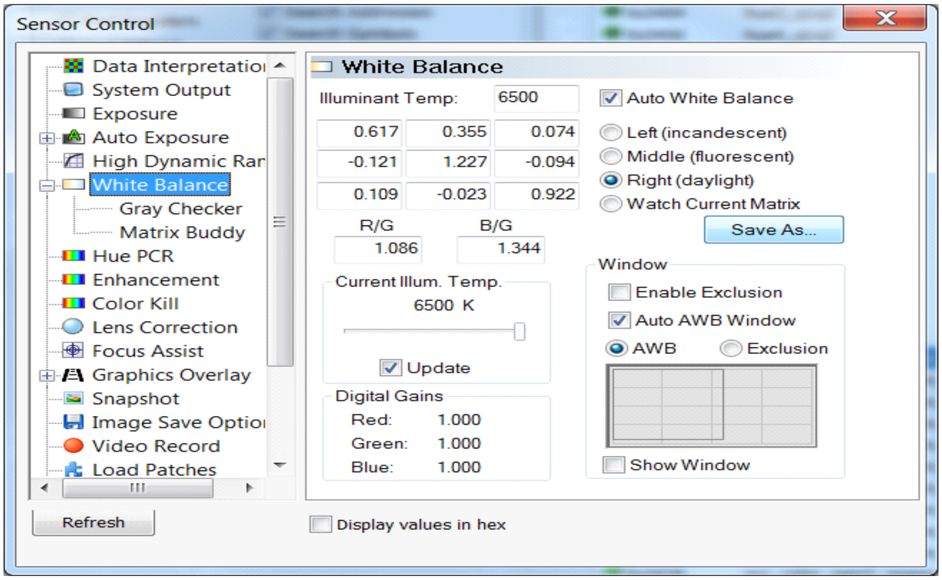
B out Gain 1.008

☑ Auto White Balance

Matrix: Right

☑ Display values in hex

8. Select Right(D65) CCM from ‘Matrix’ option from ‘Matrix Buddy’ window
9. The columns of the matrix are the RGB channels and the rows are the color components within each channel. The rows should add up 1
10. Adjust the colors as required which enable ‘Put Matrices to Sensor’ option to use new CCM values in the sensor



Illuminant Temp: 6500

0.617	0.355	0.074
-0.121	1.227	-0.094
0.109	-0.023	0.922

R/G	B/G
1.086	1.344

Current Illum. Temp.
6500 K

☑ Update

Digital Gains
Red: 1.000
Green: 1.000
Blue: 1.000

☑ Auto White Balance

○ Left (incandescent)
○ Middle (fluorescent)
● Right (daylight)
○ Watch Current Matrix

Window
☐ Enable Exclusion
☑ Auto AWB Window
● AWB ○ Exclusion
☐ Show Window

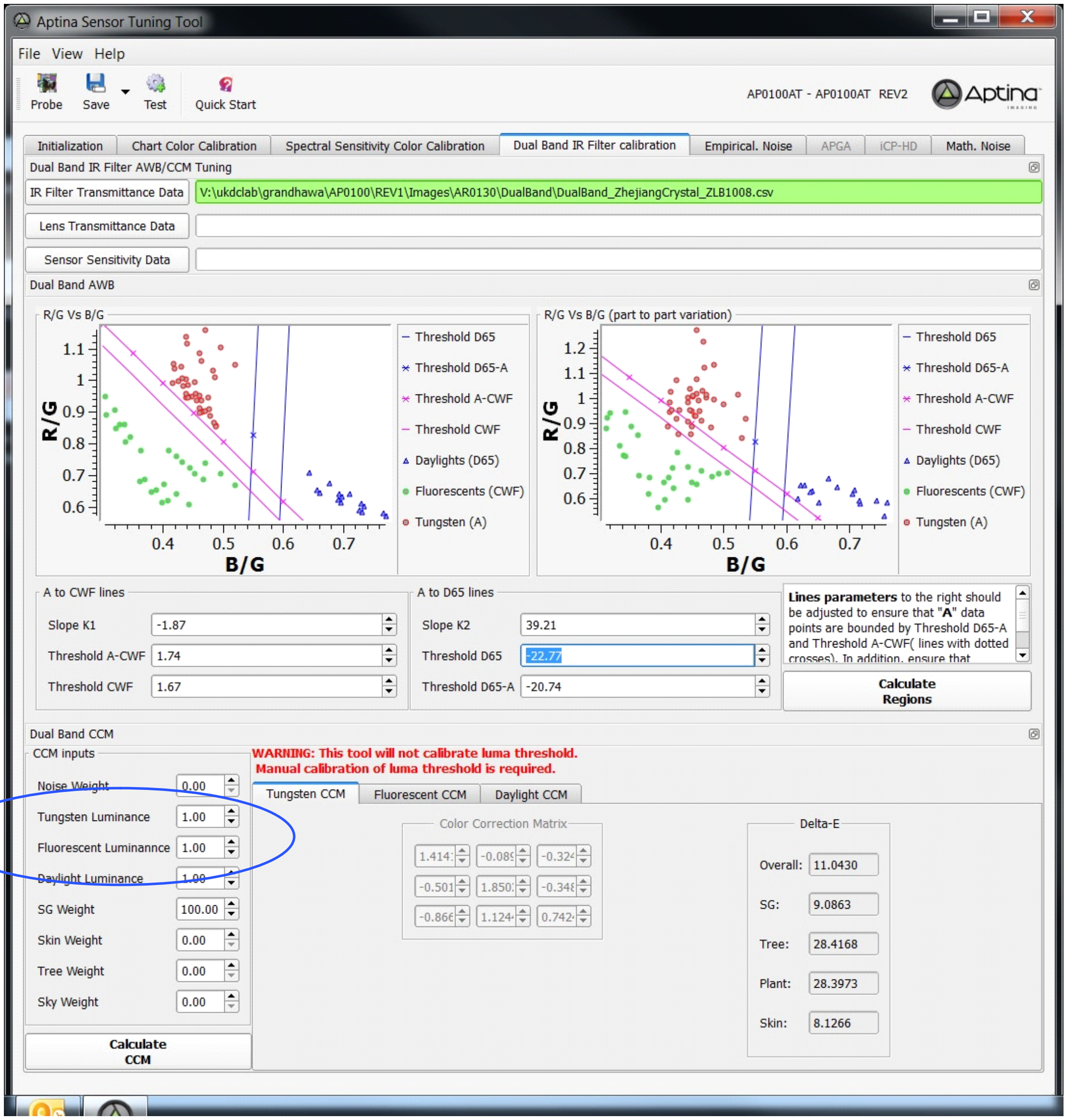
☐ Display values in hex

11. Once satisfied with the colors, save INI file from option ‘Save As’
12. Verify new CCM with range of different scenes to make sure manual adjustment in one color might have negative effect on other colors.

Same procedure applies to CWF and A light CCM matrices

DUAL BAND IRCF MANUAL TUNING

1. After using Sensor Tune if the results need further tuning then use this flow
2. Enable CCM_MODE CCM_DISABLE_NORM register
3. Check cam_awb_light_region variable changes with respect to illuminant changes (0 = A, 1 = CWF, 2 = D65)
4. If not, check saturation of CCM produces in D65, CWF, and A light. Color saturation is recommended to be around 100% or as preferred by the user
5. Adjust color saturation of D65, CWF, and A light CCMs by reducing luma threshold with the options ‘Daylight Luminance’; ‘Fluorescent Luminance’ and ‘Tungsten Luminance’ respectively. These options are highlighted with blue circle on image below. A lower value produces less saturated colors. For Example: A luma threshold value ‘1.00’ of ‘Daylight Luminance’ produces D65 CCM of 140% color saturation whereas reducing this value to ‘0.10’ produces 100% saturation for AP0102+AR0132 with Sunex dual band lens



IR Filter Transmittance Data: V:\ukdclab\grandhawa\AP0100\REV1\Images\AR0130\DualBand\DualBand_ZhejiangCrystal_ZLB1008.csv

A to CWF lines

Slope K1	-1.87
Threshold A-CWF	1.74
Threshold CWF	1.67

A to D65 lines

Slope K2	39.21
Threshold D65	-22.77
Threshold D65-A	-20.74

Lines parameters to the right should be adjusted to ensure that "**A**" data points are bounded by Threshold D65-A and Threshold A-CWF(lines with dotted crosses). In addition, ensure that

WARNING: This tool will not calibrate luma threshold. Manual calibration of luma threshold is required.

CCM inputs

Noise Weight	0.00
Tungsten Luminance	1.00
Fluorescent Luminannce	1.00
Daylight Luminance	1.00
SG Weight	100.00
Skin Weight	0.00
Tree Weight	0.00
Sky Weight	0.00

Color Correction Matrix

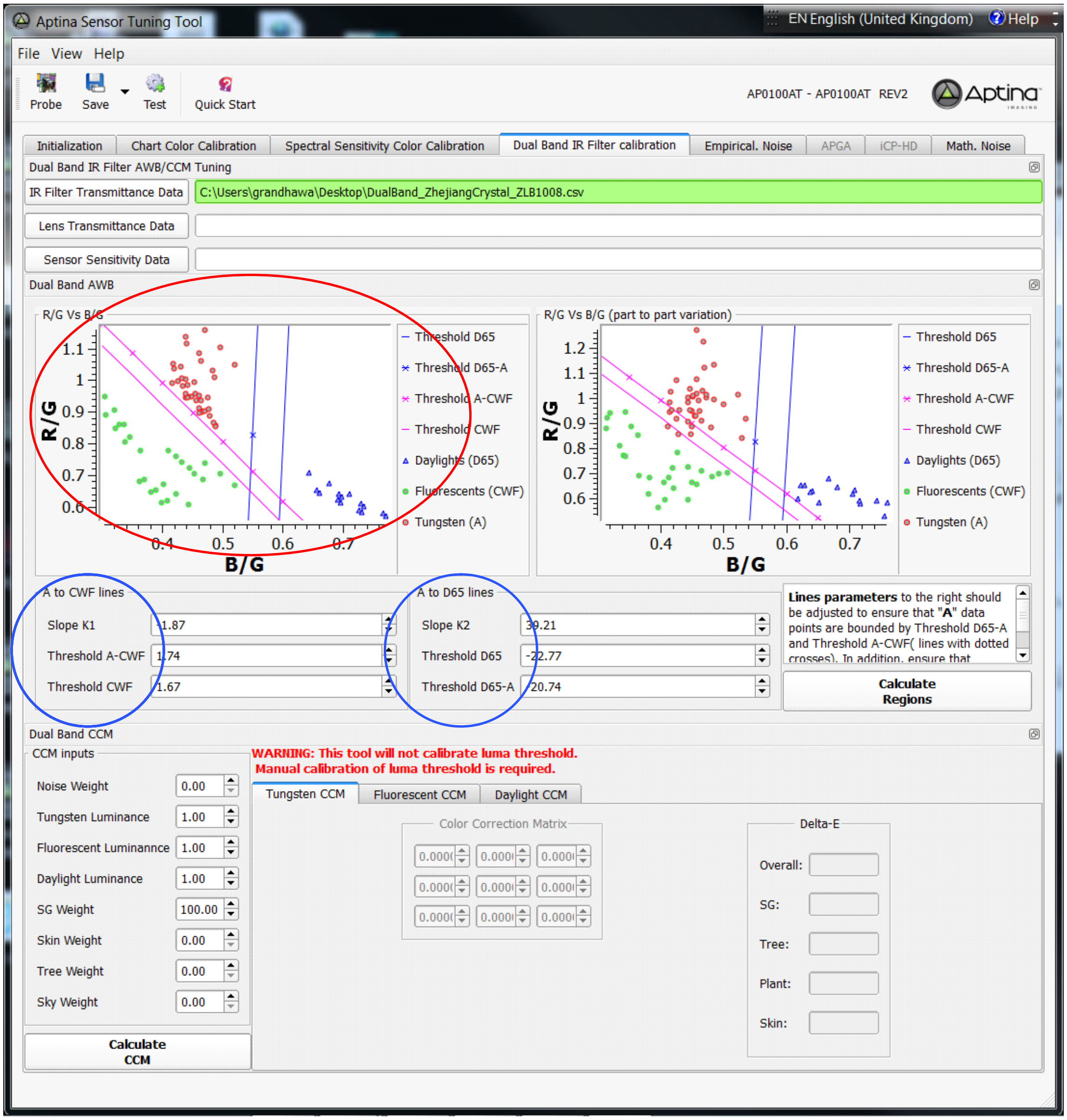
1.414	-0.089	-0.324
-0.501	1.850	-0.348
-0.866	1.124	0.742

Delta-E

Overall:	11.0430
SG:	9.0863
Tree:	28.4168
Plant:	28.3973
Skin:	8.1266

6. When user adjusts luma threshold values, new CCM parameters have to recalculate from option “Calculate CCM” and save as INI file to test on the sensor
7. Adjust awb_ir_control (0xAC28−0xAC32) parameters (blue circles) to separate three regions in R/G Vs B/G plot (highlighted with red circle in

image below) with enough transition regions to avoid oscillation



Dual Band IR Filter AWB/CCM Tuning

IR Filter Transmittance Data: C:\Users\grandhawa\Desktop\DualBand_ZhejiangCrystal_ZLB1008.csv

Lens Transmittance Data:

Sensor Sensitivity Data:

Dual Band AWB

A to CWF lines

Slope K1	1.87
Threshold A-CWF	1.74
Threshold CWF	1.67

A to D65 lines

Slope K2	39.21
Threshold D65	-22.77
Threshold D65-A	20.74

Lines parameters to the right should be adjusted to ensure that "**A**" data points are bounded by Threshold D65-A and Threshold A-CWF(lines with dotted crosses). In addition, ensure that

Dual Band CCM

CCM inputs

Noise Weight	0.00
Tungsten Luminance	1.00
Fluorescent Luminannce	1.00
Daylight Luminance	1.00
SG Weight	100.00
Skin Weight	0.00
Tree Weight	0.00
Sky Weight	0.00

WARNING: This tool will not calibrate luma threshold.
Manual calibration of luma threshold is required.

Color Correction Matrix

0.000(	0.000(	0.000(
0.000(	0.000(	0.000(
0.000(	0.000(	0.000(

Delta-E

Overall:

SG:

Tree:

Plant:

Skin:

NOTE: Description of parameters is as below:
THRESHOLD_1: interception of A−CWF boundary line
THRESHOLD_1_GATE: delta of interception of CWF to A−CWF boundary line
SLOPE_K1: it is slope of A−CWF boundary line
THRESHOLD_2: interception of D65−A boundary line
THRESHOLD_2_GATE: delta of interception of D65 to D65−A boundary line SLOPE_K2: it is slope of D65−A boundary line

8. R/G Vs B/G plot shows white point distributions at Fluorescent (region 1, with green points), Daylights (region 2, with blue points) and Tungsten (region 3, with red points)
9. User needs to manually select region parameters defined by two sets of boundary lines shown as magenta and cyan colors. A−CWF and CWF are corresponding to magenta lines to separate Fluorescent white point from Tungsten and Daylight regions. The D65−A and D65 (cyan) lines further separate white points into Tungsten and Daylight
10. Set Magenta and Cyan lines to separate three regions with enough transition regions to avoid oscillation
11. After adjustments, R/G Vs B/G plot must looks like as highlighted in red on image above
12. Save the INI file and run the test again to validate cam_awb_light_region variable changes with respect to illuminant changes (0 = A, 1 = CWF, 2 = D65)

Note that dual band tuning performance also depends on type of IRCF being used with sensors. Check with your ON Semiconductor contact if you have any question.

GRAPHICS OVERLAY

The AP0102 allows a user−defined bitmap image to be included as an overlay onto the video stream. Bitmap images must be run length encoded (RLE) in order to be used by the AP0102. There are 12 programmable layers. Each layer can be pointed to any of the 12 overlays (7 RLE, 1 Char, 2 Line Engines, & 2 Arc Engines).

RLE CODE

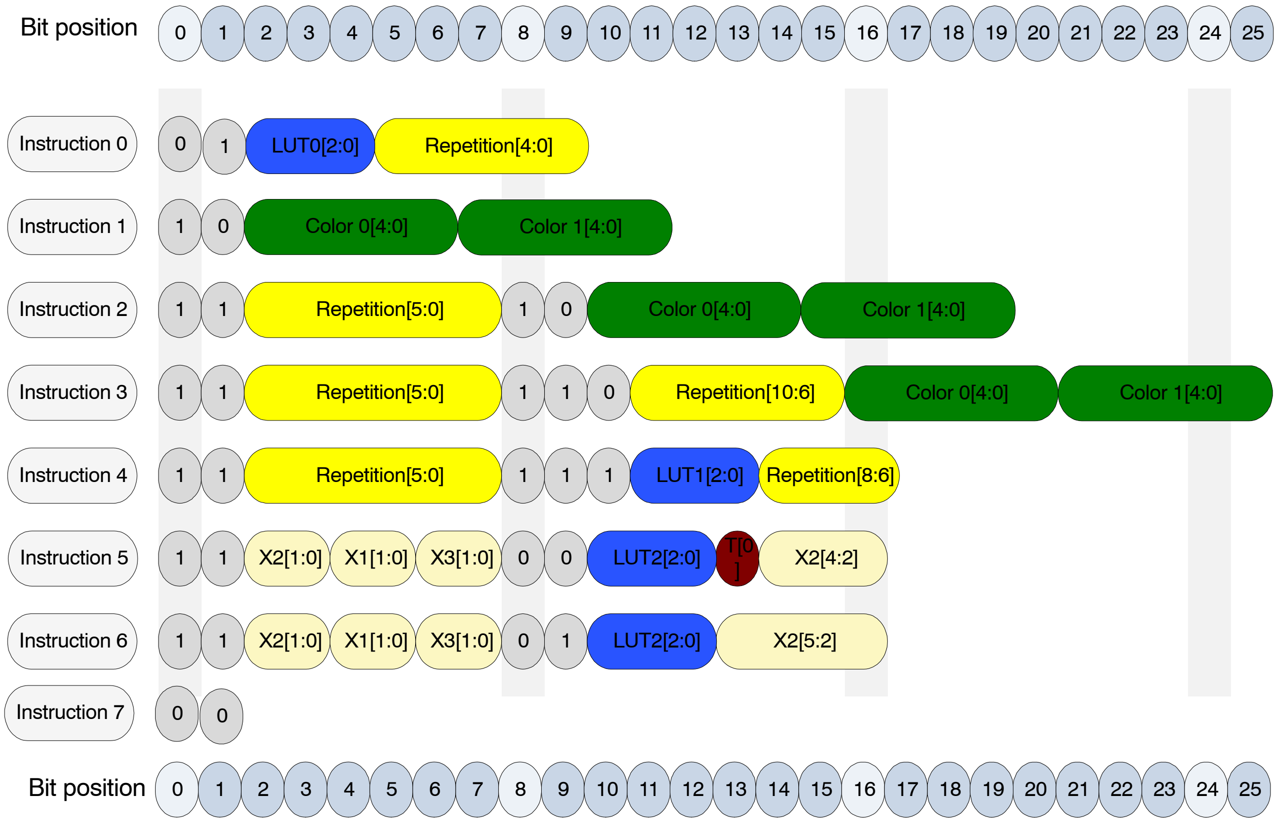
Run Length Encoding is used to provide an efficient means to store and decode images. The AP0102's instruction set is the same as that used on the AP0100 with the addition of extra color and LUT bits to support 32 colors (previously 16).

- The compressed image has a maximum length of 16 kb per buffer including 176 bytes for the look up tables. (128 bytes color, 16 bytes LUT0, 16 bytes LUT1, 16 bytes LUT2)
- Two lines (odd and even) are encoded together
- To save memory the instructions are bit aligned. Only the position of the first instruction is defined
- The encoding stops at the end of the line. Each line is encoded independently
- The color depth in a buffer is 5 bits (thirty two colors). However, the colors can be different between buffers. Effectively, an image can be generated with 192 different colors when using all six layers. Even more colors may appear when the blender combines all layers using transparency and there is overlapping of objects
- The contents of the image are translated to RLE instructions. The total number of instructions is eight (instruction 0 to 7) and the size of these instructions varies from 2 to 26 bits. Some of them are added to optimize the compression:
  - allow to use of look up tables to optimize space
  - allow the creation of common overlay shapes
- Three look−up tables can be used by the different instructions. These are located before the RLE code (48 bytes). These tables are independent between buffers and calculated when compressing the image:
  - The first look up table stores five combinations of colors that are repeated up to 32 times. This helps to reduce from 20 bits to 10 bits the area used to code the same information (i.e., instruction 2 is translated into instruction 0)
  - The second is similar to the first table but instead of less than 32 times, it's used for repetitions up to 512. In this case, 26 bits of information are compressed into 17 bits (i.e. instruction 3 is translated into instruction 4)
  - The third look up table is used for the shape optimizations. In this case, the Table contains color information and shape used. The optimization varies depending in the shape but can achieve the reduction from 10 bytes to 2 bytes. The shapes are intended to optimize Outline characters and traditional overlay figures (triangle, trapezoid, circle, …)
- A Break instruction indicates the end of the "active" RLE code

RLE INSTRUCTIONS

The AP0102AT RLE instructions are slightly modified versions of the RLE instructions used in AP0100. The modifications allow the use of 32 color tables (up from 16) (AP0100 used interlaced BT656 input data). The modifications are as follows:

1. There are now 32 color tables, so each color index is now 5 bits up from 4 bits in AP0100. This increases the length of instructions 1 and 2 by 2−bits each
2. Repetitions are not allowed to continue past the end of the line. This allows the RLE decoder to more easily "rewind" to the beginning of the line for the 2nd line of each 2− line pair. To simplify the HW and additional instruction "Instruction 7" was added. This also required Instruction 0 to increase by a single bit
3. Since the maximum repeat is now 2047, instruction 3 had the Repetition field size reduced to 11−bits. Since the color table indexes increased by 1−bit each, the overall length of instruction 2 remains unchanged

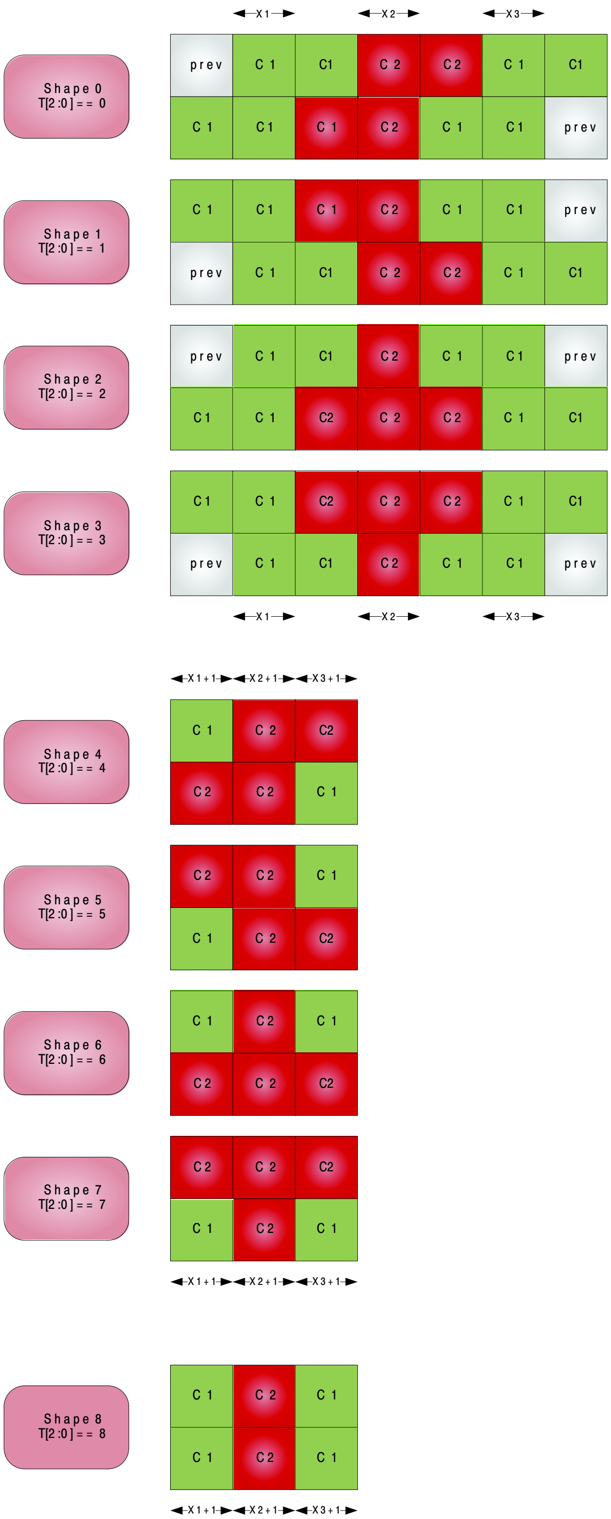

Bit position
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
Instruction 0
0 1 LUT0[2:0] Repetition[4:0]
Instruction 1
1 0 Color 0[4:0] Color 1[4:0]
Instruction 2
1 1 Repetition[5:0] 1 0 Color 0[4:0] Color 1[4:0]
Instruction 3
1 1 Repetition[5:0] 1 1 0 Repetition[10:6] Color 0[4:0] Color 1[4:0]
Instruction 4
1 1 Repetition[5:0] 1 1 1 LUT1[2:0] Repetition[8:6]
Instruction 5
1 1 X2[1:0] X1[1:0] X3[1:0] 0 0 LUT2[2:0] T[0] X2[4:2]
Instruction 6
1 1 X2[1:0] X1[1:0] X3[1:0] 0 1 LUT2[2:0] X2[5:2]
Instruction 7
0 0
Bit position
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

RLE Look Up Tables

LUT0 == 0	Color 0[4:0]	Color 1[4:0]	LUT1 == 0	Color 0[4:0]	Color 1[4:0]
LUT0 == 1	Color 0[4:0]	Color 1[4:0]	LUT1 == 1	Color 0[4:0]	Color 1[4:0]
LUT0 == 2	Color 0[4:0]	Color 1[4:0]	LUT1 == 2	Color 0[4:0]	Color 1[4:0]
LUT0 == 3	Color 0[4:0]	Color 1[4:0]	LUT1 == 3	Color 0[4:0]	Color 1[4:0]
LUT0 == 4	Color 0[4:0]	Color 1[4:0]	LUT1 == 4	Color 0[4:0]	Color 1[4:0]
LUT0 == 5	Color 0[4:0]	Color 1[4:0]	LUT1 == 5	Color 0[4:0]	Color 1[4:0]
LUT0 == 6	Color 0[4:0]	Color 1[4:0]	LUT1 == 6	Color 0[4:0]	Color 1[4:0]
LUT0 == 7	Color 0[4:0]	Color 1[4:0]	LUT1 == 7	Color 0[4:0]	Color 1[4:0]

LUT2 == 0	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 1	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 2	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 3	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 4	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 5	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 6	T[2:1]	C 1[4:0]	C 2[4:0]
LUT2 == 7	T[2:1]	C 1[4:0]	C 2[4:0]

RLE Shapes



Shape 0
T[2:0] == 0

	X1		X2		X3	
prev	C1	C1	C2	C2	C1	C1
C1	C1	C1	C2	C1	C1	prev

Shape 1
T[2:0] == 1

C1	C1	C1	C2	C1	C1	prev
prev	C1	C1	C2	C2	C1	C1

Shape 2
T[2:0] == 2

prev	C1	C1	C2	C1	C1	prev
C1	C1	C2	C2	C2	C1	C1

Shape 3
T[2:0] == 3

	X1		X2		X3	
C1	C1	C2	C2	C2	C1	C1
prev	C1	C1	C2	C1	C1	prev

Shape 4
T[2:0] == 4

X1+1	X2+1	X3+1
C1	C2	C2
C2	C2	C1

Shape 5
T[2:0] == 5

C2	C2	C1
C1	C2	C2

Shape 6
T[2:0] == 6

C1	C2	C1
C2	C2	C2

Shape 7
T[2:0] == 7

X1+1	X2+1	X3+1
C2	C2	C2
C1	C2	C1

Shape 8
T[2:0] == 8

X1+1	X2+1	X3+1
C1	C2	C1
C1	C2	C1

Table 37. INSTRUCTION DEFINITION

Instruction	Bits	Description
0	10	Extracts the pixel color information from look up table 0. This color information is repeated for the following pixels the amount of times as Repetition field +1 indicates. This instruction allows repetitions of up to 32 times
1	12	Extracts the pixel color information from the arguments. Note that there are no repetitions for this color
2	20	Extracts color information from the Color arguments. This color information is repeated for the following pixels the amount of times as Repetition field +1 indicates. This instruction allows repetitions of up to 64 times. Note: Special case if the Repetition field is zero then this is a Break instruction. See Instruction Break
3	26	Extracts the pixel color information from the Color arguments. This color information is repeated for the following pixels the amount of times as Repetition field +1 indicates. This instruction allows repetitions of up to 2048 times
4	17	Extracts the pixel color information from look up table 1. This color information is repeated for the following pixels the amount of times as Repetition field +1 indicates. This instruction allows repetitions of up to 512 times
5	17	Extracts the pixel color information from look up table 2. The T[2:0] bits (T[2:1] in the LUT2 and the T[0]) embedded in the instruction) describe one of the shape numbered from 0 to 7 in the previous figure. Note for Shape 0 to 3 there is a third color info that is extracted from the previous pixel. The X1,X2,X3 fields describe the number of repetitions for the different parts of the shape (Check Figure for more details)
6	17	Extracts the pixel color information from look up table 2. The T[2:1] bits in LUT2 are ignored and the shape used is always Shape 8. The X1, X2, X3 fields describe the number of repetitions for the different parts of the shape (Check Figure for more details). Note that this Shape x2 field has an extra bit compared to instruction 5
7	2	This instruction marks the end of a new line pair
Break	18	This instruction marks the end of the RLE code. See Instruction 2. It contains no image information because it should lie outside the image window

Example of Encoding

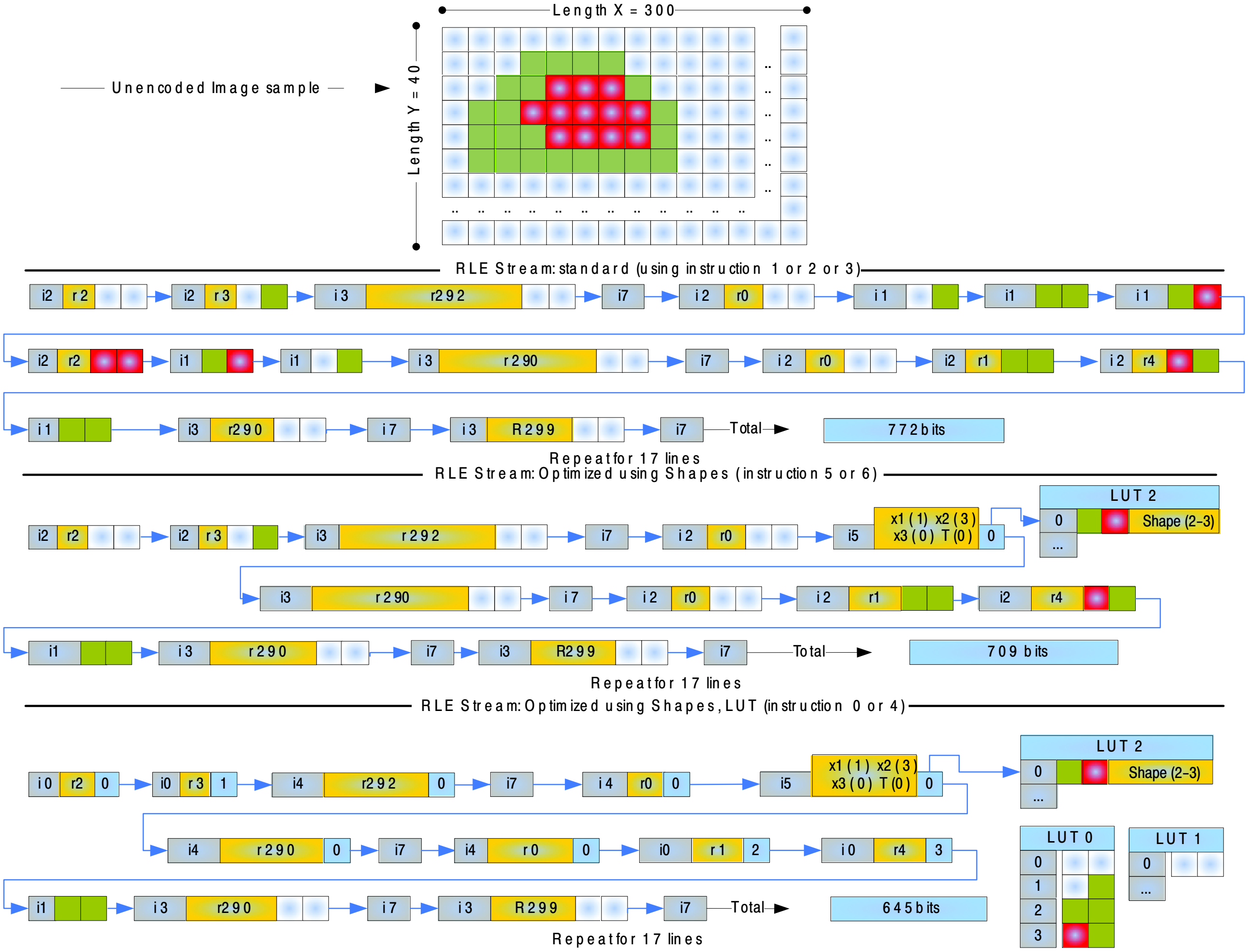


Figure 38.

OVERLAY RAM MEMORY MAP

The Overlay RAM memory map consists of seven RLE RAMs, two Line Descriptor RAMs, two Arc Descriptor RAMs, and Char/Font Generation RAMs.

RLE Layers Memory Map

Each of the RLE buffers has a header stored together with the RLE image data. This helps the DMA to stream the data continuously every time a buffer needs to be updated.

The Address mapping for the seven buffers is the are:

Table 38.

Name	Address
Buffer 0	0x0_0000
Buffer 1	0x0_5000
Buffer 2	0x0_A000
GAP	0x0_F000
Buffer 3	0x1_0000
Buffer 4	0x1_5000
Buffer 5	0x1_A000
GAP	0x1_F000
Buffer 6	0x2_0000

5. The RLE RAM interface can only be used during vertical blanking or when the layer is not enabled. Any accesses outside these time will be ignored but will generate a RAM access error.

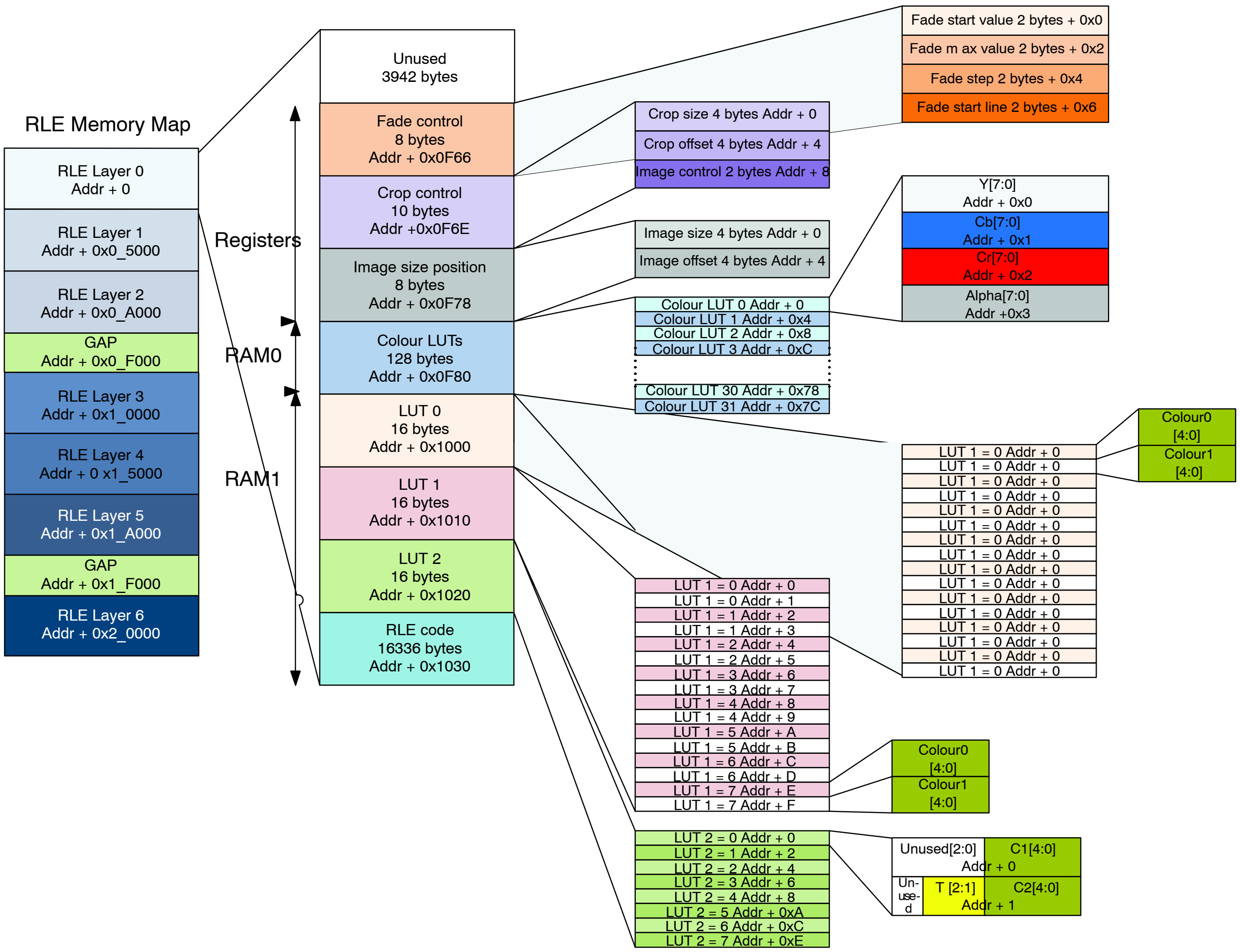


Figure 39. RLE Buffers Memory Mapping

Internally each of the buffers contains the following information:

Table 39.

Register Name	Description
Fader control	Fade for the overlay image
Length X,Y for crop window	Crop window horizontal and vertical size
X,Y Offset for crop window	Crop window horizontal/vertical starting point
Image Control	Image control bits
Length X,Y	Image horizontal and vertical size
X,Y Offset	Image horizontal/vertical starting point
LUT0 32 rep/color	Look up table for 32 repetitions
LUT1 512 rep/color	Look up table for 512 repetitions
LUT2 shape/color	Look up table for shapes
Color LUTs	Y/Cb/Cr/Alpha values for the 32 colors
RLE DATA	RLE Image Data

All the addresses shown below are relative to the Buffer addresses. For instance in case the user writes into RLE Buffer 2 Image control bits the Address would be 0x0_AF76 (0x0F76 from image control register + 0x0_A000 from Buffer 2).

Fader Control

Fader Start Value

Bit	Name	R/W	Address	Description
0	Fader start value high	R/W	0x0F66	Fader starting value
7:0	Fader start value low	R/W	0x0F67	Fader starting value

Fader Max Value

Bit	Name	R/W	Address	Description
0	Fader max value high	R/W	0x0F68	Fader max value
7:0	Fader max value low	R/W	0x0F69	Fader max value

Fader Step Value

Bit	Name	R/W	Address	Description
7:0	Fader step value high	R/W	0x0F6A	Fader step value (9.4 format)
7:3	Fader step value low	R/W	0x0F6B	Fader step value

Fader Start Line

Bit	Name	R/W	Address	Description
2:0	Fader start line high	R/W	0x0F6C	Fader start line
7:0	Fader start line low	R/W	0x0F6D	Fader start line

Length X,Y Crop Window

Bit	Name	R/W	Address	Description
2:0	Length X high	R/W	0x0F6E	Crop Horizontal Length
7:0	Length X low	R/W	0x0F6F	Crop Horizontal Length
2:0	Length Y high	R/W	0x0F70	Crop Vertical Length
7:0	Length Y low	R/W	0x0F71	Crop Vertical Length

X,Y Offset Start Crop Window

Bit	Name	R/W	Address	Description
2:0	X Offset high	R/W	0x0F72	Crop Horizontal Offset relative to overlay buffer
7:0	X Offset low	R/W	0x0F73	Crop Horizontal Offset relative to overlay buffer
2:0	Y Offset high	R/W	0x0F74	Crop Vertical Offset relative to overlay buffer
7:0	Y Offset low	R/W	0x0F75	Crop Vertical Offset relative to overlay buffer

Image Control

Bit	Name	R/W	Address	Description
0	Calibration enable	R/W	0x0F76	Image is using calibration registers
1	Crop enable	R/W		Enable cropping for this buffer inside the window specified in crop window offset and length
2	Crop out enable	R/W		0 – Crop inside the window 1 – Crop outside the window
3	Fader enable	R/W		Enable the fader.

Length X,Y Register

Bit	Name	R/W	Address	Description
2:0	Length X high	R/W	0x0F78	Image Horizontal Length
7:0	Length X low	R/W	0x0F79	Image Horizontal Length
2:0	Length Y high	R/W	0x0F7A	Image Vertical Length
7:0	Length Y low	R/W	0x0F7B	Image Vertical Length

X,Y Offset

Bit	Name	R/W	Address	Description
2:0	X Offset high	R/W	0x0F7C	Image Horizontal Offset
7:0	X Offset low	R/W	0x0F7D	Image Horizontal Offset
2:0	Y Offset high	R/W	0x0F7E	Image Vertical Offset
7:0	Y Offset low	R/W	0x0F7F	Image Vertical Offset

COLOR LUTS

Color0 LUT

Bit	Name	R/W	Address	Description
7:0	Color Y value	R/W	0x0F80	Color information for Luma
7:0	Color Cb value	R/W	0x0F81	Color information for Chroma
7:0	Color Cr value	R/W	0x0F82	Color information for Chroma
7:0	Color Alpha value	R/W	0x0F83	Transparency information

Color1 LUT

Bit	Name	R/W	Address	Description
7:0	Color Y value	R/W	0x0F84	Color information for Luma
7:0	Color Cb value	R/W	0x0F85	Color information for Chroma
7:0	Color Cr value	R/W	0x0F86	Color information for Chroma
7:0	Color Alpha value	R/W	0x0F87	Transparency information

6. Color 2 LUT − Color 30 LUT cover address range 0x0F88 − 0x0FFB.

Color31 LUT

Bit	Name	R/W	Address	Description
7:0	Color Y value	R/W	0x0FFC	Color information for Luma
7:0	Color Cb value	R/W	0x0FFD	Color information for Chroma
7:0	Color Cr value	R/W	0x0FFE	Color information for Chroma
7:0	Color Alpha value	R/W	0x0FFF	Transparency information

LUT0 32 REP/COLOR

Bit	Name	R/W	Address	Description
4:0	Color0	R/W	0x1000	Color 0 descriptor for slot 0
4:0	Color1	R/W	0x1001	Color 1 descriptor for slot 0
4:0	Color0	R/W	0x1002	Color 0 descriptor for slot 1
4:0	Color1	R/W	0x1003	Color 1 descriptor for slot 1
4:0	Color0	R/W	0x1004	Color 0 descriptor for slot 2
4:0	Color1	R/W	0x1005	Color 1 descriptor for slot 2
4:0	Color0	R/W	0x1006	Color 0 descriptor for slot 3
4:0	Color1	R/W	0x1007	Color 1 descriptor for slot 3
4:0	Color0	R/W	0x1008	Color 0 descriptor for slot 4
4:0	Color1	R/W	0x1009	Color 1 descriptor for slot 4
4:0	Color0	R/W	0x100A	Color 0 descriptor for slot 5
4:0	Color1	R/W	0x100B	Color 1 descriptor for slot 5
4:0	Color0	R/W	0x100C	Color 0 descriptor for slot 6
4:0	Color1	R/W	0x100D	Color 1 descriptor for slot 6
4:0	Color0	R/W	0x100E	Color 0 descriptor for slot 7
4:0	Color1	R/W	0x100F	Color 1 descriptor for slot 7

LUT1 512 REP/COLOR

Bit	Name	R/W	Address	Description
4:0	Color0	R/W	0x1010	Color 0 descriptor for slot 0
4:0	Color1	R/W	0x1011	Color 1 descriptor for slot 0
4:0	Color0	R/W	0x1012	Color 0 descriptor for slot 1
4:0	Color1	R/W	0x1013	Color 1 descriptor for slot 1
4:0	Color0	R/W	0x1014	Color 0 descriptor for slot 2
4:0	Color1	R/W	0x1015	Color 1 descriptor for slot 2
4:0	Color0	R/W	0x1016	Color 0 descriptor for slot 3
4:0	Color1	R/W	0x1017	Color 1 descriptor for slot 3
4:0	Color0	R/W	0x1018	Color 0 descriptor for slot 4
4:0	Color1	R/W	0x1019	Color 1 descriptor for slot 4
4:0	Color0	R/W	0x101A	Color 0 descriptor for slot 5
4:0	Color1	R/W	0x101B	Color 1 descriptor for slot 5
4:0	Color0	R/W	0x101C	Color 0 descriptor for slot 6
4:0	Color1	R/W	0x101D	Color 1 descriptor for slot 6
4:0	Color0	R/W	0x101E	Color 0 descriptor for slot 7
4:0	Color1	R/W	0x101F	Color 1 descriptor for slot 7

LUT2 SHAPE/COLOR

Bit	Name	R/W	Address	Description
4:0	C1	R/W	0x1020	Color 0 descriptor for slot 0
6:0	T[2:1],C2	R/W	0x1021	Shape & Color 1 descriptor for slot 0
4:0	C1	R/W	0x1022	Color 0 descriptor for slot 1
6:0	T[2:1],C2	R/W	0x1023	Shape & Color 1 descriptor for slot 1
4:0	C1	R/W	0x1024	Color 0 descriptor for slot 2
6:0	T[2:1],C2	R/W	0x1025	Shape & Color 1 descriptor for slot 2
4:0	C1	R/W	0x1026	Color 0 descriptor for slot 3
6:0	T[2:1],C2	R/W	0x1027	Shape & Color 1 descriptor for slot 3
4:0	C1	R/W	0x1028	Color 0 descriptor for slot 4
6:0	T[2:1],C2	R/W	0x1029	Shape & Color 1 descriptor for slot 4
4:0	C1	R/W	0x102A	Color 0 descriptor for slot 5
6:0	T[2:1],C2	R/W	0x102B	Shape & Color 1 descriptor for slot 5
4:0	C1	R/W	0x102C	Color 0 descriptor for slot 6
6:0	T[2:1],C2	R/W	0x102D	Shape & Color 1 descriptor for slot 6
4:0	C1	R/W	0x102E	Color 0 descriptor for slot 7
6:0	T[2:1],C2	R/W	0x102F	Shape & Color 1 descriptor for slot 7

7. Note: For more info about shapes, see section explaining Run Length Encoding

RLE DATA

Bit	Name	R/W	Address	Description
7:0	RLE code 0	R/W	0x1030	Code for byte 0
…	RLE code 1−8014		…	
7:0	RLE code 8015	R/W	0x4FFF	Code for byte 16335

There are in total 16336 bytes available to copy the RLE image data.

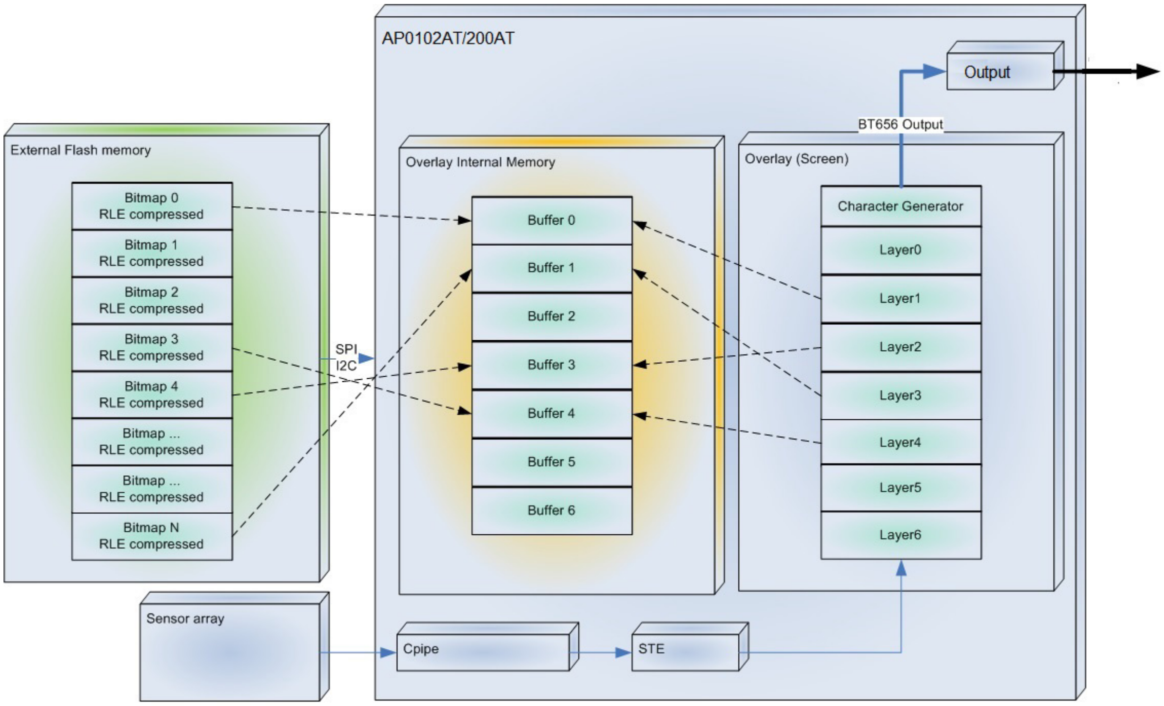


Figure 40. Overlay Data Flow

Overlay Adjustment

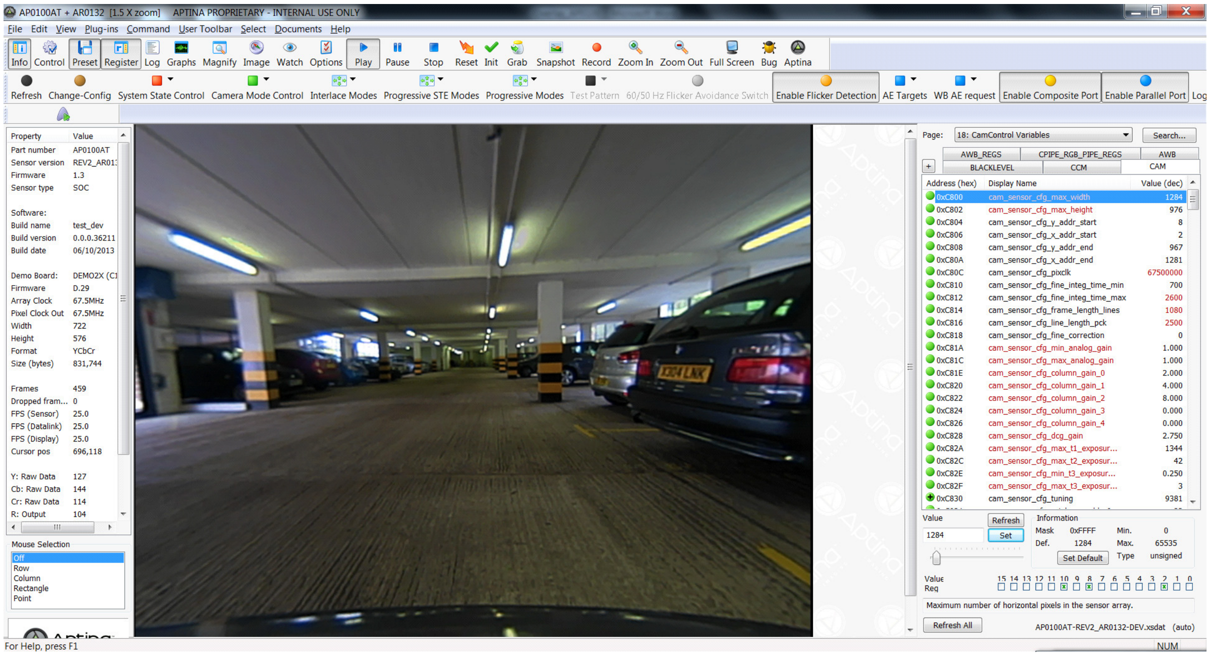
To ensure a correct position of the overlay to compensate for assembly deviation, the overlay can be adjusted with assistance from the calibration statistics engine:

- The calibration statistics engine supports a windowed 8−bin luma histogram, either row−wise (vertical) or column−wise (horizontal).
- The example calibration statistics can be used to perform an automatic successive approximation search of a cross−hair target within the scene.
- On the first frame, the firmware performs a coarse horizontal search, followed by a coarse vertical search in the second frame.
- In subsequent frames, the firmware reduces the region−of−interest of the search to the histogram bins containing the greatest accumulator values, thereby refining the search.
- The resultant X, Y location of the cross−hair target can be used to assign a calibration value of offset selected overlay graphic image positions within the output image.
- The calib stats also supports a manual mode, which allows the host to access the raw accumulator values directly.

Reverse Parking Guidance Using Overlay

The example below shows a way to create a tire tracks guidance and warning to the driver while reverse parking the car.

1. Start DevWare and point the camera to a scene where user wants to park the car


AP0100AT + AR0132 [1.5 X zoom] APTINA PROPRIETARY - INTERNAL USE ONLY
File Edit View Plug-ins Command User Toolbar Select Documents Help
Info Control Preset Register Log Graphs Magnify Image Watch Options Play Pause Stop Reset Init Grab Snapshot Record Zoom In Zoom Out Full Screen Bug Aptina
Refresh Change-Config System State Control Camera Mode Control Interlace Modes Progressive STE Modes Progressive Modes Test Pattern 60/50 Hz Flicker Avoidance Switch Enable Flicker Detection AE Targets WB AE request Enable Composite Port Enable Parallel Port Log
Page: 18: CamControl Variables
Search...
AWB_REGS CPIPE_RGB_PIPE_REGS AWB
BLACKLEVEL CCM CAM
Refresh
Set
Set Default
Refresh All
For Help, press F1
NUM

Property	Value
Part number	AP0100AT
Sensor version	REV2_AR01:
Firmware	1.3
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.36211
Build date	06/10/2013
Demo Board:	DEMO2X (C1
Firmware	D.29
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Size (bytes)	831,744
Frames	459
Dropped fram...	0
FPS (Sensor)	25.0
FPS (Datalink)	25.0
FPS (Display)	25.0
Cursor pos	696,118
Y: Raw Data	127
Cb: Raw Data	144
Cr: Raw Data	114
R: Output	104

Mouse Selection

- Off
- Row
- Column
- Rectangle
- Point

Address (hex)	Display Name	Value (dec)
0xC800	cam_sensor_cfg_max_width	1284
0xC802	cam_sensor_cfg_max_height	976
0xC804	cam_sensor_cfg_y_addr_start	8
0xC806	cam_sensor_cfg_x_addr_start	2
0xC808	cam_sensor_cfg_y_addr_end	967
0xC80A	cam_sensor_cfg_x_addr_end	1281
0xC80C	cam_sensor_cfg_pixclk	67500000
0xC810	cam_sensor_cfg_fine_integ_time_min	700
0xC812	cam_sensor_cfg_fine_integ_time_max	2600
0xC814	cam_sensor_cfg_frame_length_lines	1080
0xC816	cam_sensor_cfg_line_length_pck	2500
0xC818	cam_sensor_cfg_fine_correction	0
0xC81A	cam_sensor_cfg_min_analog_gain	1.000
0xC81C	cam_sensor_cfg_max_analog_gain	1.000
0xC81E	cam_sensor_cfg_column_gain_0	2.000
0xC820	cam_sensor_cfg_column_gain_1	4.000
0xC822	cam_sensor_cfg_column_gain_2	8.000
0xC824	cam_sensor_cfg_column_gain_3	0.000
0xC826	cam_sensor_cfg_column_gain_4	0.000
0xC828	cam_sensor_cfg_dcg_gain	2.750
0xC82A	cam_sensor_cfg_max_t1_exposur...	1344
0xC82C	cam_sensor_cfg_max_t2_exposur...	42
0xC82E	cam_sensor_cfg_min_t3_exposur...	0.250
0xC82F	cam_sensor_cfg_max_t3_exposur...	3
0xC830	cam_sensor_cfg_tuning	9381

Value: 1284

Information

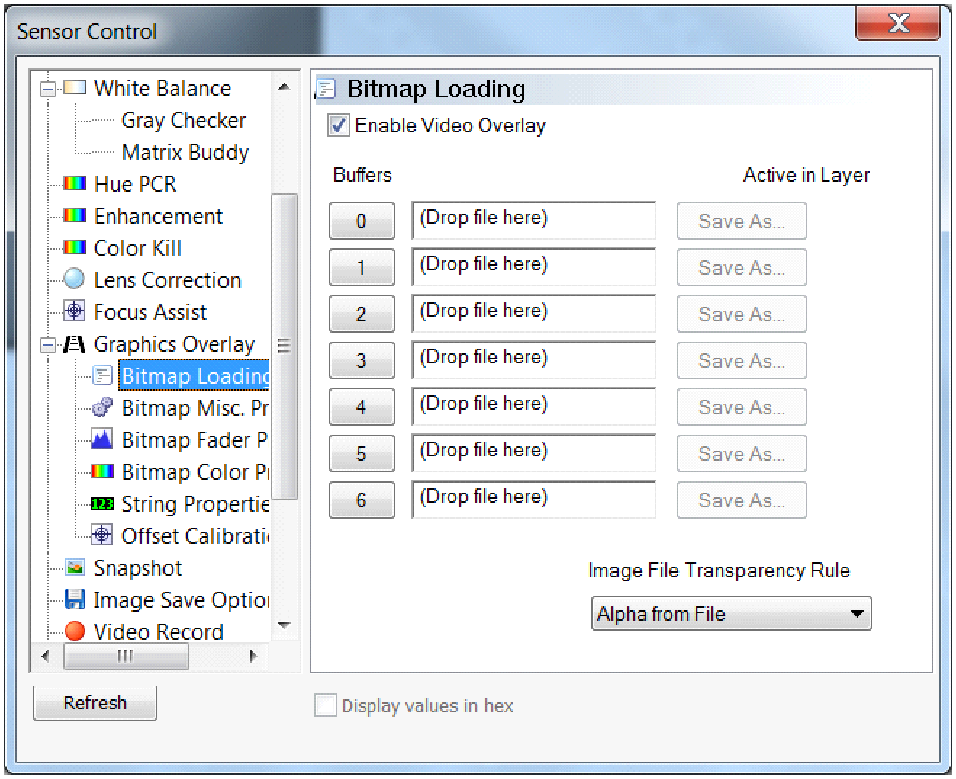
Mask	0xFFFF	Min.	0
Def.	1284	Max.	65535
		Type	unsigned

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value Reg	□	□	□	□	□	☒	□	☒	□	□	□	□	□	☒	□	□

Maximum number of horizontal pixels in the sensor array.

AP0100AT-REV2_AR0132-DEV.xsdat (auto)

2. Open *Graphics Overlay* from the camera−mode control toolbar and select *Enable Video Overlay* option


Sensor Control

Bitmap Loading

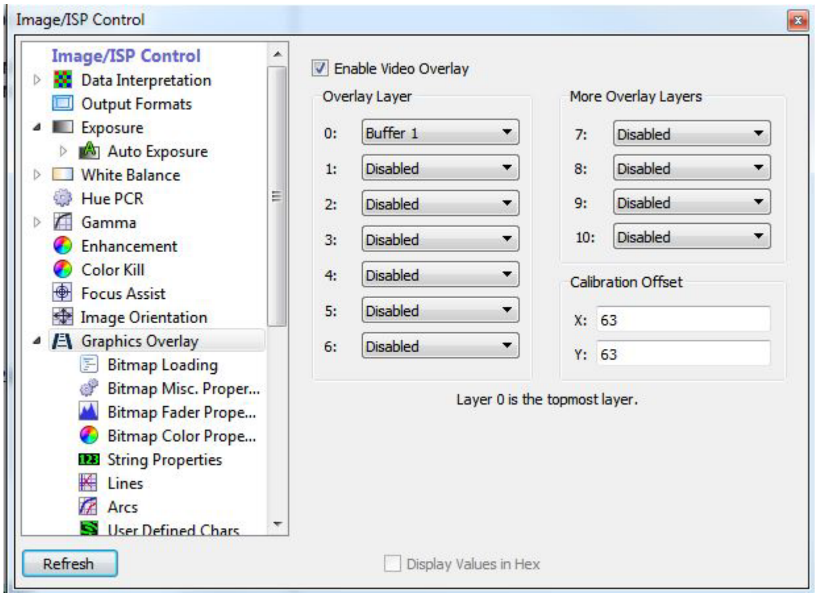
☑ Enable Video Overlay

Buffers		Active in Layer
0	(Drop file here)	Save As...
1	(Drop file here)	Save As...
2	(Drop file here)	Save As...
3	(Drop file here)	Save As...
4	(Drop file here)	Save As...
5	(Drop file here)	Save As...
6	(Drop file here)	Save As...

Image File Transparency Rule: Alpha from File

□ Display values in hex

3. Load or Drop required bitmap (Tire tracks guidance bmp image in this example) at option Buffers ‘0’. Enable Buffer 0 from *Overlay Layer* as shown below



☑ Enable Video Overlay

Overlay Layer

0:	Buffer 1
1:	Disabled
2:	Disabled
3:	Disabled
4:	Disabled
5:	Disabled
6:	Disabled

More Overlay Layers

7:	Disabled
8:	Disabled
9:	Disabled
10:	Disabled

Calibration Offset

X: 63

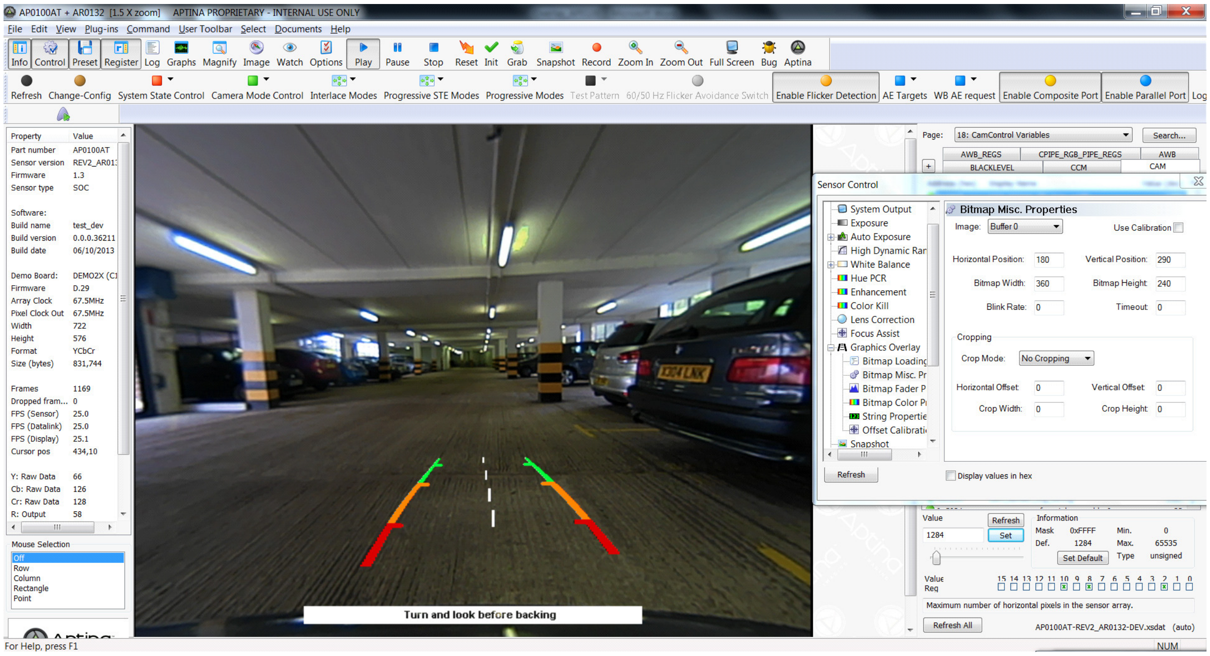
Y: 63

Layer 0 is the topmost layer.

□ Display Values in Hex

4. Similarly, Buffer 0 can be enabled under *Bitmap Misc. Properties, Bitmap Fader Properties* and *Bitmap Color Properties* tabs

5. Add the location of Region of Interest where you want to guide the driver in Horizontal /Vertical position options under *Bitmap Misc. Properties* tab


AP0100AT + AR0132 [1.5 X zoom] APTINA PROPRIETARY - INTERNAL USE ONLY
File Edit View Plug-ins Command User Toolbar Select Documents Help
Info Control Preset Register Log Graphs Magnify Image Watch Options Play Pause Stop Reset Init Grab Snapshot Record Zoom In Zoom Out Full Screen Bug Aptina
Refresh Change-Config System State Control Camera Mode Control Interlace Modes Progressive STE Modes Progressive Modes Test Pattern 60/50 Hz Flicker Avoidance Switch Enable Flicker Detection AE Targets WB AE request Enable Composite Port Enable Parallel Port Log
Turn and look before backing
Sensor Control
System Output
Exposure
Auto Exposure
High Dynamic Ran
White Balance
Hue PCR
Enhancement
Color Kill
Lens Correction
Focus Assist
Graphics Overlay
Bitmap Loading
Bitmap Misc. Pr
Bitmap Fader P
Bitmap Color Pr
String Propertie
Offset Calibrati
Snapshot
Refresh
AWB_REGS
CPIPE_RGB_PIPE_REGS
AWB
BLACKLEVEL
CCM
CAM
Search...
Refresh
Set
Set Default
Refresh All
For Help, press F1
NUM

Property	Value
Part number	AP0100AT
Sensor version	REV2_AR01:
Firmware	1.3
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.36211
Build date	06/10/2013
Demo Board:	DEMO2X (C1
Firmware	D.29
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Size (bytes)	831,744
Frames	1169
Dropped fram...	0
FPS (Sensor)	25.0
FPS (Datalink)	25.0
FPS (Display)	25.1
Cursor pos	434,10
Y: Raw Data	66
Cb: Raw Data	126
Cr: Raw Data	128
R: Output	58

Mouse Selection: Off

Page: 18: CamControl Variables

Bitmap Misc. Properties

Image: Buffer 0 Use Calibration □

Horizontal Position:	180	Vertical Position:	290
Bitmap Width:	360	Bitmap Height:	240
Blink Rate:	0	Timeout:	0

Cropping

Crop Mode: No Cropping

Horizontal Offset:	0	Vertical Offset:	0
Crop Width:	0	Crop Height:	0

□ Display values in hex

Value: 1284

Information

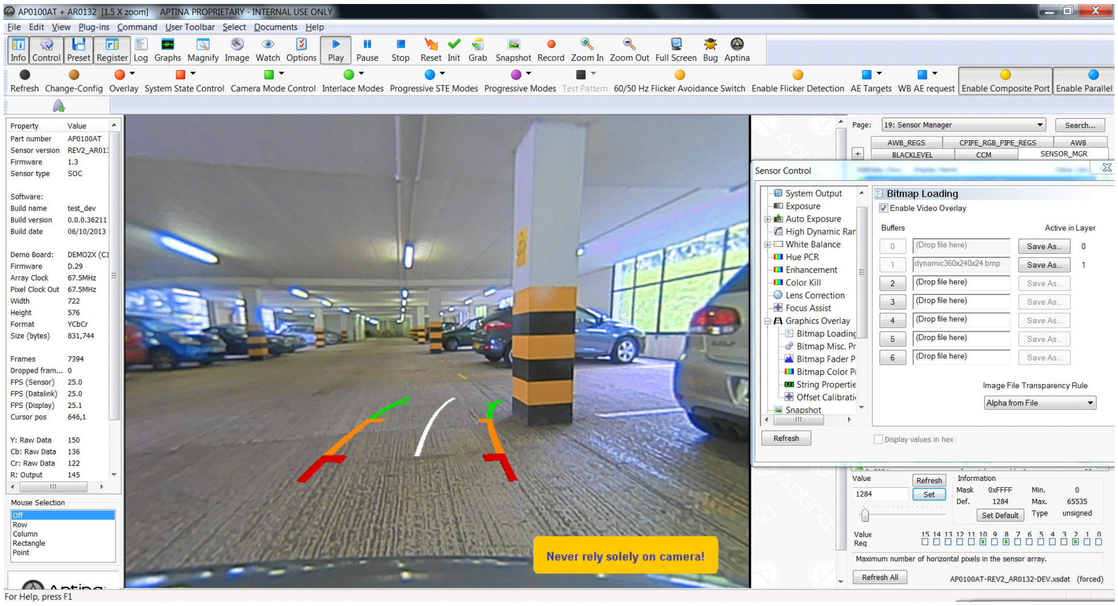
Mask	0xFFFF	Min.	0
Def.	1284	Max.	65535
		Type	unsigned

Value Reg	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	□	□	□	□	□	☒	□	☒	□	□	□	□	□	☒	□	□

Maximum number of horizontal pixels in the sensor array.

AP0100AT-REV2_AR0132-DEV.xsdat (auto)

6. Seven overlays could be loaded simultaneously in AP0102AT system. When the car moves closer to the parking space spot in this example, two different overlays are loaded to guide and warn the driver as shown in below


AP0100AT + AR0132 [1.5 X zoom] APTINA PROPRIETARY - INTERNAL USE ONLY

Property	Value
Part number	AP0100AT
Sensor version	REV2_AR01
Firmware	1.3
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.36211
Build date	06/10/2013
Demo Board:	DEMO2X (C1
Firmware	D.29
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Size (bytes)	831,744
Frames	7394
Dropped fram...	0
FPS (Sensor)	25.0
FPS (Datalink)	25.0
FPS (Display)	25.1
Cursor pos	646,1
Y: Raw Data	150
Cb: Raw Data	136
Cr: Raw Data	122
R: Output	145

Page: 19: Sensor Manager

Sensor Control — Bitmap Loading

☑ Enable Video Overlay

Buffers			Active in Layer
0	(Drop file here)	Save As...	0
1	dynamic360x240x24.bmp	Save As...	1
2	(Drop file here)	Save As...	
3	(Drop file here)	Save As...	
4	(Drop file here)	Save As...	
5	(Drop file here)	Save As...	
6	(Drop file here)	Save As...	

Image File Transparency Rule: Alpha from File

□ Display values in hex

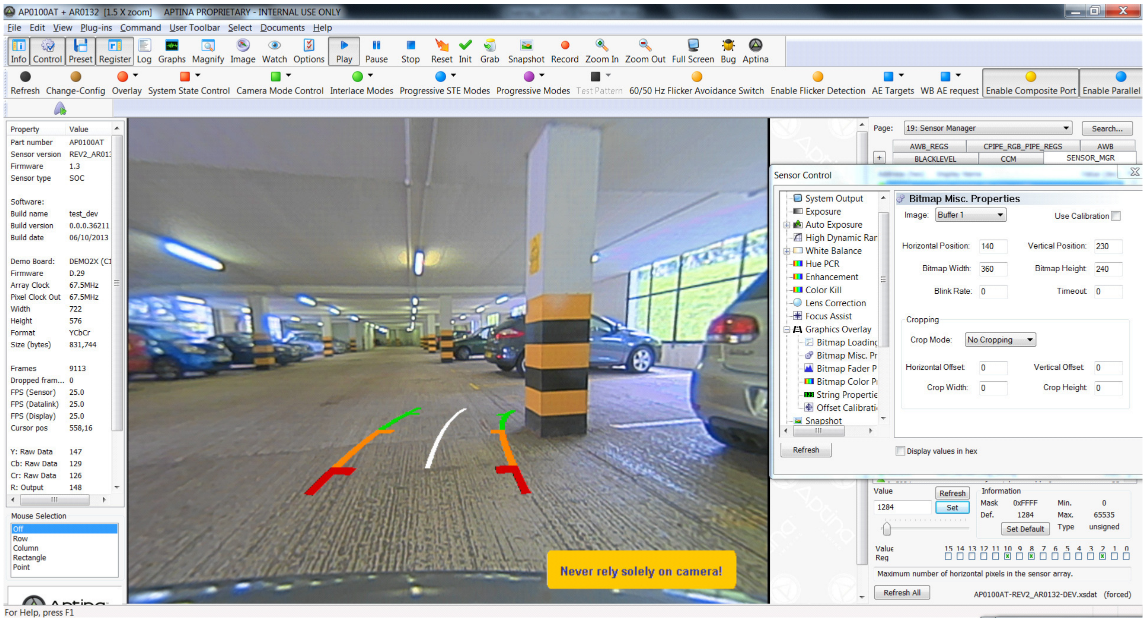
Value: 1284 — Information: Mask 0xFFFF, Def. 1284, Min. 0, Max. 65535, Type unsigned

Maximum number of horizontal pixels in the sensor array.

AP0100AT-REV2_AR0132-DEV.xsdat (forced)

For Help, press F1

7. Position of overlays is controlled same as in step 6 from *Bitmap Misc. Properties* tab


AP0100AT + AR0132 [1.5 X zoom] APTINA PROPRIETARY - INTERNAL USE ONLY

Property	Value
Part number	AP0100AT
Sensor version	REV2_AR01
Firmware	1.3
Sensor type	SOC
Software:	
Build name	test_dev
Build version	0.0.0.36211
Build date	06/10/2013
Demo Board:	DEMO2X (C1
Firmware	D.29
Array Clock	67.5MHz
Pixel Clock Out	67.5MHz
Width	722
Height	576
Format	YCbCr
Size (bytes)	831,744
Frames	9113
Dropped fram...	0
FPS (Sensor)	25.0
FPS (Datalink)	25.0
FPS (Display)	25.0
Cursor pos	558,16
Y: Raw Data	147
Cb: Raw Data	129
Cr: Raw Data	126
R: Output	148

Page: 19: Sensor Manager

Sensor Control — Bitmap Misc. Properties

Image: Buffer 1 — Use Calibration □

Horizontal Position: 140 — Vertical Position: 230

Bitmap Width: 360 — Bitmap Height: 240

Blink Rate: 0 — Timeout: 0

Cropping — Crop Mode: No Cropping

Horizontal Offset: 0 — Vertical Offset: 0

Crop Width: 0 — Crop Height: 0

□ Display values in hex

Value: 1284 — Information: Mask 0xFFFF, Def. 1284, Min. 0, Max. 65535, Type unsigned

Maximum number of horizontal pixels in the sensor array.

AP0100AT-REV2_AR0132-DEV.xsdat (forced)

For Help, press F1

In this example DevWare has been used to show how to produce dynamic overlays that change with car steering. The actual application would have the overlays stored in NVM and the steering information would be communicated via CAN/LIN bus to a microcontroller. The microcontroller would then load a different overlay depending on the steering information.

An important consideration for this type of application is that when moving the steering wheel the "tire tracks" bitmaps will be switched, however it is important that glitches in the video stream are avoided and this can be achieved using the off screen buffer.

- Layer [n] showing buffer [x] and buffer [y] is empty
- Host request loading a new bitmap to buffer [y] and assign to layer[n] (Using the load_buffer command)
- AP0102AT loads the buffer [y] straight away because it is not being used.

SPATIAL TRANSFORM ENGINE (STE)

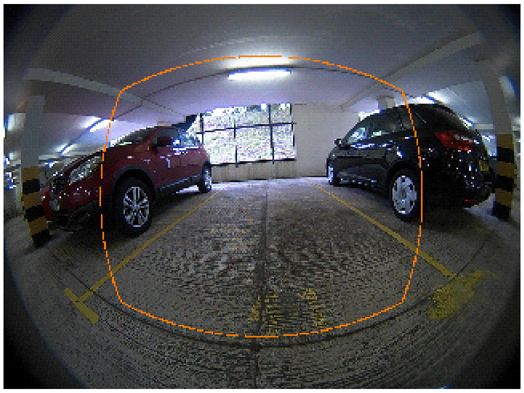
A spatial transform is defined as a transform in which some pixels are in different positions within the input and output pictures. Examples include zoom, lens distortion correction, turn, rotate, roaming, and projection. STE is a fully programmable engine that can perform spatial transforms and eliminates the need for an expensive DSP for image correction.

Lens Distortion Correction

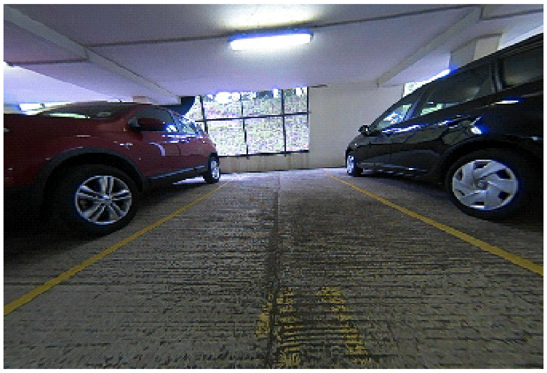
Automotive backup cameras typically feature a wide FOV lens so that a single camera−mounted above the center of the rear bumper can present the driver with a view of all potential obstacles immediately behind the full width of the vehicle. Lenses with a wide field of view typically exhibit at least a noticeable amount of barrel distortion. Barrel distortion is caused by a reduction in object magnification the further away from the optical axis.

For the image to appear natural to the driver, the AP0102AT corrects this barrel distortion and reprocesses the image so that the resulting distortion is much smaller. This is called distortion correction. Distortion correction is the ability to digitally correct the lens barrel distortion and to provide a natural view of objects. In addition, with barrel distortion one can adjust the perspective view to enhance the visibility by virtually elevating the point of viewing objects.

Below is an example of an image before and after lens distortion correction. Input image is full FOV of 1280x960 which is corrected as 1280x720 output.



Distorted full FOV image



Corrected 720P image

The distorted rectangle drawn in the uncorrected image illustrates the field of view that is maintained in the corrected image before vertical cropping. Notice that a significant amount of the original field of view in the corners of the image is lost due to distortion correction. The field of view to be maintained was selected so that the original horizontal field of view at the center of the image is maintained. The AP0102AT only needs to perform distortion correction for the field of view that will be maintained in the output image.

Perspective View

A backup camera has to be able to virtually adjust the vertical perspective as if the camera were placed immediately behind the vehicle pointed directly down, as illustrated in Figure 41. The vertical perspective adjustment may be employed temporarily to assist with parking conditions, or it may be enabled permanently by loading new parameters.

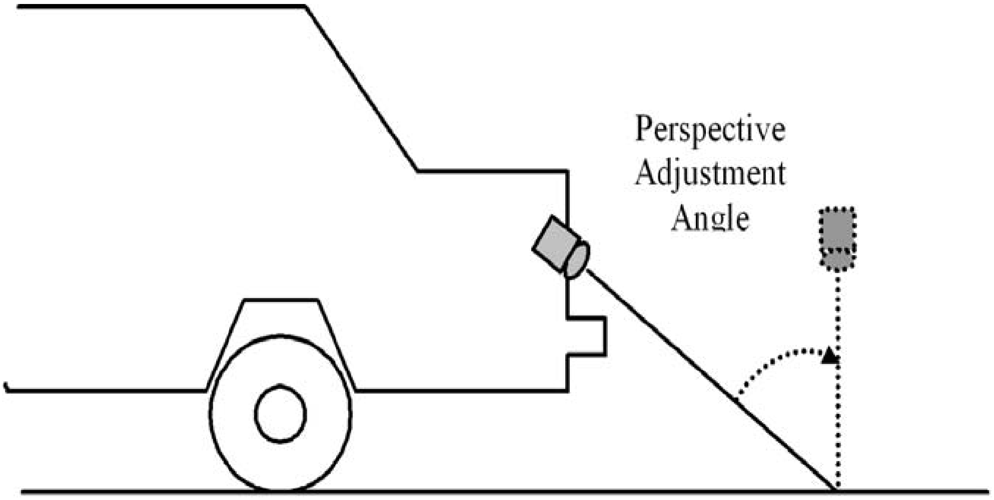
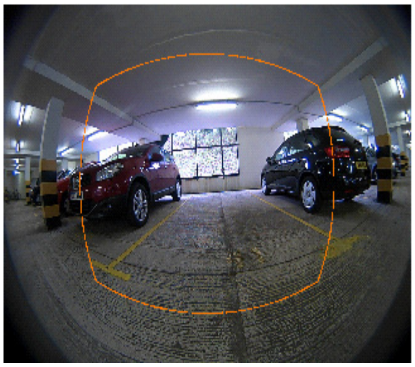
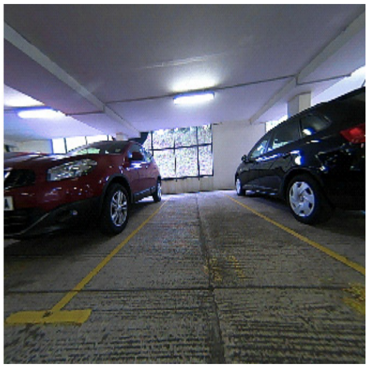


Figure 41. Vertical Perspective Adjustment

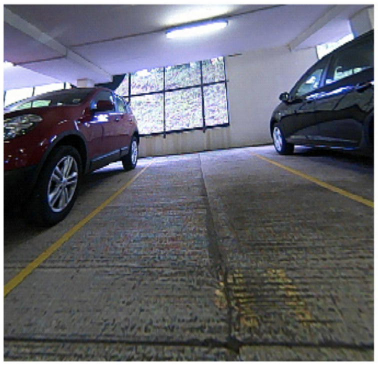
The example below shows perspective view adjustment to guide the driver to park the car.



Distorted Full FOV
input Image



Distortion Corrected
image with highlighted
Parking region

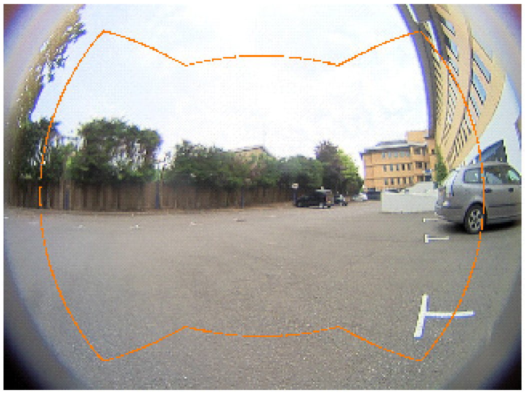


Perspective adjusted output
image to guide parking spot

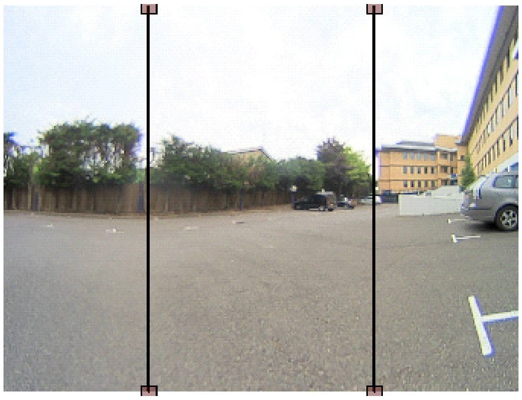
Multi−Panel

STE supports multi−panel views; these can be two or three panels. This feature is ideally suited for applications where viewing at a junction is required.

The example below shows the three panels view of selected part of input image.



Distorted Full FOV input image



Distortion Corrected three panels view image

STE Control and GUI

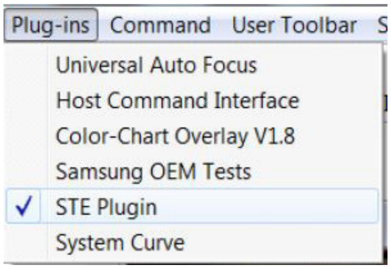
STE is a very powerful feature and it is controlled or configured using DevWare plug−ins. It controls much more than image transforms. The output from STE is called a “blob” (in xml format). Table 40 shows different features for which STE can be used for.

Table 40. FEATURES OF STE

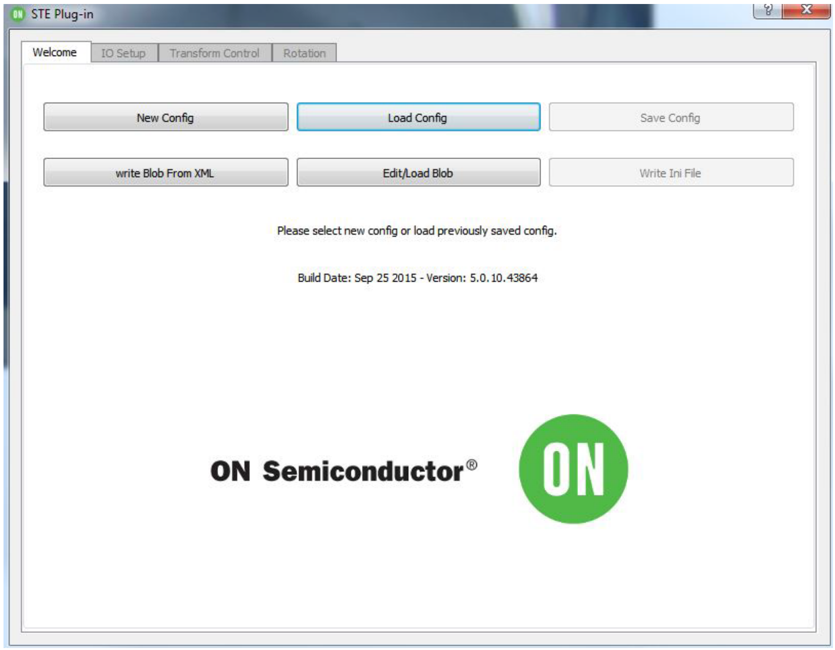
Feature	Notes
Input interface	Parallel or HiSPi
Image timing	H/V timing
PLL timing	Changing sensor or CC PLL
Operation mode	SDR vs. HDR
Image orientation	Flip/Mirror
STE transform	Zoom, Lens Distortion Correction, Mirror/Flip, Turn, Rotate, Pan, Tilt, Diptych/Triptych Displays, Aspect Ratio, Roaming and Projection, etc.

Following is a detailed explanation of different functions of the STE plug−in tool:

1. STE Plug−in GUI launches from Plug−ins option of DevWare



2. On start−up, IO setup and Transform Control options are grayed out

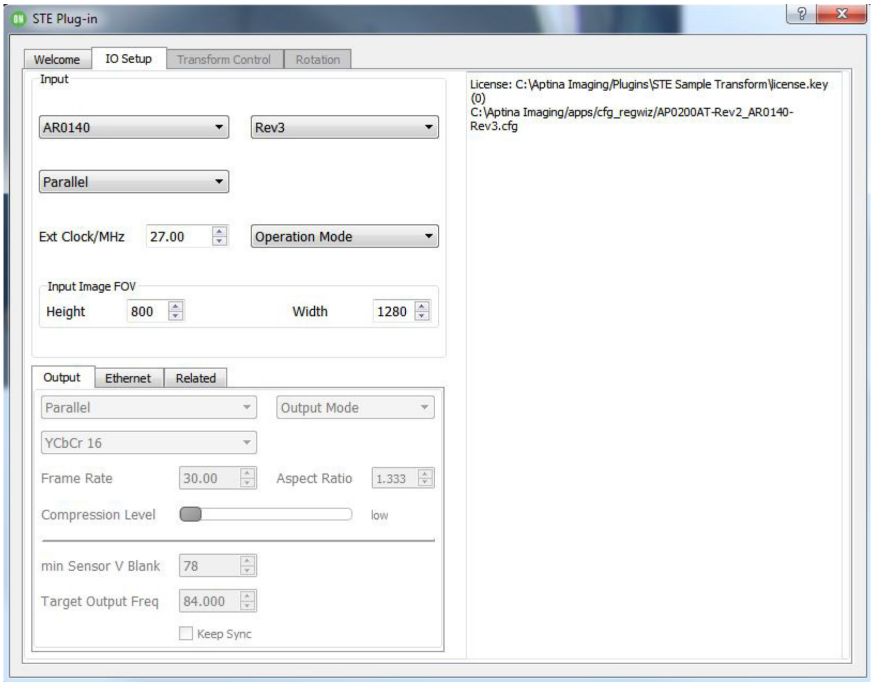


Please select new config or load previously saved config.

Build Date: Sep 25 2015 - Version: 5.0.10.43864

- New Config – This creates a new configuration
- Save Config – This saves the configuration to a file in ‘xml’ format
- Load Config –This loads an existing configuration from file
- Write Ini File
  This saves the configuration to a file in ‘Ini’ format

3. Select *New Config.*
   It opens the *IO Setup* tab with some of the options like Sensor Name, Sensor Revision, Input Image FOV (Height and Width) automatically detected from the sensor connected to the AP0102AT



AR0140 | Rev3

Parallel

Ext Clock/MHz 27.00 | Operation Mode

Input Image FOV

Height 800 | Width 1280

Parallel | Output Mode

YCbCr 16

Frame Rate 30.00 | Aspect Ratio 1.333

Compression Level

min Sensor V Blank 78

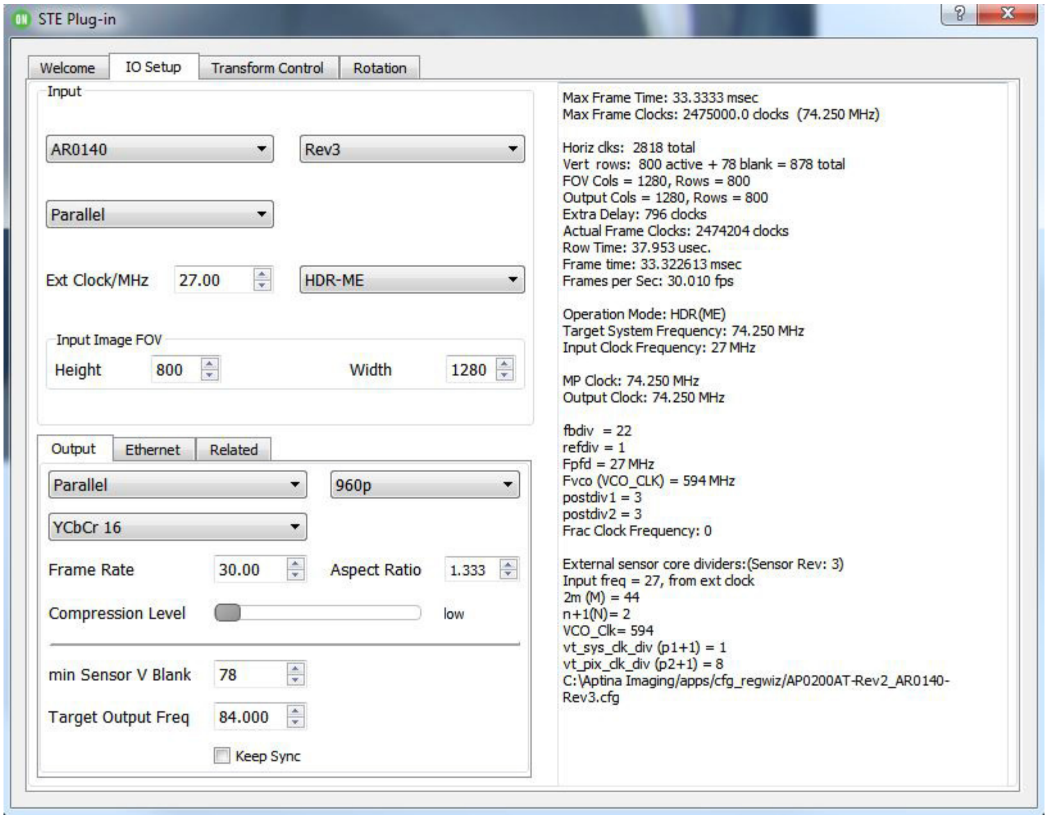
Target Output Freq 84.000

□ Keep Sync

License: C:\Aptina Imaging/Plugins\STE Sample Transform\license.key
(0)
C:\Aptina Imaging/apps/cfg_regwiz/AP0200AT-Rev2_AR0140-Rev3.cfg

- – Sensor Name – Select attached supported sensor
- – Sensor Rev – 1, 2…
- – Input Interface – Parallel, HiSPi12, HiSPi14
- – Input Ext clock – 6–30 MHz (27 MHz is typical)
- – Operation mode – SDR, HDR (ME) and HDR (DLO)
- – Input Image FOV – This is the input FOV, excluding calibration pixels of the attached sensor

4. Select type of *Input Interface* and *Operation Mode*, which activates the *output, ethernet* and *related tabs*



AR0140 | Rev3

Parallel

Ext Clock/MHz 27.00 | HDR-ME

Input Image FOV

Height 800 | Width 1280

Parallel | 960p

YCbCr 16

Frame Rate 30.00 | Aspect Ratio 1.333

Compression Level

min Sensor V Blank 78

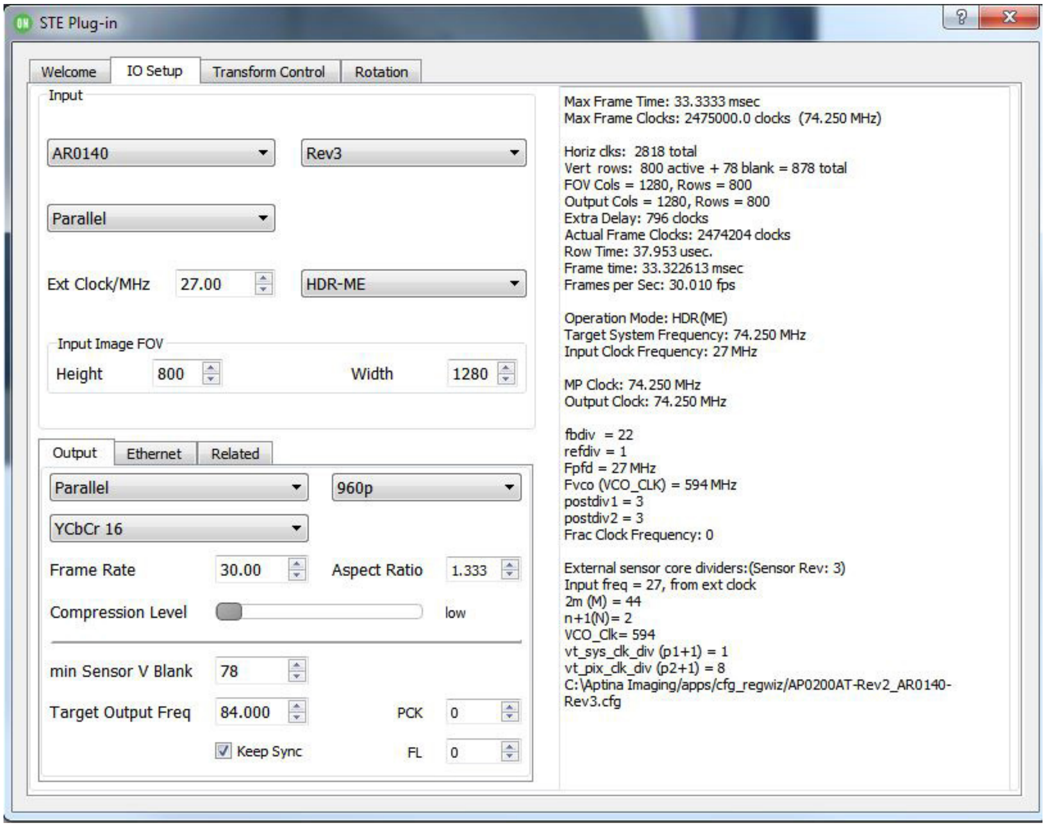
Target Output Freq 84.000

☐ Keep Sync

```
Max Frame Time: 33.3333 msec
Max Frame Clocks: 2475000.0 clocks  (74.250 MHz)

Horiz clks:  2818 total
Vert  rows:  800 active + 78 blank = 878 total
FOV Cols = 1280, Rows = 800
Output Cols = 1280, Rows = 800
Extra Delay: 796 clocks
Actual Frame Clocks: 2474204 clocks
Row Time: 37.953 usec.
Frame time: 33.322613 msec
Frames per Sec: 30.010 fps

Operation Mode: HDR(ME)
Target System Frequency: 74.250 MHz
Input Clock Frequency: 27 MHz

MP Clock: 74.250 MHz
Output Clock: 74.250 MHz

fbdiv  = 22
refdiv = 1
Fpfd = 27 MHz
Fvco (VCO_CLK) = 594 MHz
postdiv1 = 3
postdiv2 = 3
Frac Clock Frequency: 0

External sensor core dividers:(Sensor Rev: 3)
Input freq = 27, from ext clock
2m (M) = 44
n+1(N)= 2
VCO_Clk= 594
vt_sys_clk_div (p1+1) = 1
vt_pix_clk_div (p2+1) = 8
C:\Aptina Imaging/apps/cfg_regwiz/AP0200AT-Rev2_AR0140-
Rev3.cfg
```

Output options – Parallel or Ethernet.
Output Mode – 960p/800p/720p
Aspect Ration, Frame Rate, Output size and output Freq selections in the Output Tab
Bit alignment option in the related tab

5. After filling in the required operating mode, the calculation appears on right hand side and *Transform Control* tab activates

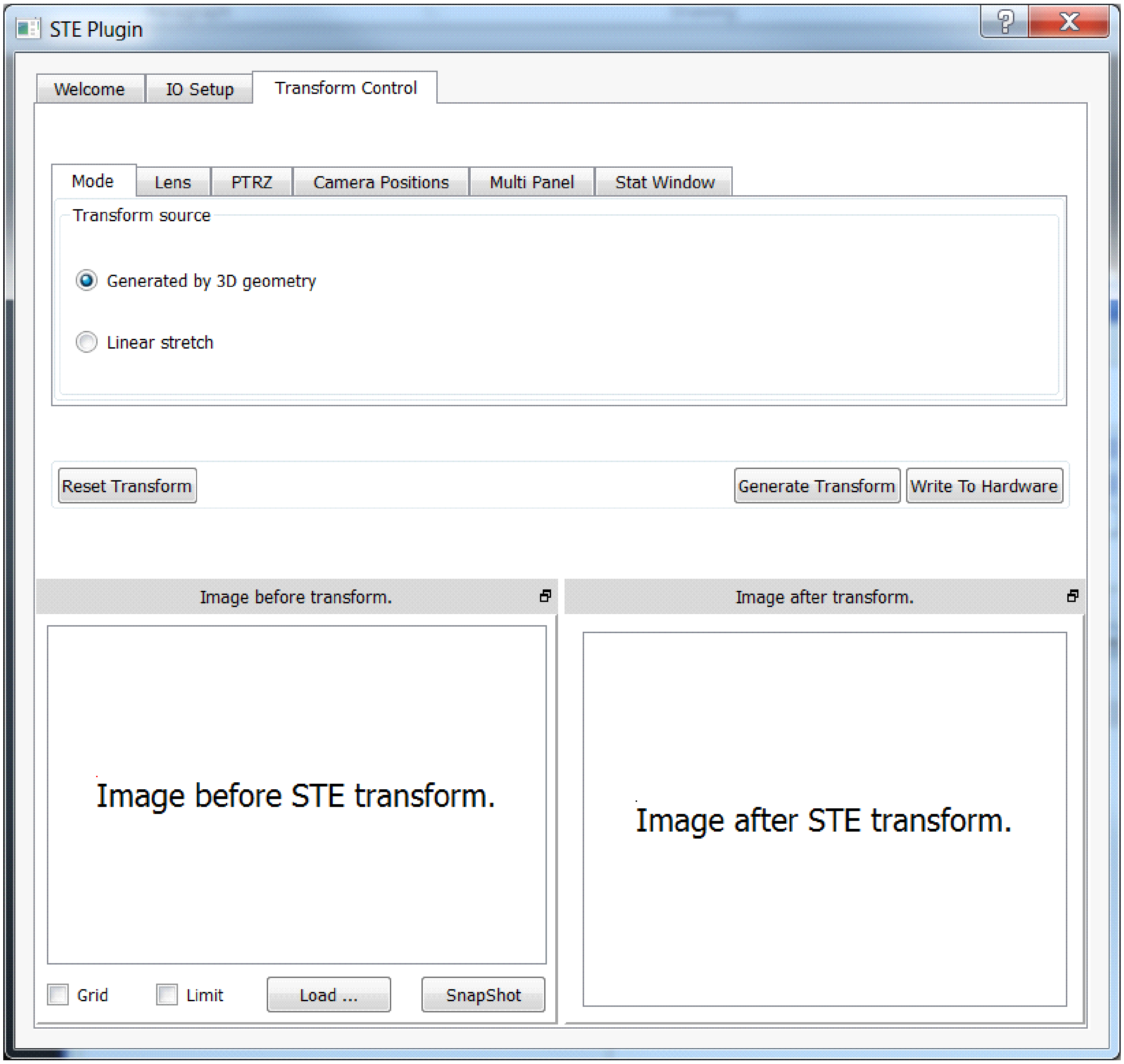


AR0140
Rev3
Parallel
Ext Clock/MHz 27.00
HDR-ME
Height 800
Width 1280

Parallel
960p
YCbCr 16
Frame Rate 30.00
Aspect Ratio 1.333
Compression Level low
min Sensor V Blank 78
Target Output Freq 84.000
PCK 0
☑ Keep Sync
FL 0

```
Max Frame Time: 33.3333 msec
Max Frame Clocks: 2475000.0 clocks  (74.250 MHz)

Horiz clks:  2818 total
Vert  rows:  800 active + 78 blank = 878 total
FOV Cols = 1280, Rows = 800
Output Cols = 1280, Rows = 800
Extra Delay: 796 clocks
Actual Frame Clocks: 2474204 clocks
Row Time: 37.953 usec.
Frame time: 33.322613 msec
Frames per Sec: 30.010 fps

Operation Mode: HDR(ME)
Target System Frequency: 74.250 MHz
Input Clock Frequency: 27 MHz

MP Clock: 74.250 MHz
Output Clock: 74.250 MHz

fbdiv  = 22
refdiv = 1
Fpfd = 27 MHz
Fvco (VCO_CLK) = 594 MHz
postdiv1 = 3
postdiv2 = 3
Frac Clock Frequency: 0

External sensor core dividers:(Sensor Rev: 3)
Input freq = 27, from ext clock
2m (M) = 44
n+1(N)= 2
VCO_Clk= 594
vt_sys_clk_div (p1+1) = 1
vt_pix_clk_div (p2+1) = 8
C:\Aptina Imaging/apps/cfg_regwiz/AP0200AT-Rev2_AR0140-
Rev3.cfg
```

6. Select *Transform Control* option

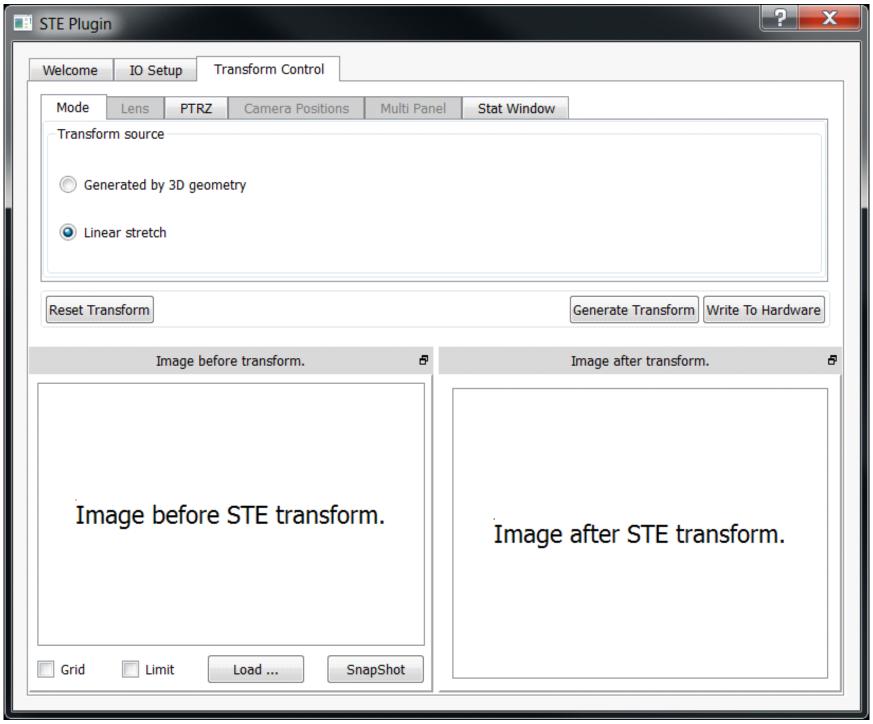


Transform source

● Generated by 3D geometry

○ Linear stretch

NOTE: For Linear Stretch, most of the tabs are grayed out like shown below.

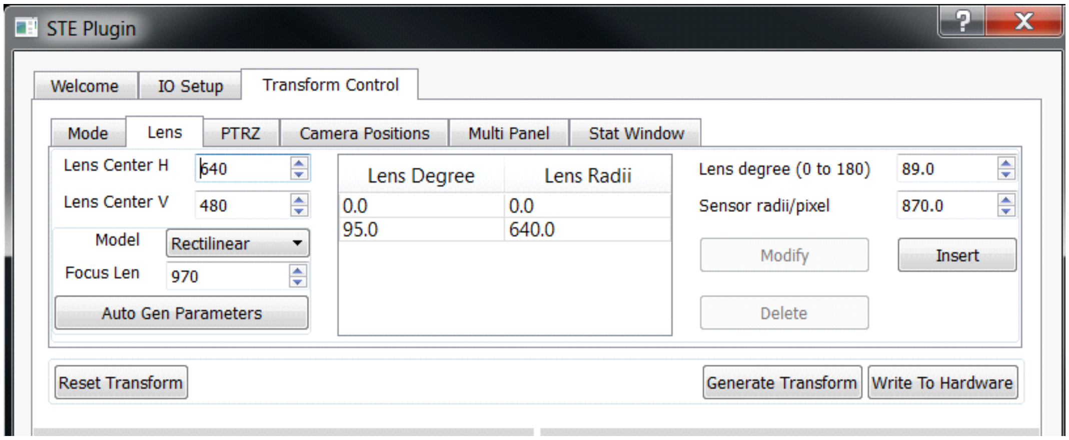


Transform source

○ Generated by 3D geometry

● Linear stretch

7. Transform−Lens



Lens Center H: 640

Lens Center V: 480

Model: Rectilinear

Focus Len: 970

Lens Degree	Lens Radii
0.0	0.0
95.0	640.0

Lens degree (0 to 180): 89.0

Sensor radii/pixel: 870.0

Figure 42.

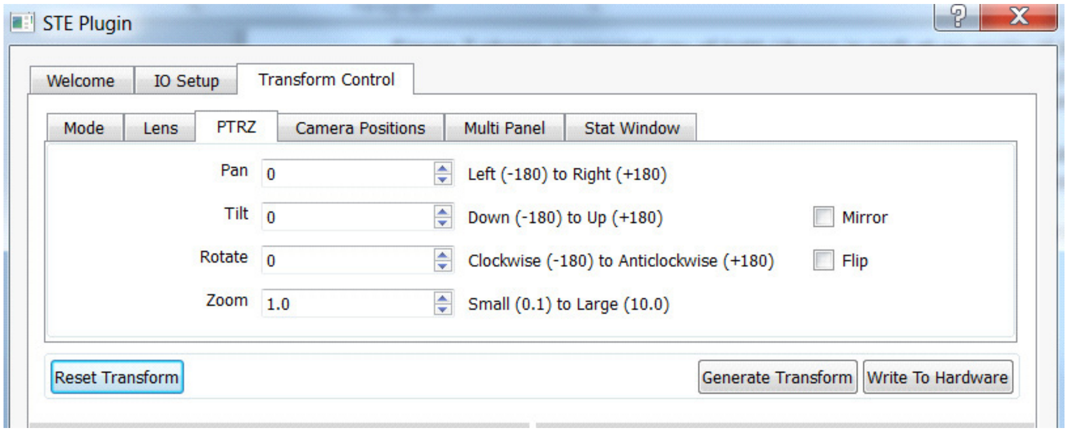
Lens Center H/V− The horizontal and vertical centers of the lens with respect to the sensor.

Lens Degree/Radii− This would be the information of position of light at angle x as a position y in pixels on the sensor. The lens vendor will provide this to the customer, or the customer would measure this.

The default numbers are what have been used by ON Semiconductor for demo camera lens.

Model/Focus Lens− AP0102AT supports different lens models for the users as they wish to use them with supported sensors. They would need to input Focus Length in pixels.

8. Transform−PRTZ (Pan, Rotate, Tilt, Zoom)



Pan 0 Left (-180) to Right (+180)

Tilt 0 Down (-180) to Up (+180) □ Mirror

Rotate 0 Clockwise (-180) to Anticlockwise (+180) □ Flip

Zoom 1.0 Small (0.1) to Large (10.0)

Pan− + pan to right −pan to left. From −180 to +180

Tilt− +tilt up −tilt down. From −180 to +180

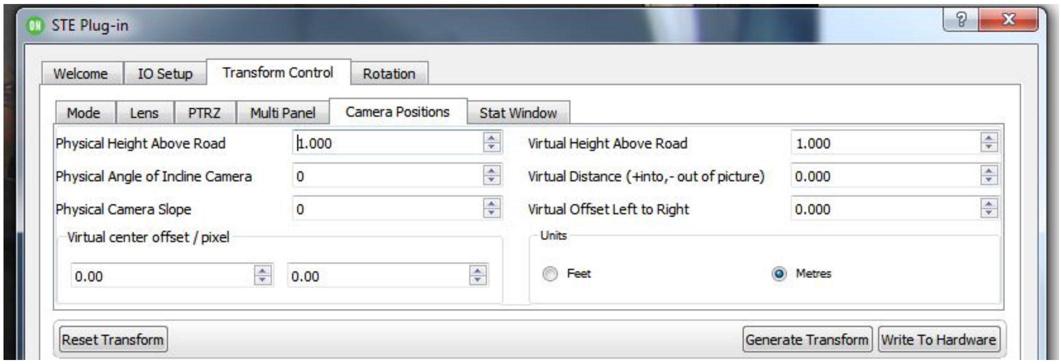
Rotate− +rotate clockwise −rotate anticlockwise. From −180 to +180.

Zoom− As the number gets bigger the user will zoom in. From 0.1 to 10.0

Mirror− Mirror the image

Flip− Flip the image

9. Transform−Camera Positions



Physical Height Above Road 1.000

Physical Angle of Incline Camera 0

Physical Camera Slope 0

Virtual center offset / pixel 0.00 0.00

Virtual Height Above Road 1.000

Virtual Distance (+into,- out of picture) 0.000

Virtual Offset Left to Right 0.000

Units ○ Feet ● Metres

Physical− Take into account of physical position of the camera

Virtual− Is with regard to virtual camera concept

Camera Position controls the orientation of the ground plane with respect to the actual camera, and the position within this frame of reference to the virtual camera. Note that moving the virtual camera away from the actual may cause unpleasant distortion in the image, particularly in those parts of the input picture which are not, in fact, on the ground plane.

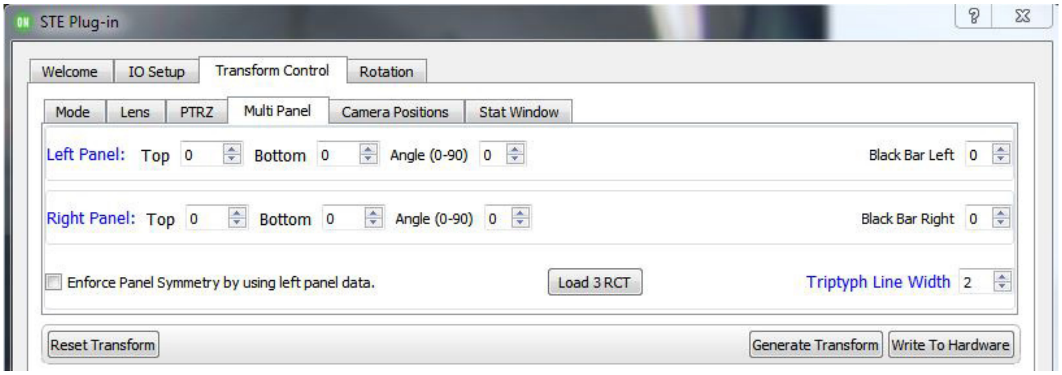
Physical Height Above Road is the height of the real camera above the road surface (measured at the nearest point, perpendicular to the road. Virtual Height Above Road is the corresponding measurement for the virtual camera.

Physical Angle of Incline Camera controls the mounting angle of the actual camera in an up/down sense. 0° means that the camera is perpendicular to the road (pointing straight down). 90° means that the camera is horizontal, with its axis parallel to the road.

Physical Camera Slope controls the mounting angle of the actual camera in a left/right sense. The aim is to compensate for any slight left/right slope resulting from manufacturing errors in the mounting position.

Virtual offset in pixels can be added.

10. Multi Panel



Left Panel: Top 0 Bottom 0 Angle (0-90) 0 Black Bar Left 0

Right Panel: Top 0 Bottom 0 Angle (0-90) 0 Black Bar Right 0

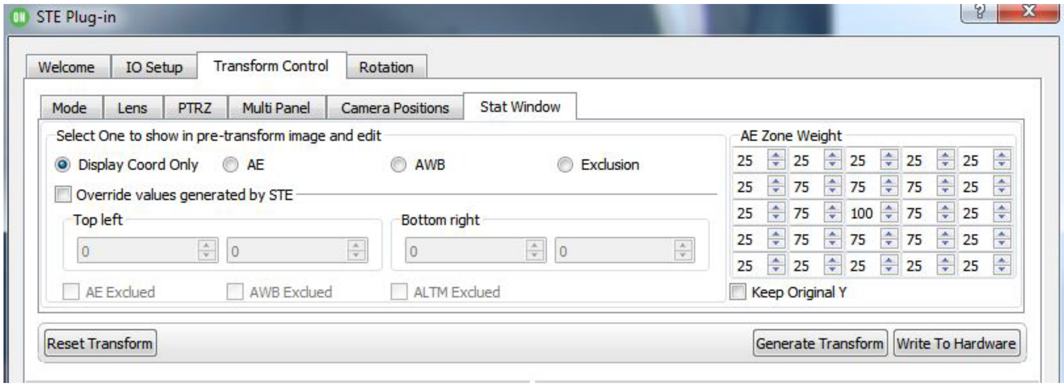
□ Enforce Panel Symmetry by using left panel data.

Triptyph Line Width 2

Multi–Panel allows the user to specify up to three planes onto which the output picture is to be projected. These planes are then unfolded along their boundaries with one another to give a combined flat picture. An optional black line is placed over the boundary to improve scene understanding.

- Left Panel Top – This is in output pixels with 0 being top left
- Left Panel Bottom – This is in output pixels with 0 being bottom left
- Right Panel Top – This is in output pixels with 0 being top right
- Right Panel Bottom – This is in output pixels with 0 being bottom right
- Angle – This is the angle you wish to view in the panel (0–90)
- Black bar – Will put a black bar on the left and right of the image that number of pixels wide
- Triptych line width – Width of dividing lines between the panels (can be zero if required)
- Enforce panel symmetry – This will apply same settings to left and right panel

11. Transform–Stat Window



Select One to show in pre-transform image and edit

● Display Coord Only ○ AE ○ AWB ○ Exclusion

□ Override values generated by STE

Top left: 0 0

Bottom right: 0 0

□ AE Exclued □ AWB Exclued □ ALTM Exclued

AE Zone Weight

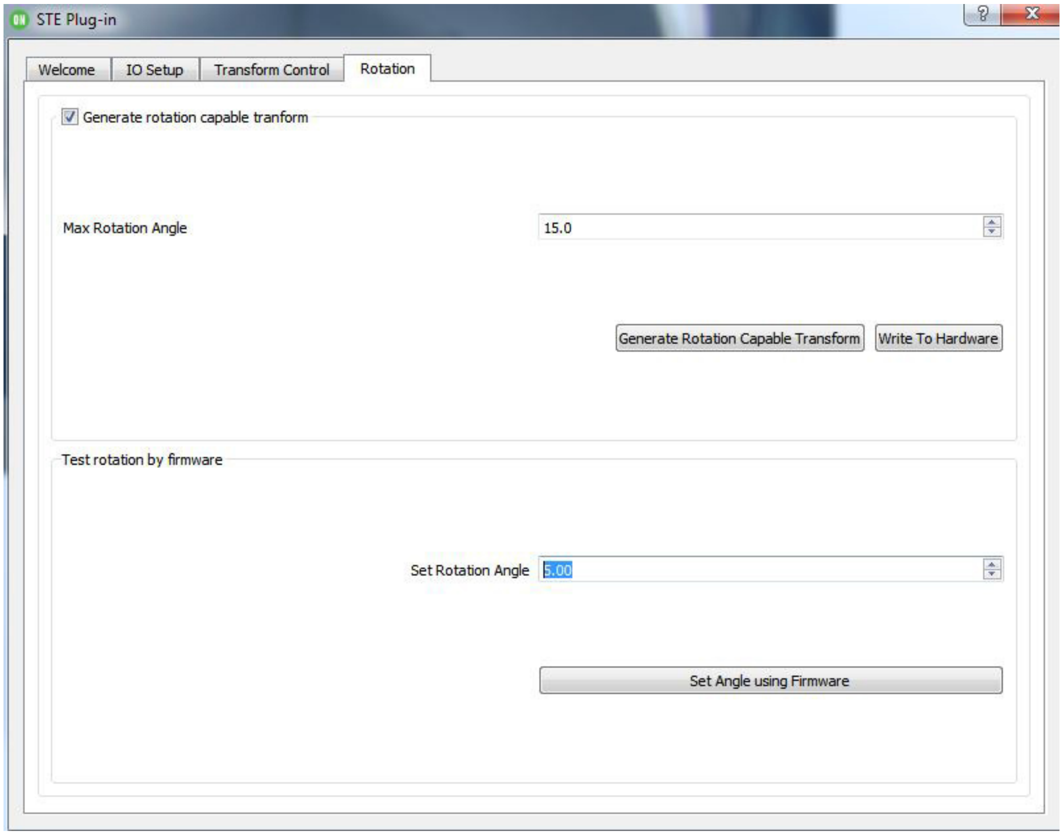
25	25	25	25	25
25	75	75	75	25
25	75	100	75	25
25	75	75	75	25
25	25	25	25	25

□ Keep Original Y

The user may configure the ‘stats window’ for each hardware collection engine. This controls which pixels in the scene are processed by the hardware.

More information about Stat Window can be found in STE Statistics section.

12. Rotation tab allows to easily set the STE rotation ranges and value



STE Plug-in

☑ Generate rotation capable tranform

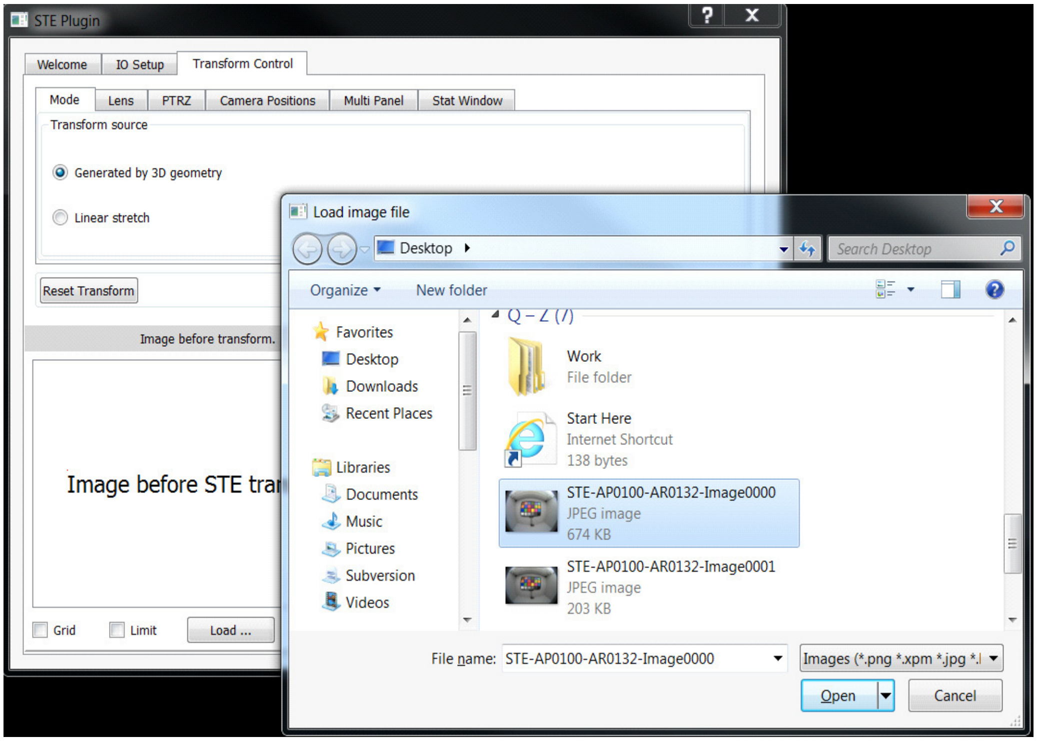
Max Rotation Angle: 15.0

Test rotation by firmware

Set Rotation Angle: 5.00

13. Transform−Load Image

    a. The user can load a saved full resolution image.



STE Plugin

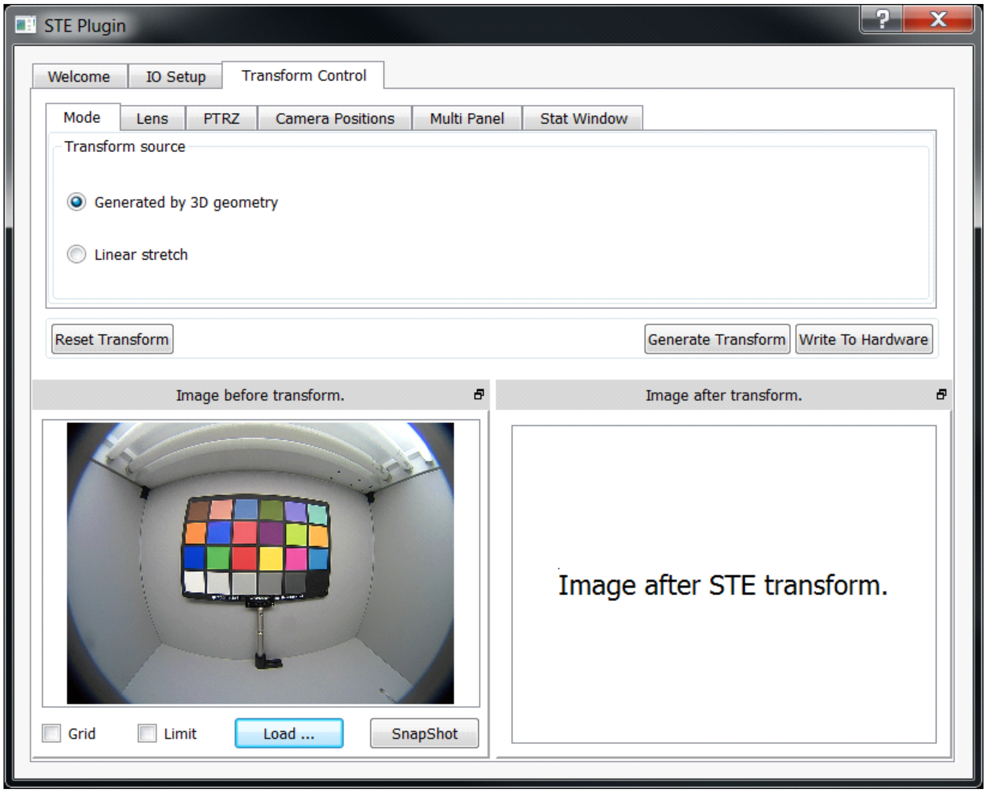
Transform source

● Generated by 3D geometry

○ Linear stretch

File name: STE-AP0100-AR0132-Image0000

Images (*.png *.xpm *.jpg *.l



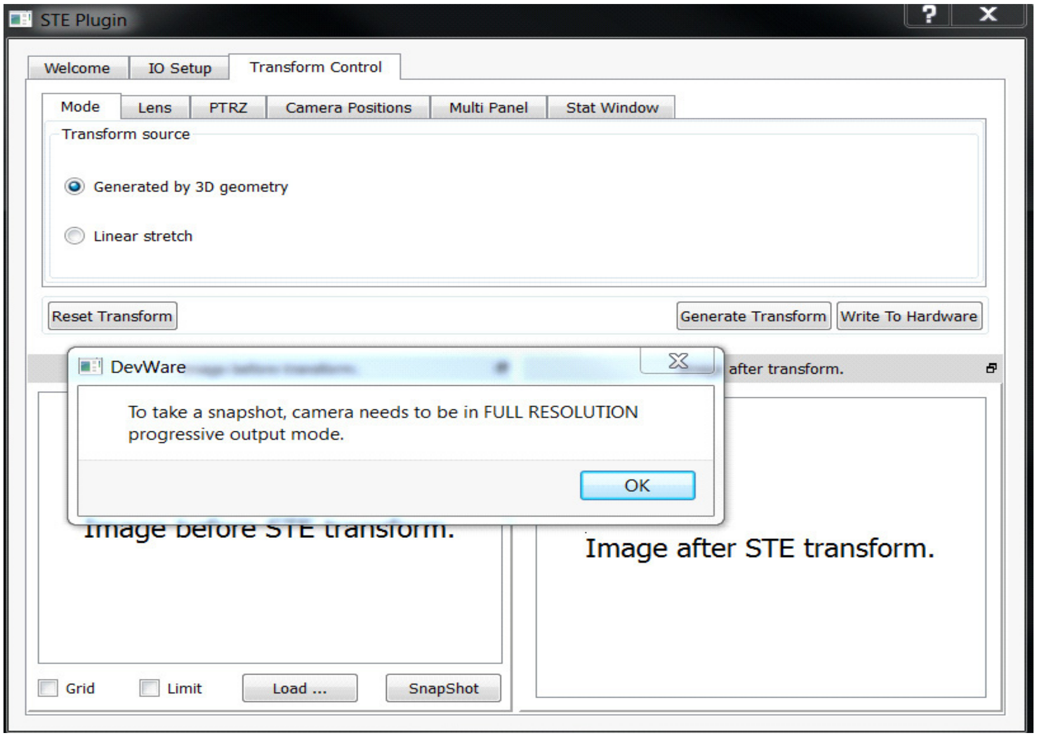
Transform source

● Generated by 3D geometry

○ Linear stretch

□ Grid □ Limit

b. The user can also snap an image



Transform source

● Generated by 3D geometry

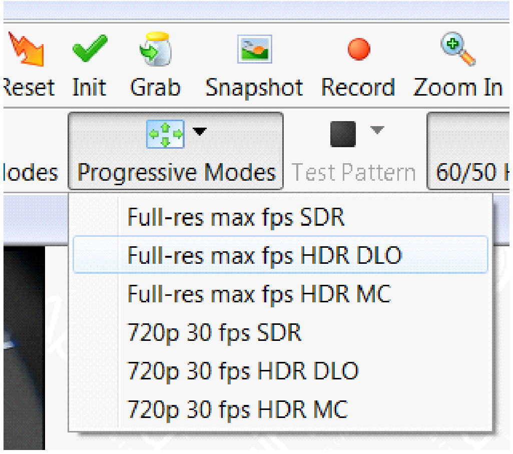
○ Linear stretch

To take a snapshot, camera needs to be in FULL RESOLUTION progressive output mode.

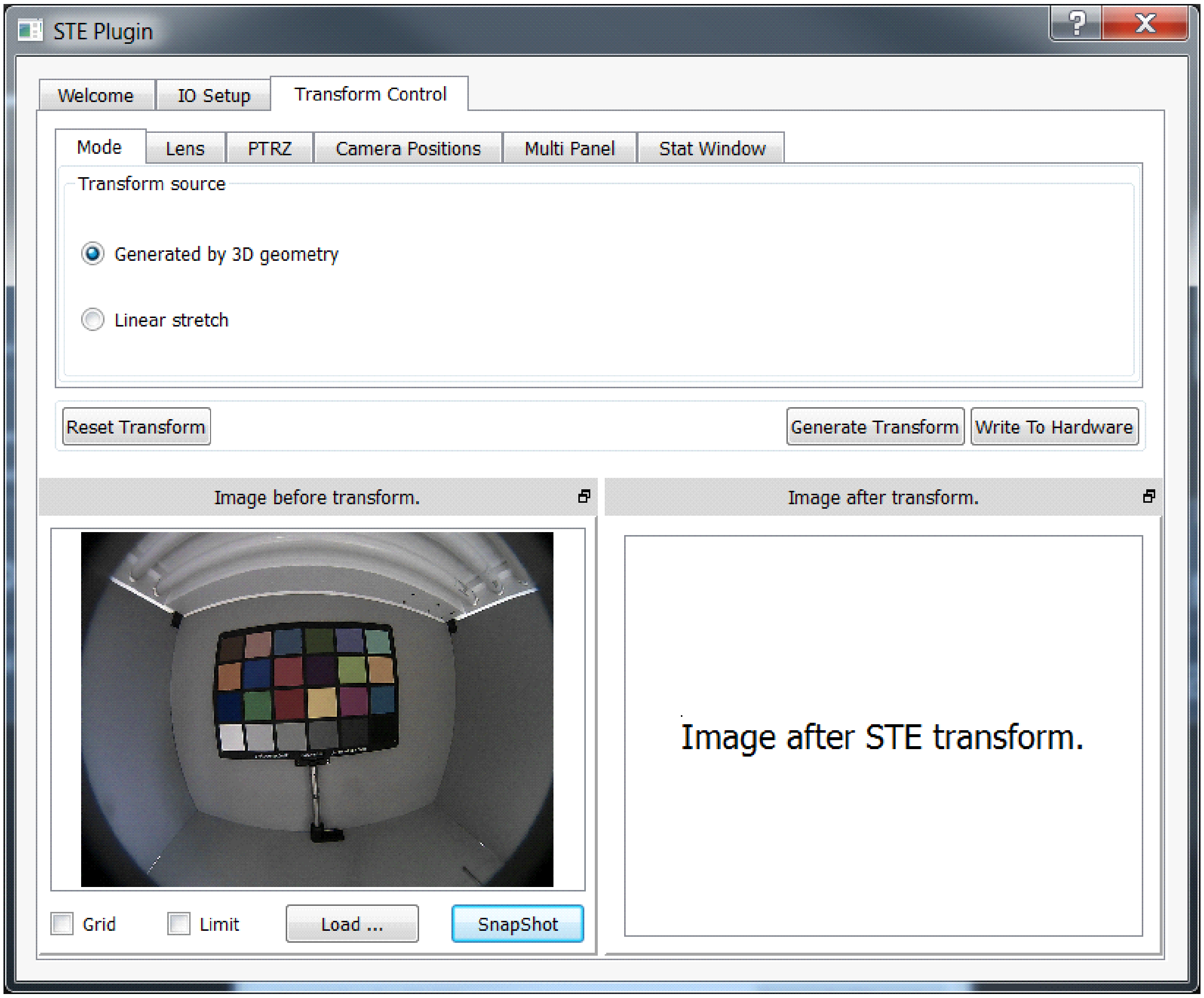
□ Grid □ Limit

Procedure to snap an image:

a. Load "Full−res max fps HDR DLO" DevWare toolbar



b. Click "snapshot" on STE Plug−in tool

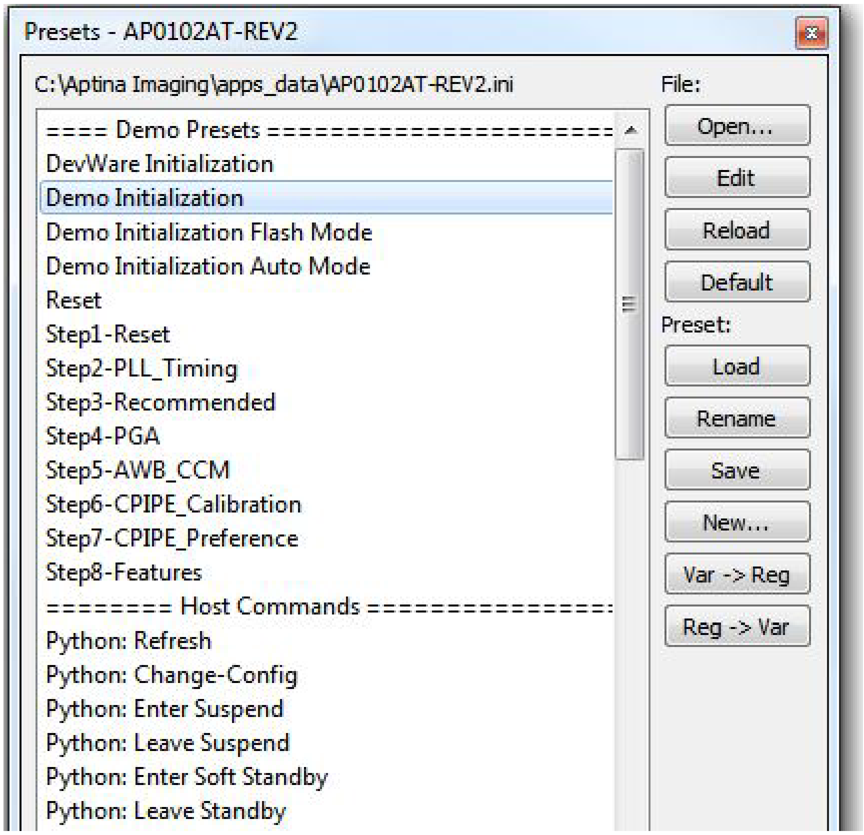


Transform source

● Generated by 3D geometry

○ Linear stretch

c. Put camera back into default condition by running “demo initialization” or “DevWare initialization” from Preset

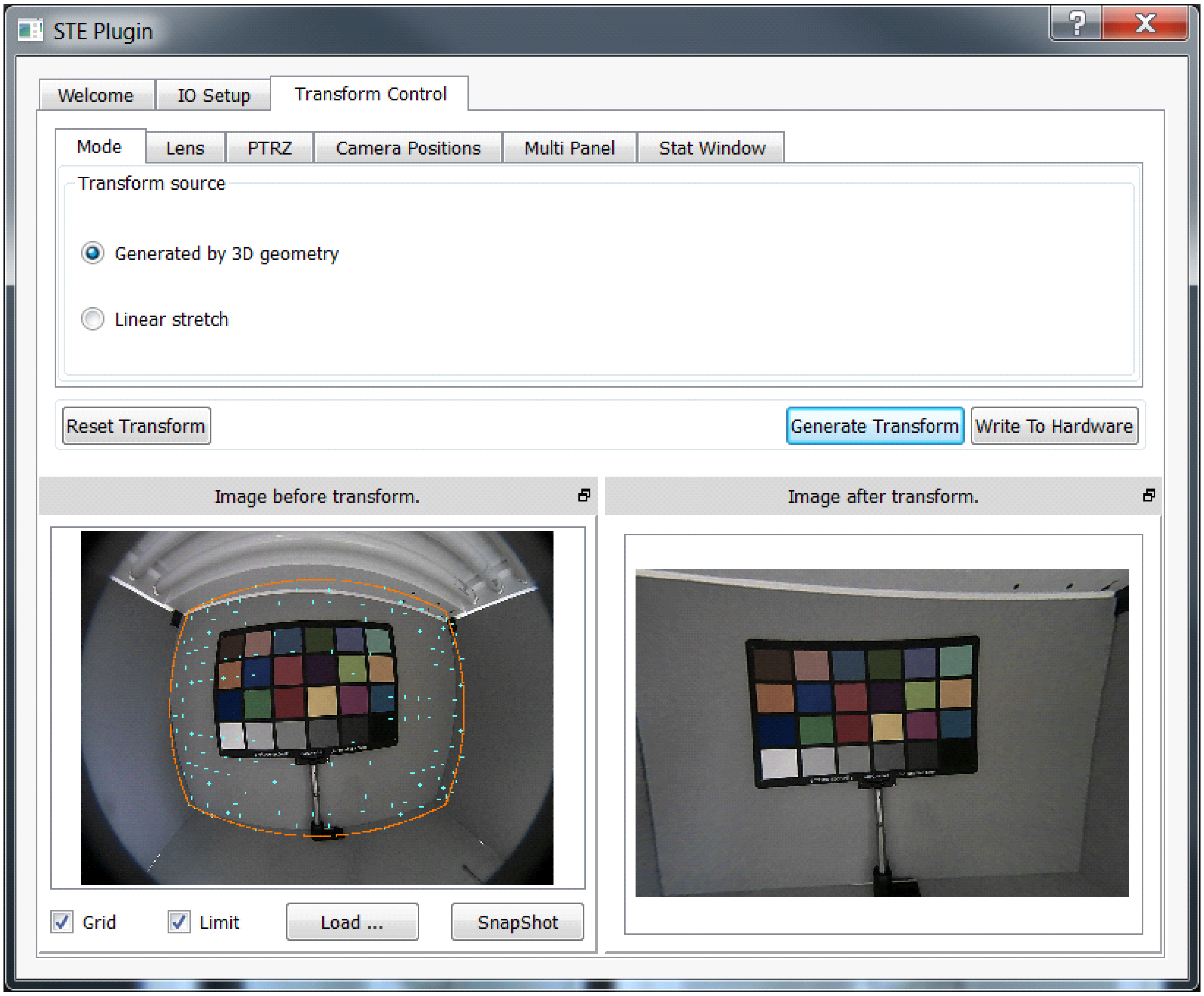


Presets - AP0102AT-REV2

C:\Aptina Imaging\apps_data\AP0102AT-REV2.ini

==== Demo Presets ======================
DevWare Initialization
Demo Initialization
Demo Initialization Flash Mode
Demo Initialization Auto Mode
Reset
Step1-Reset
Step2-PLL_Timing
Step3-Recommended
Step4-PGA
Step5-AWB_CCM
Step6-CPIPE_Calibration
Step7-CPIPE_Preference
Step8-Features
======== Host Commands ================
Python: Refresh
Python: Change-Config
Python: Enter Suspend
Python: Leave Suspend
Python: Enter Soft Standby
Python: Leave Standby

14. Transform−Load Image − Grid/Limit

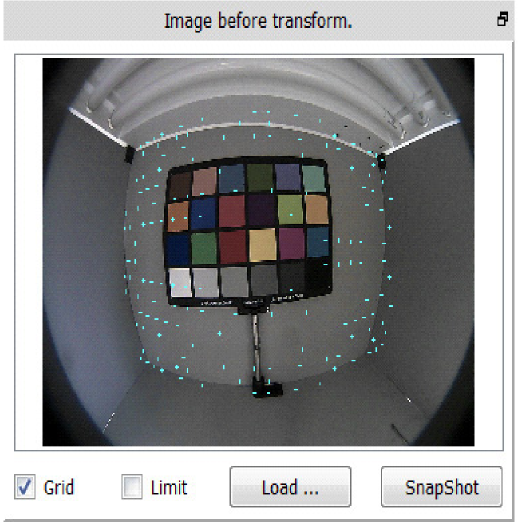


Transform source

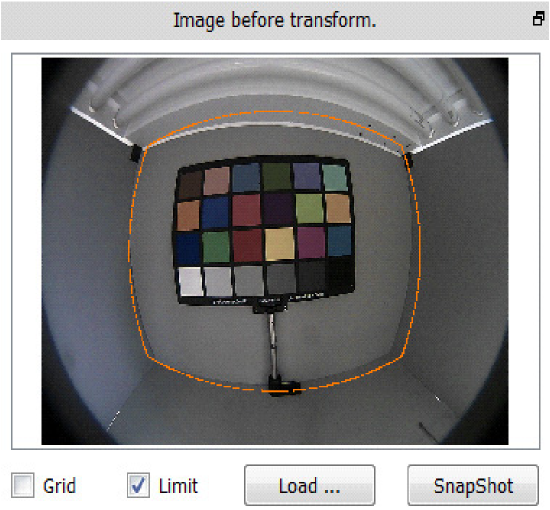
● Generated by 3D geometry

○ Linear stretch

☑ Grid ☑ Limit



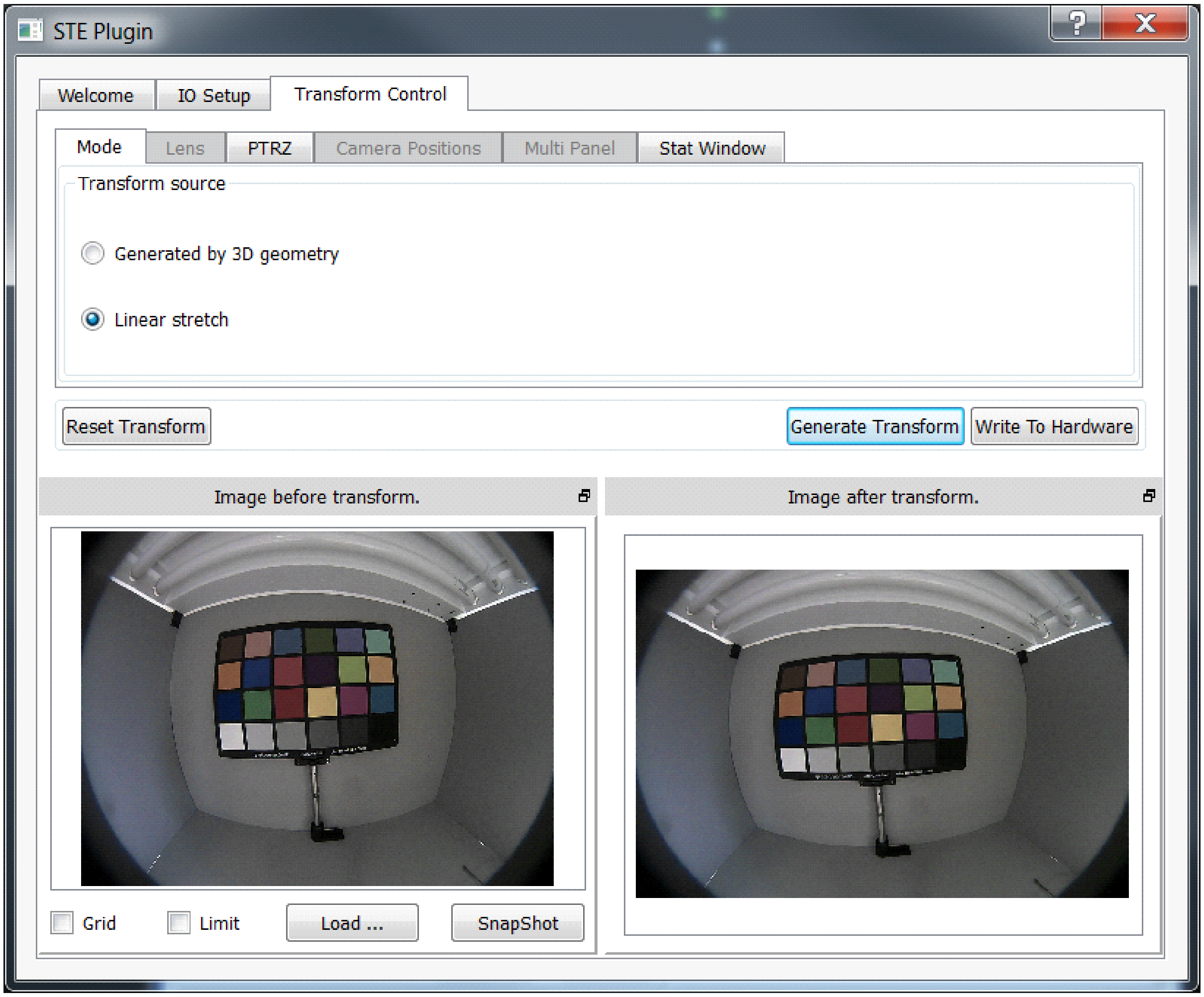
☑ Grid □ Limit



□ Grid ☑ Limit

The cyan dots show the position of every corresponding point in the input picture.The orange line is the limit of the field of view in the output picture.

15. Generate Transform



Transform source

○ Generated by 3D geometry

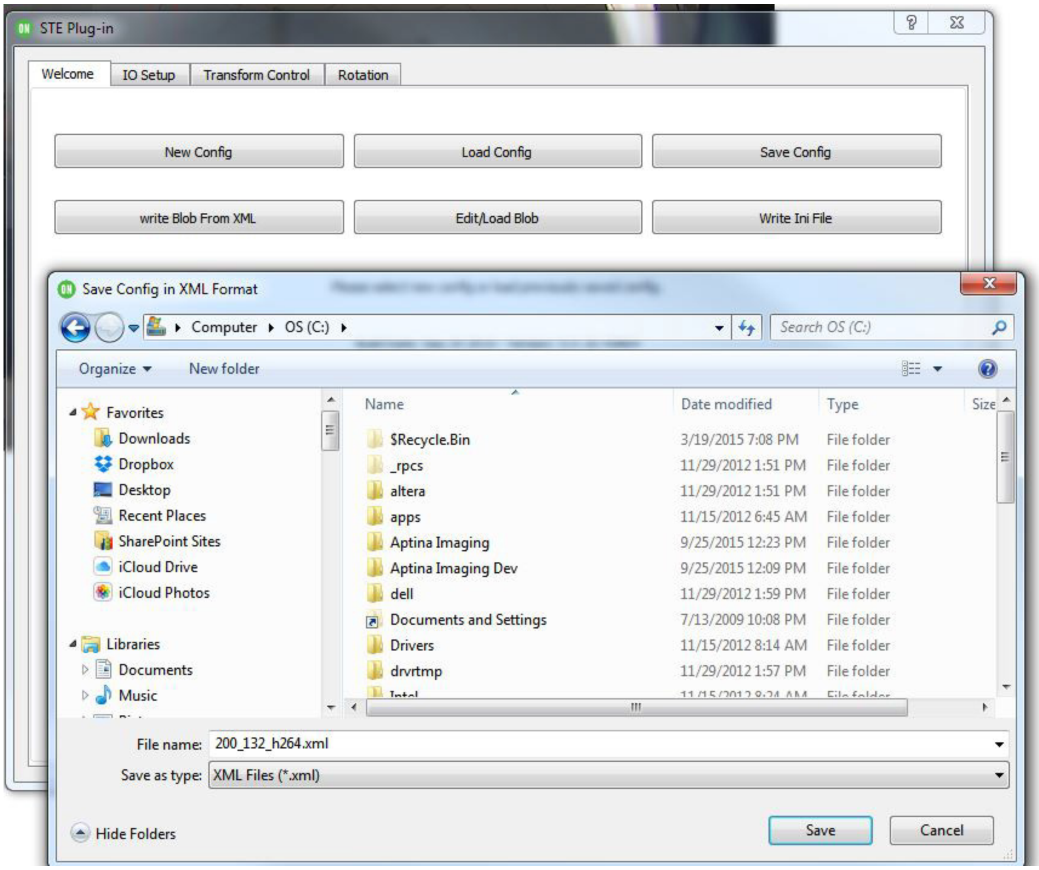
● Linear stretch

□ Grid □ Limit

When "Generate transform" is hit this will compute the transform and produce a simulated output image. Green bar indicates that there is enough STE memory available. Conversely, red means the transform will not work.

"Write to Hardware" will apply the transform to AP0102AT and the viewing image on DevWare (or other source of output like Analog Monitor) should change.

16. Save Config



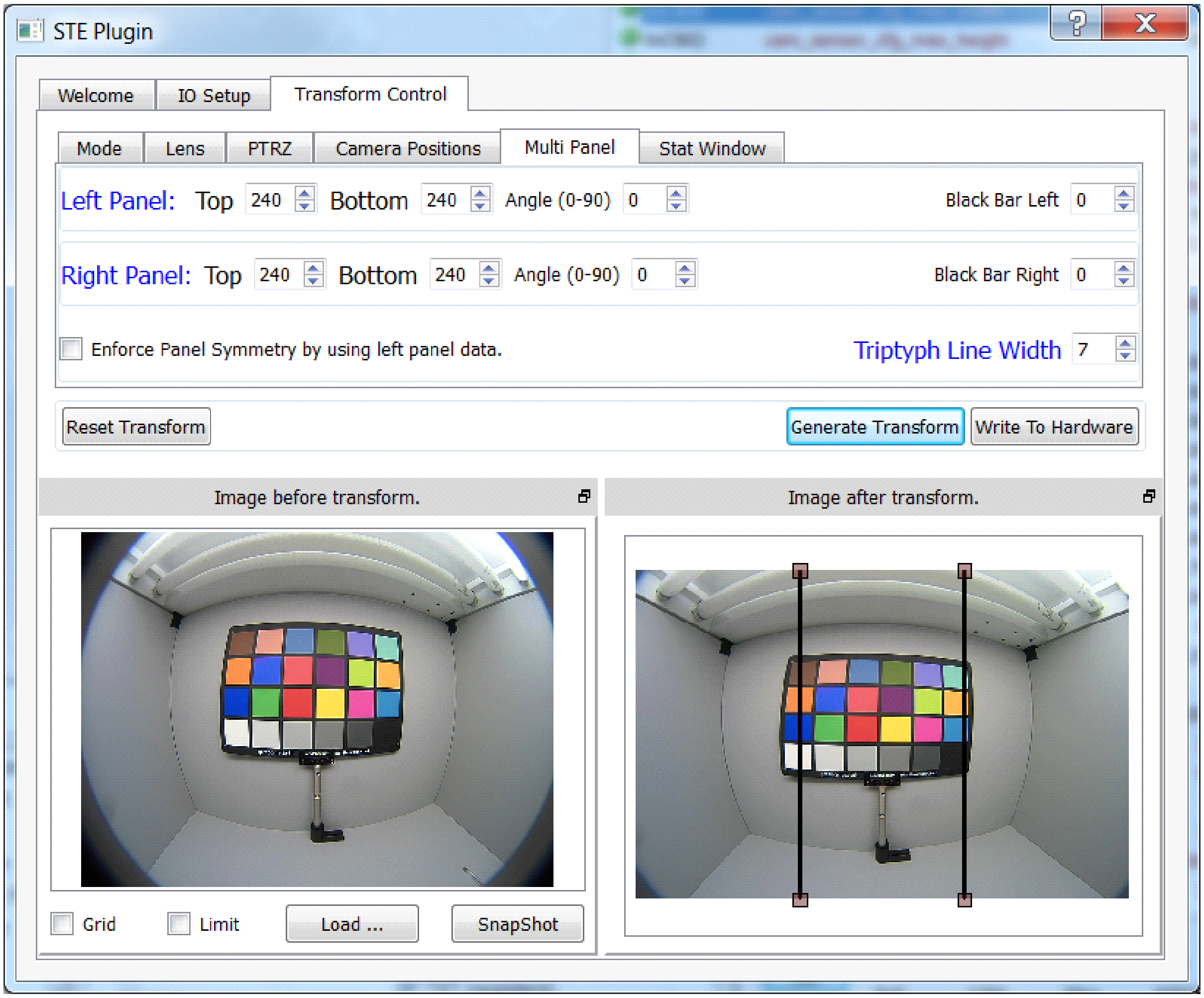
Name	Date modified	Type	Size
$Recycle.Bin	3/19/2015 7:08 PM	File folder	
_rpcs	11/29/2012 1:51 PM	File folder	
altera	11/29/2012 1:51 PM	File folder	
apps	11/15/2012 6:45 AM	File folder	
Aptina Imaging	9/25/2015 12:23 PM	File folder	
Aptina Imaging Dev	9/25/2015 12:09 PM	File folder	
dell	11/29/2012 1:59 PM	File folder	
Documents and Settings	7/13/2009 10:08 PM	File folder	
Drivers	11/15/2012 8:14 AM	File folder	
drvrtmp	11/29/2012 1:57 PM	File folder	

File name: 200_132_h264.xml

Save as type: XML Files (*.xml)

The user can save generated config to file in xml format from “Save Config” option. Similarly config file can be saved in INI format from “Write Ini File” option.

17. Generate Triptych (three−panels view)



Left Panel: Top 240 Bottom 240 Angle (0-90) 0 Black Bar Left 0

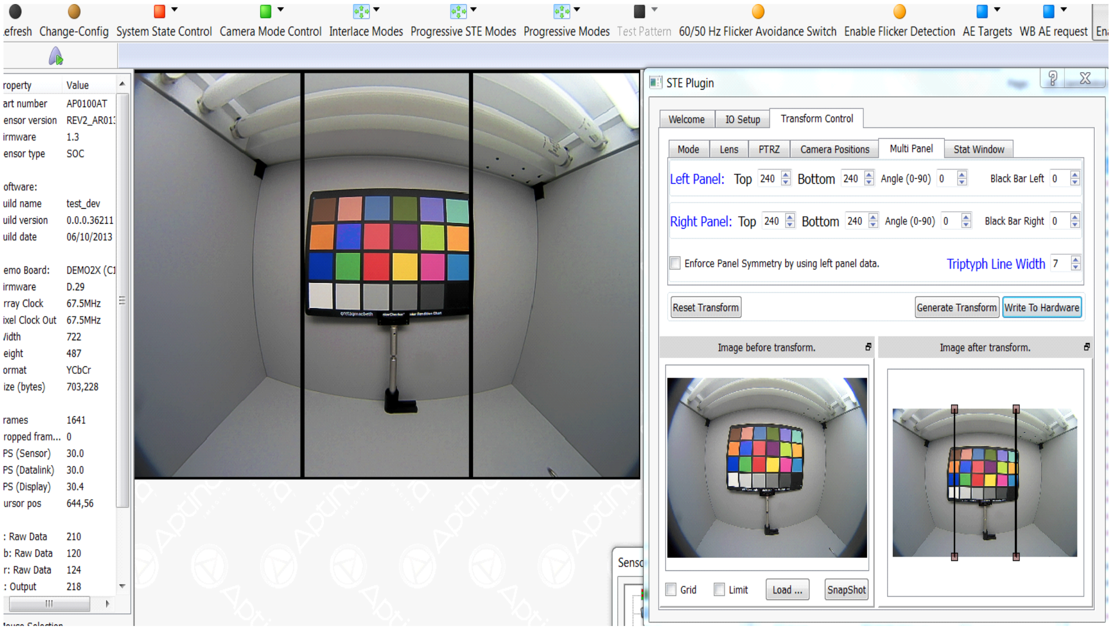
Right Panel: Top 240 Bottom 240 Angle (0-90) 0 Black Bar Right 0

□ Enforce Panel Symmetry by using left panel data.

Triptyph Line Width 7

□ Grid □ Limit

Write to Hardware option shows triptych on DevWare live video.



roperty	Value
art number	AP0100AT
ensor version	REV2_AR013
irmware	1.3
ensor type	SOC
oftware:	
uild name	test_dev
uild version	0.0.0.36211
uild date	06/10/2013
emo Board:	DEMO2X (C1
irmware	D.29
rray Clock	67.5MHz
ixel Clock Out	67.5MHz
/idth	722
eight	487
ormat	YCbCr
ize (bytes)	703,228
rames	1641
ropped fram...	0
PS (Sensor)	30.0
PS (Datalink)	30.0
PS (Display)	30.4
ursor pos	644,56
: Raw Data	210
b: Raw Data	120
r: Raw Data	124
: Output	218

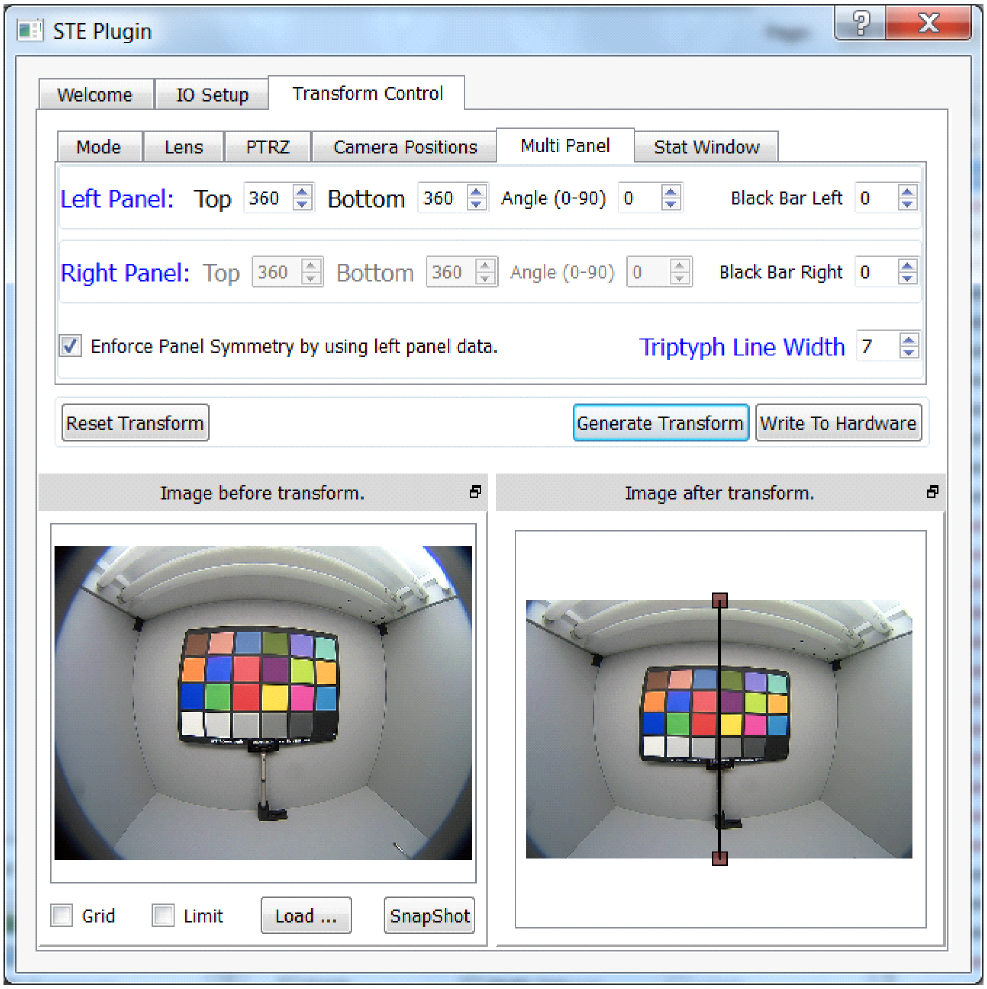
Left Panel: Top 240 Bottom 240 Angle (0-90) 0 Black Bar Left 0

Right Panel: Top 240 Bottom 240 Angle (0-90) 0 Black Bar Right 0

☐ Enforce Panel Symmetry by using left panel data. Triptyph Line Width 7

☐ Grid ☐ Limit

18. Generate Diptych (two−panel view)



Left Panel: Top 360 Bottom 360 Angle (0-90) 0 Black Bar Left 0

Right Panel: Top 360 Bottom 360 Angle (0-90) 0 Black Bar Right 0

☑ Enforce Panel Symmetry by using left panel data. Triptyph Line Width 7

☐ Grid ☐ Limit

STE Plug−in Use Cases

In all of the following examples in Figure 43 below through Figure 47, the camera has NOT physically changed position.

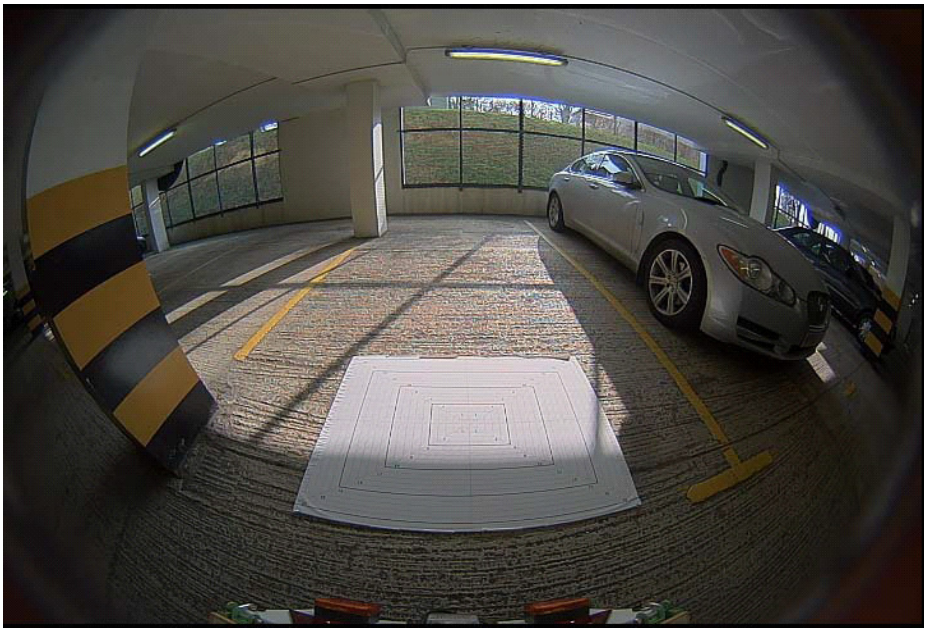


Figure 43. Uncorrected Image

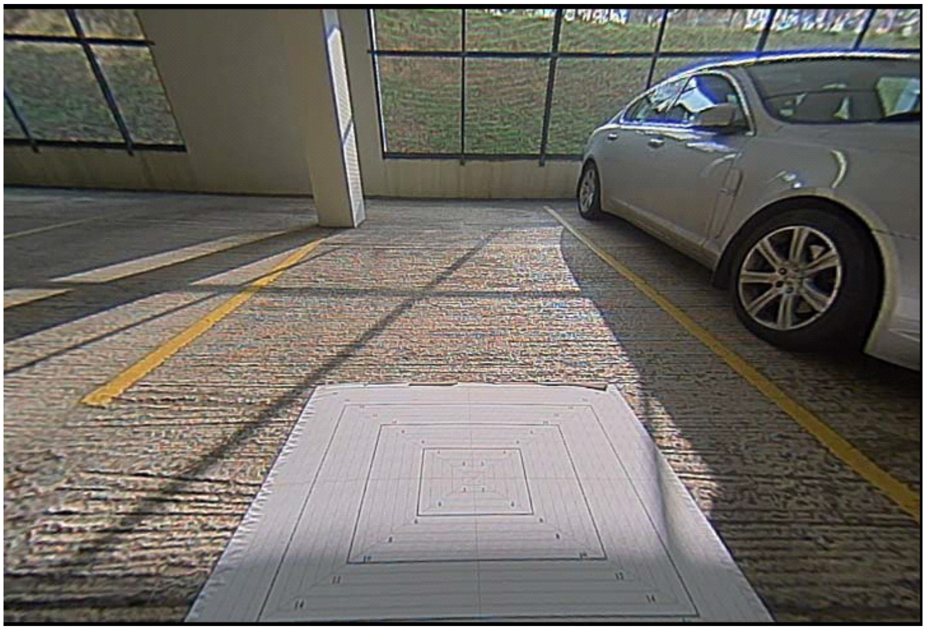


Figure 44. Zoom

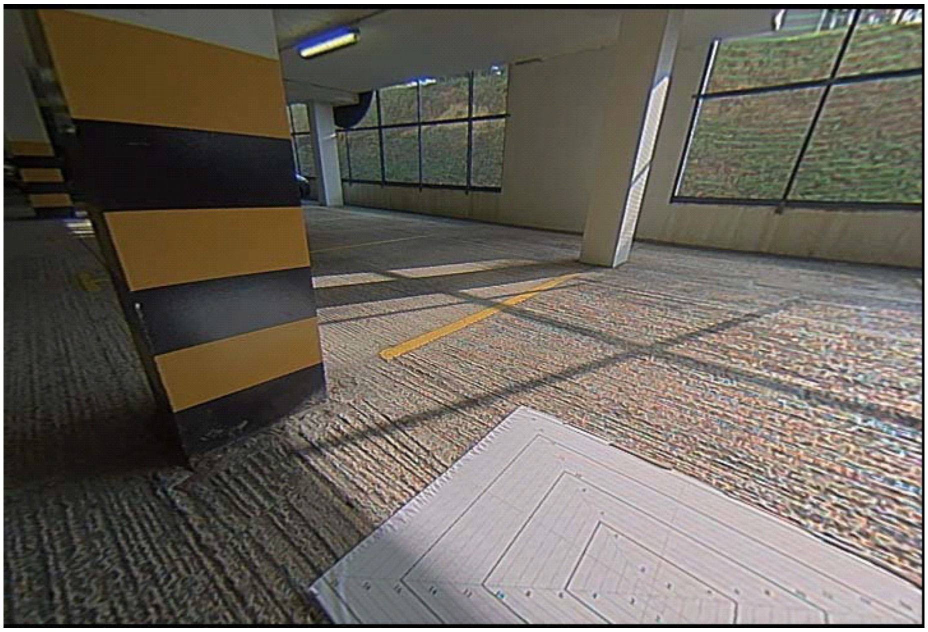


Figure 45. Zoom and Look Left

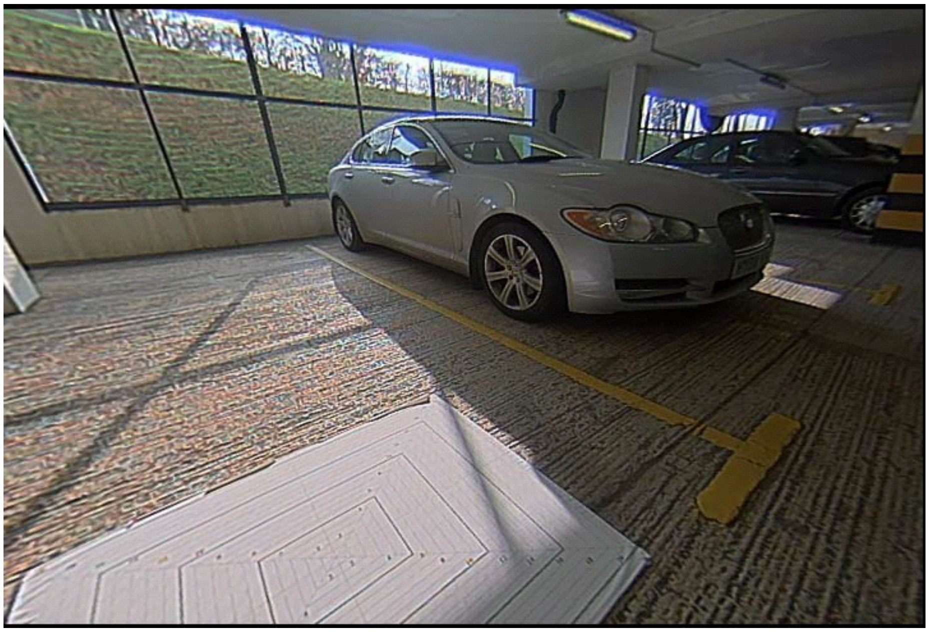


Figure 46. Zoom and Look Right

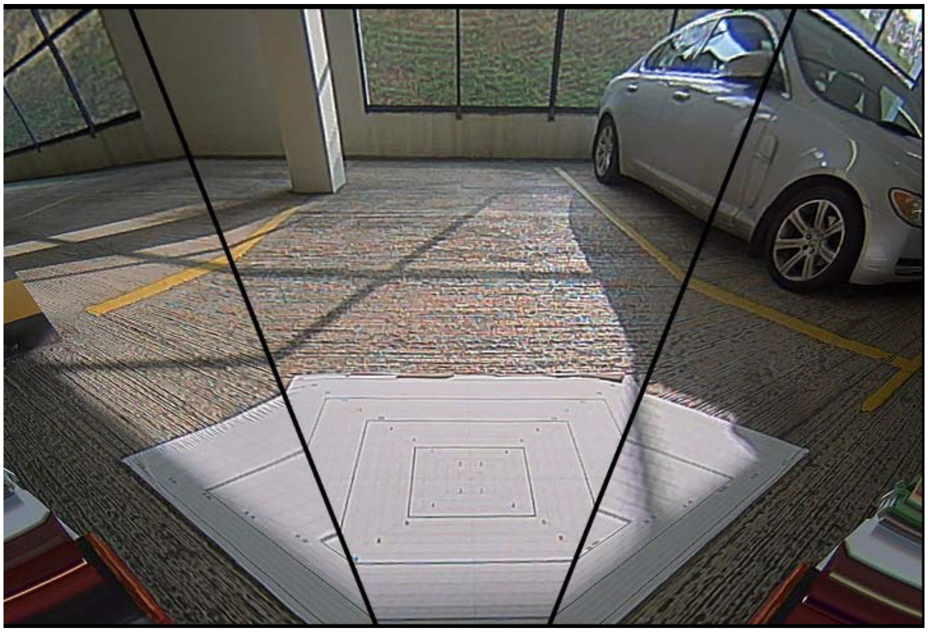
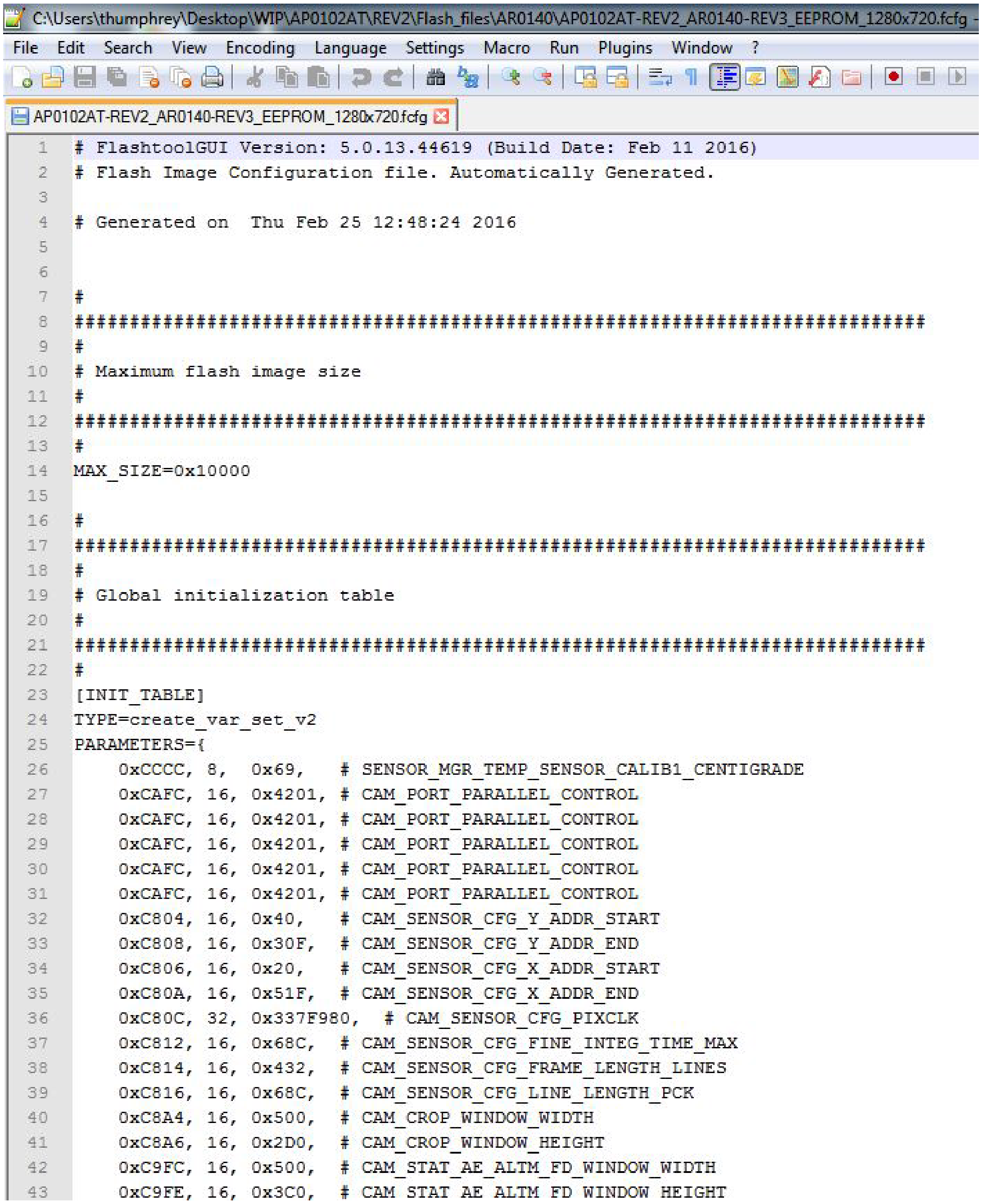


Figure 47. Multi−Panel

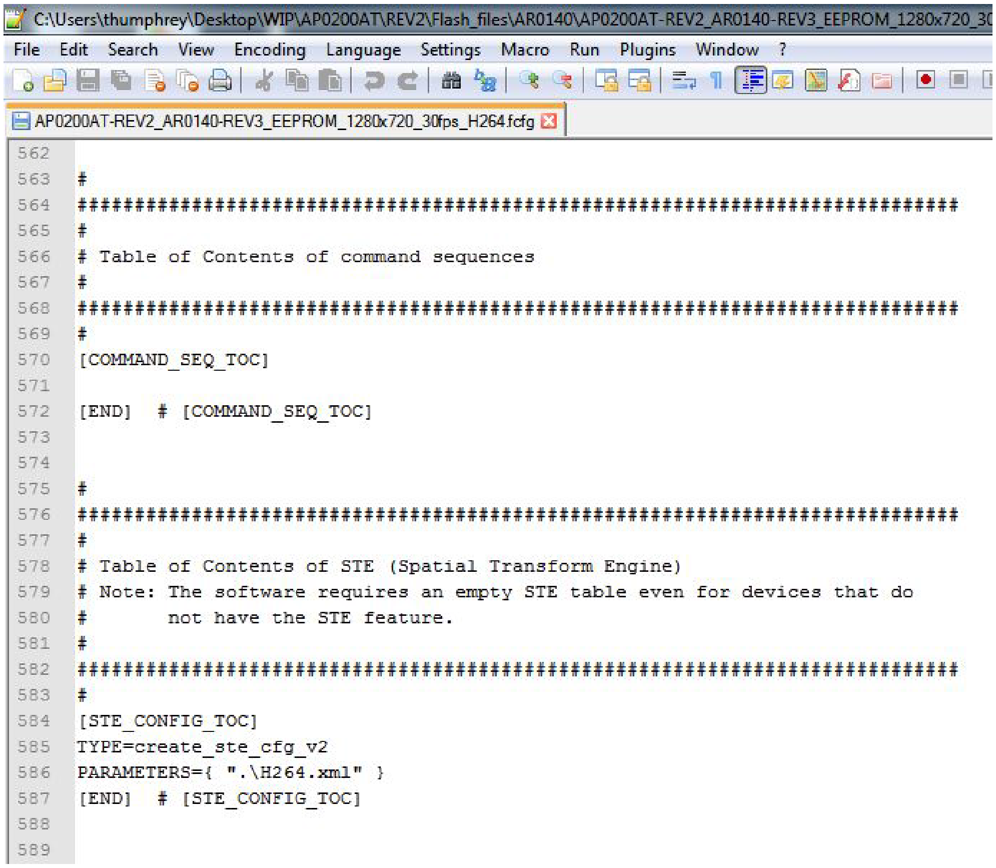
Using STE in Flash

The STE initialization table in the flash configuration file configures the STE subsystem. This flash file is created with Flash Tool and shown as an example for initialization table.


C:\Users\thumphrey\Desktop\WIP\AP0102AT\REV2\Flash_files\AR0140\AP0102AT-REV2_AR0140-REV3_EEPROM_1280x720.fcfg -
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
AP0102AT-REV2_AR0140-REV3_EEPROM_1280x720.fcfg

```
# FlashtoolGUI Version: 5.0.13.44619 (Build Date: Feb 11 2016)
# Flash Image Configuration file. Automatically Generated.

# Generated on  Thu Feb 25 12:48:24 2016


#
################################################################################
#
# Maximum flash image size
#
################################################################################
#
MAX_SIZE=0x10000

#
################################################################################
#
# Global initialization table
#
################################################################################
#
[INIT_TABLE]
TYPE=create_var_set_v2
PARAMETERS={
    0xCCCC, 8,  0x69,    # SENSOR_MGR_TEMP_SENSOR_CALIB1_CENTIGRADE
    0xCAFC, 16, 0x4201, # CAM_PORT_PARALLEL_CONTROL
    0xCAFC, 16, 0x4201, # CAM_PORT_PARALLEL_CONTROL
    0xCAFC, 16, 0x4201, # CAM_PORT_PARALLEL_CONTROL
    0xCAFC, 16, 0x4201, # CAM_PORT_PARALLEL_CONTROL
    0xCAFC, 16, 0x4201, # CAM_PORT_PARALLEL_CONTROL
    0xC804, 16, 0x40,    # CAM_SENSOR_CFG_Y_ADDR_START
    0xC808, 16, 0x30F,   # CAM_SENSOR_CFG_Y_ADDR_END
    0xC806, 16, 0x20,    # CAM_SENSOR_CFG_X_ADDR_START
    0xC80A, 16, 0x51F,   # CAM_SENSOR_CFG_X_ADDR_END
    0xC80C, 32, 0x337F980,  # CAM_SENSOR_CFG_PIXCLK
    0xC812, 16, 0x68C,   # CAM_SENSOR_CFG_FINE_INTEG_TIME_MAX
    0xC814, 16, 0x432,   # CAM_SENSOR_CFG_FRAME_LENGTH_LINES
    0xC816, 16, 0x68C,   # CAM_SENSOR_CFG_LINE_LENGTH_PCK
    0xC8A4, 16, 0x500,   # CAM_CROP_WINDOW_WIDTH
    0xC8A6, 16, 0x2D0,   # CAM_CROP_WINDOW_HEIGHT
    0xC9FC, 16, 0x500,   # CAM_STAT_AE_ALTM_FD_WINDOW_WIDTH
    0xC9FE, 16, 0x3C0,   # CAM_STAT_AE_ALTM_FD_WINDOW_HEIGHT
```

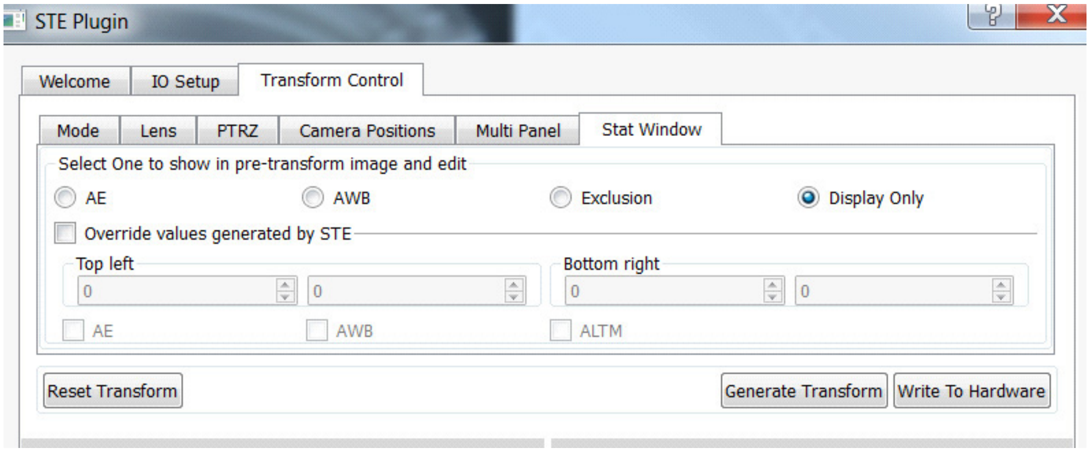
The path and name of the STE configuration xml file go in the *Table of Contents* of the STE section, which is located at the end of the flash configuration file.


C:\Users\thumphrey\Desktop\WIP\AP0200AT\REV2\Flash_files\AR0140\AP0200AT-REV2_AR0140-REV3_EEPROM_1280x720_30
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
AP0200AT-REV2_AR0140-REV3_EEPROM_1280x720_30fps_H264.fcfg

```

#
#############################################################################
#
# Table of Contents of command sequences
#
#############################################################################
#
[COMMAND_SEQ_TOC]

[END]  # [COMMAND_SEQ_TOC]


#
#############################################################################
#
# Table of Contents of STE (Spatial Transform Engine)
# Note: The software requires an empty STE table even for devices that do
#       not have the STE feature.
#
#############################################################################
#
[STE_CONFIG_TOC]
TYPE=create_ste_cfg_v2
PARAMETERS={ ".\H264.xml" }
[END]  # [STE_CONFIG_TOC]


```

STE STATISTICS

The following section explains how the AP0102AT statistics (STATS) component is configured when the Spatial Transform Engine (STE) is enabled.



Select One to show in pre-transform image and edit

○ AE ○ AWB ○ Exclusion ● Display Only

□ Override values generated by STE

Top left: 0, 0

Bottom right: 0, 0

□ AE □ AWB □ ALTM

Overview

The AP0102AT Statistics (STATS) component and its hardware driver are responsible for the gathering and partial processing of the scene statistics. The collection of the scene statistics is controlled by three primary windows; the AE/ALTM/FD acquisition window, the AWB/CLIP acquisition window and the Exclusion window.

The acquisition windows can be auto−calculated by the AP0102AT or set by the user relative to either the sensor window or the output window via the camera control variables or STE Plug−in.

The exclusion window provides the user with the ability to exclude a rectangular region of the scene from the statistics acquisition (for example, a car bumper that may reflect the sun into the back−up camera lens). The user specifies the exclusion window relative to the full FOV calibrated FOV of the sensor since this represents a fixed point of reference in the calibrated scene.

The user may configure the ‘stats window’ for each hardware collection engine. This controls which pixels in the scene are processed by the hardware.

AE/ALTM/FD Window

The AE/ALTM/FD acquisition window defines the region of the scene where the AE, ALTM and FD metrics are collected.

Auto Exposure Statistics

For the Auto Exposure statistics, STATS splits the acquisition window into a 5x5 matrix of 25 equal area zones. For each zone, STATS collates the average brightness, or luma (Y), average log2(Y) and the number of samples. The 25 zone based values are used by STATS to compute the full scene average Y, average log2(Y) and log2(average Y) values for use by other components.

Refer to the Auto Exposure section for more details.

ALTM Statistics

The ALTM statistics are hardware generated frame based metrics. During a config change or refresh, STATS programs the associated hardware with the acquisition window and exclusion window co−ordinates. When enabled, the STATS component reads the ALTM statistics at the start of each vertical blanking period.

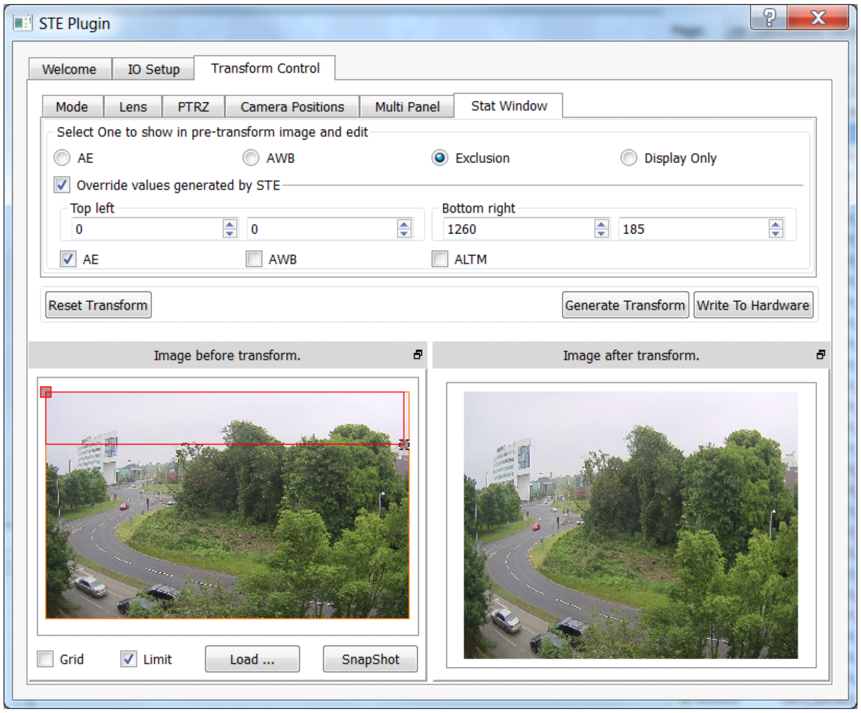
AWB/Clip Window

The AWB/CLIP acquisition window defines the region of the scene where the AWB and CCM−clip counter statistics are collected. On config change or refresh, the STATS component programs the associated hardware with the acquisition window and exclusion window co−ordinates. When enabled, the STATS component reads the frame based metrics at the start of each vertical blanking period.

STE Plug−in Stat Window GUI

The example below shows how to exclude sky from the scene which might affect ALTM, AWB or AE.

1. Exclusion Window Example for AE



Select One to show in pre-transform image and edit

○ AE ○ AWB ● Exclusion ○ Display Only

☑ Override values generated by STE

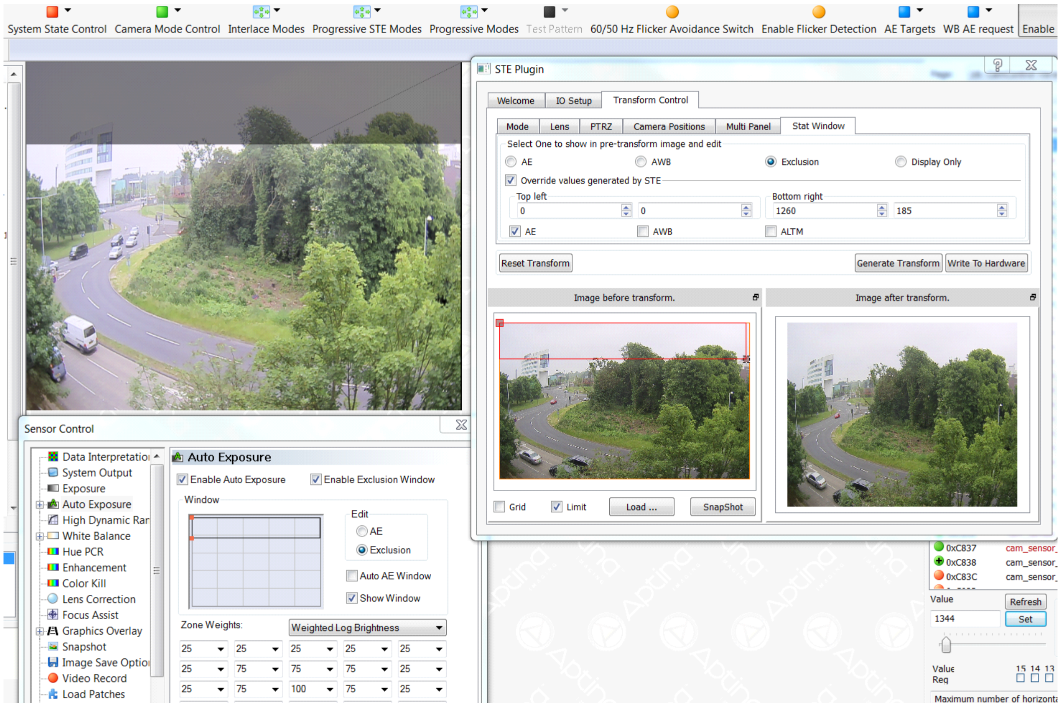
Top left: 0, 0

Bottom right: 1260, 185

☑ AE □ AWB □ ALTM

□ Grid ☑ Limit

Press “Write to Hardware” and enable AE exclusion window in DevWare to look at selected /excluded window on the live video.



Auto Exposure

☑ Enable Auto Exposure ☑ Enable Exclusion Window

Window

Edit: ○ AE ● Exclusion

□ Auto AE Window

☑ Show Window

Zone Weights: Weighted Log Brightness

25	25	25	25	25
25	75	75	75	25
25	75	100	75	25

Select One to show in pre-transform image and edit

○ AE ○ AWB ● Exclusion ○ Display Only

☑ Override values generated by STE

Top left: 0, 0

Bottom right: 1260, 185

☑ AE □ AWB □ ALTM

□ Grid ☑ Limit

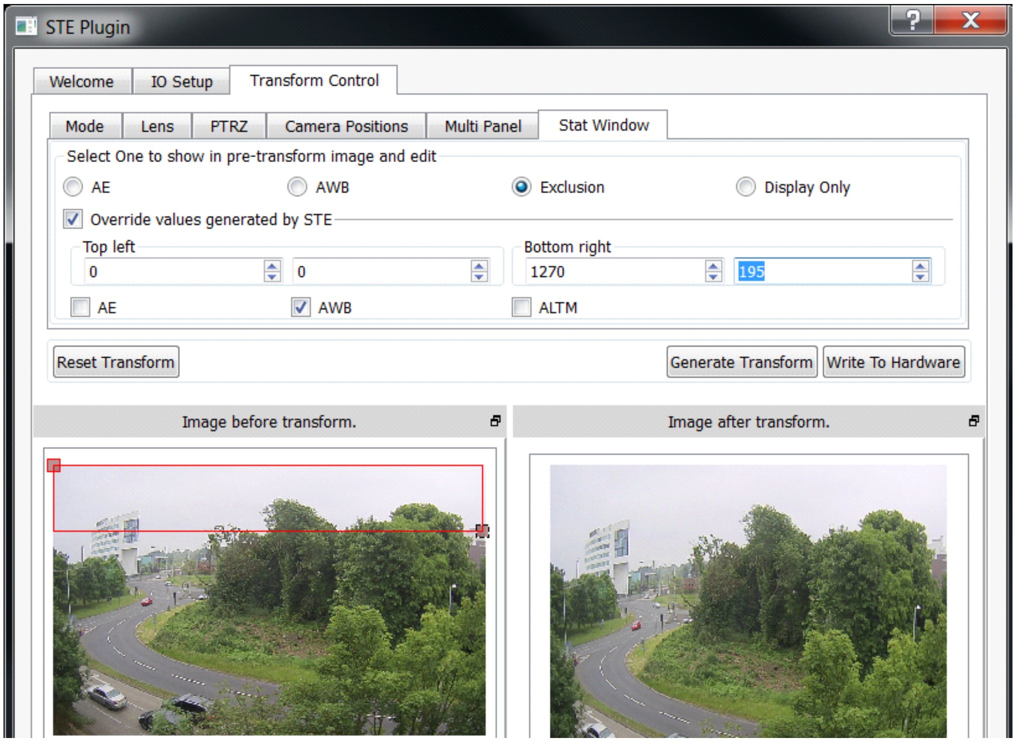
0xC837 cam_sensor_
0xC838 cam_sensor_
0xC83C cam_sensor_

Value 1344

Value Req 15 14 13

Maximum number of horizont

2. Exclusion Window Example for AWB



STE Plugin

Welcome | IO Setup | Transform Control

Mode | Lens | PTRZ | Camera Positions | Multi Panel | Stat Window

Select One to show in pre-transform image and edit

○ AE ○ AWB ● Exclusion ○ Display Only

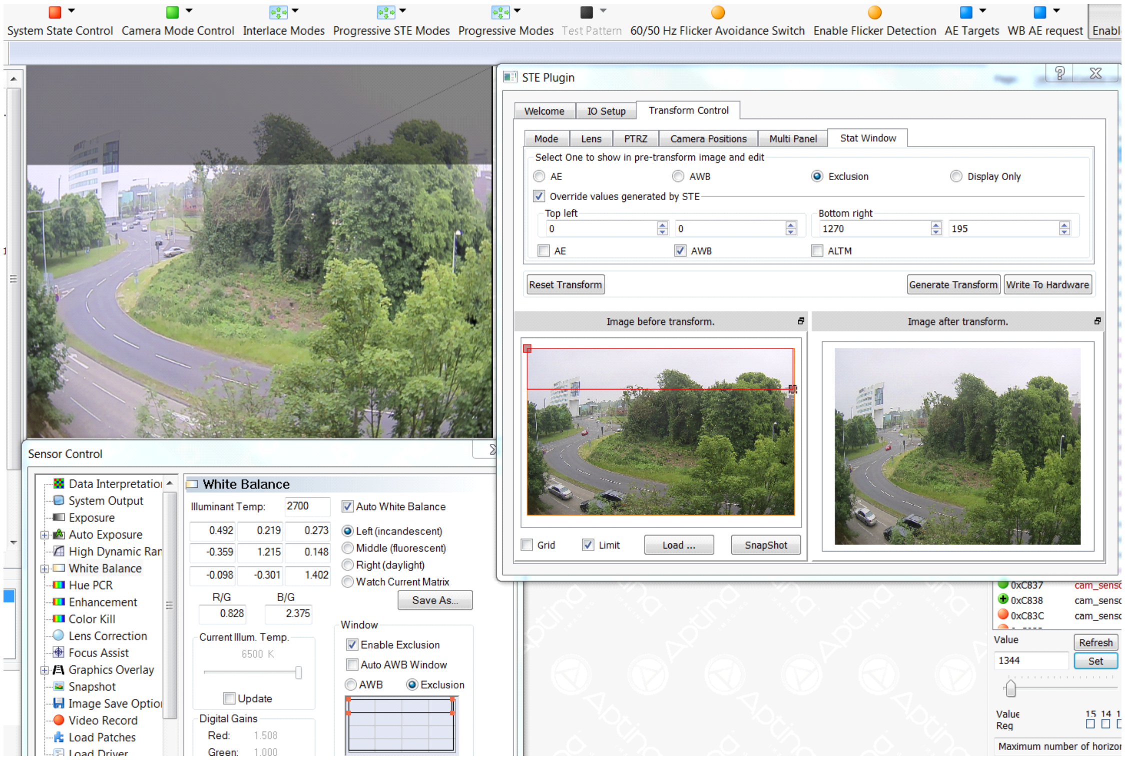
☑ Override values generated by STE

Top left: 0, 0

Bottom right: 1270, 195

□ AE ☑ AWB □ ALTM

Press "Write to Hardware" and enable AWB exclusion window in DevWare to look at selected /excluded window on the live video.



STE Plugin

Welcome | IO Setup | Transform Control

Mode | Lens | PTRZ | Camera Positions | Multi Panel | Stat Window

Select One to show in pre-transform image and edit

○ AE ○ AWB ● Exclusion ○ Display Only

☑ Override values generated by STE

Top left: 0, 0

Bottom right: 1270, 195

□ AE ☑ AWB □ ALTM

Sensor Control

White Balance

Illuminant Temp: 2700

0.492	0.219	0.273
-0.359	1.215	0.148
-0.098	-0.301	1.402

R/G: 0.828 B/G: 2.375

Current Illum. Temp.: 6500 K

□ Update

Digital Gains
Red: 1.508
Green: 1.000

☑ Auto White Balance

● Left (incandescent)
○ Middle (fluorescent)
○ Right (daylight)
○ Watch Current Matrix

Window

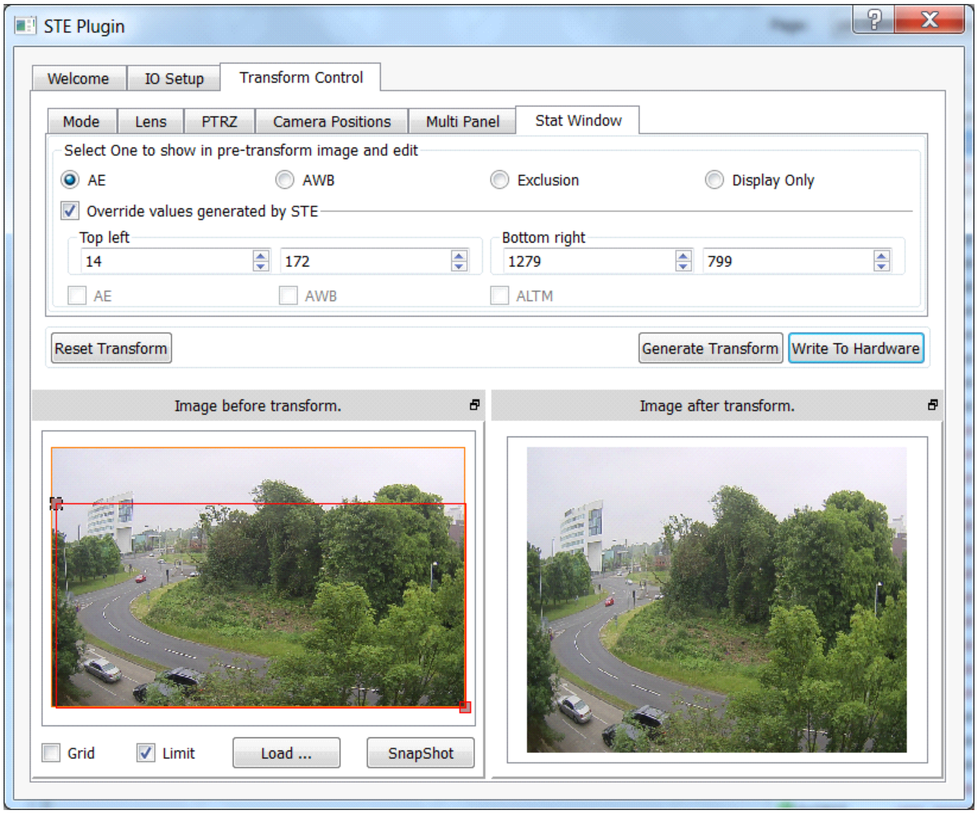
☑ Enable Exclusion
□ Auto AWB Window
○ AWB ● Exclusion

0xC837	cam_sensc
0xC838	cam_sensc
0xC83C	cam_sensc

Value: 1344

Maximum number of horizor

3. AE Window Example



Select One to show in pre-transform image and edit

● AE ○ AWB ○ Exclusion ○ Display Only

☑ Override values generated by STE

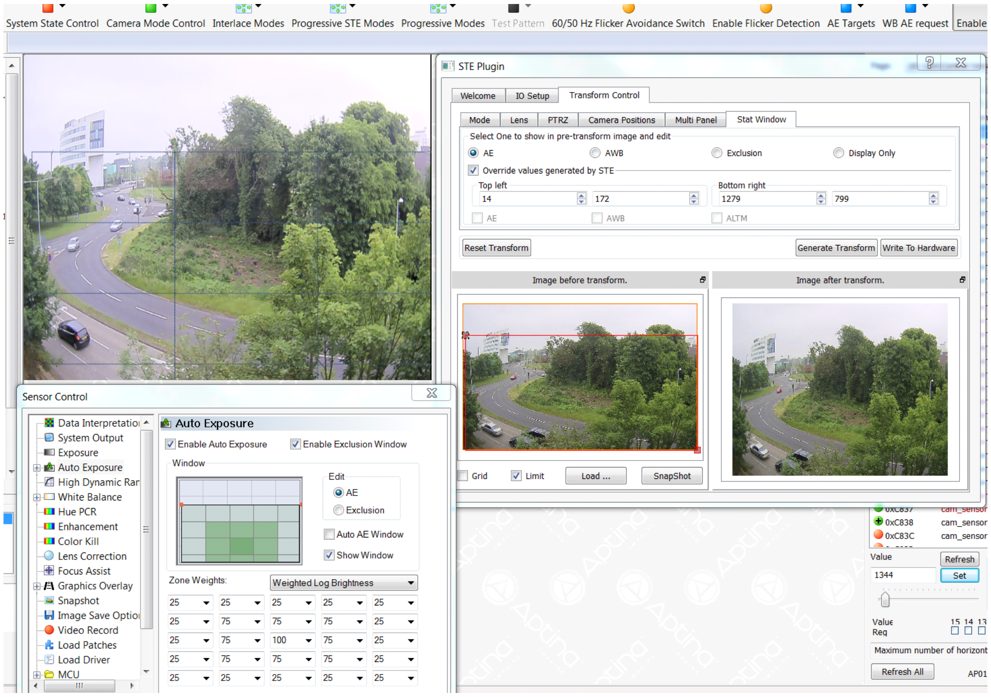
Top left: 14, 172

Bottom right: 1279, 799

□ AE □ AWB □ ALTM

□ Grid ☑ Limit

Press “Write to Hardware” and enable AE exclusion window in DevWare to look at selected AWB window on the live video



Select One to show in pre-transform image and edit

● AE ○ AWB ○ Exclusion ○ Display Only

☑ Override values generated by STE

Top left: 14, 172

Bottom right: 1279, 799

□ AE □ AWB □ ALTM

□ Grid ☑ Limit

Auto Exposure

☑ Enable Auto Exposure ☑ Enable Exclusion Window

Window

Edit: ● AE ○ Exclusion

□ Auto AE Window

☑ Show Window

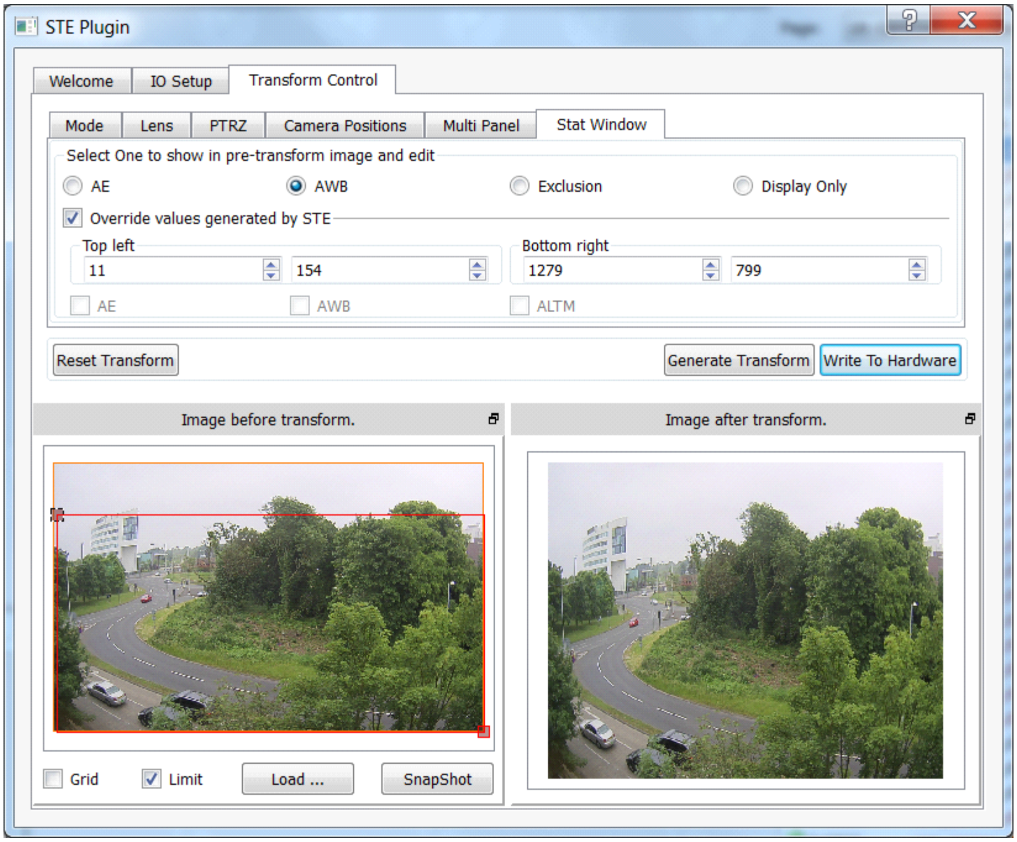
Zone Weights: Weighted Log Brightness

25	25	25	25	25
25	75	75	75	25
25	75	100	75	25
25	75	75	75	25
25	25	25	25	25

Value: 1344

Value 15 14 13

4. AWB Window Example



Select One to show in pre-transform image and edit

○ AE ● AWB ○ Exclusion ○ Display Only

☑ Override values generated by STE

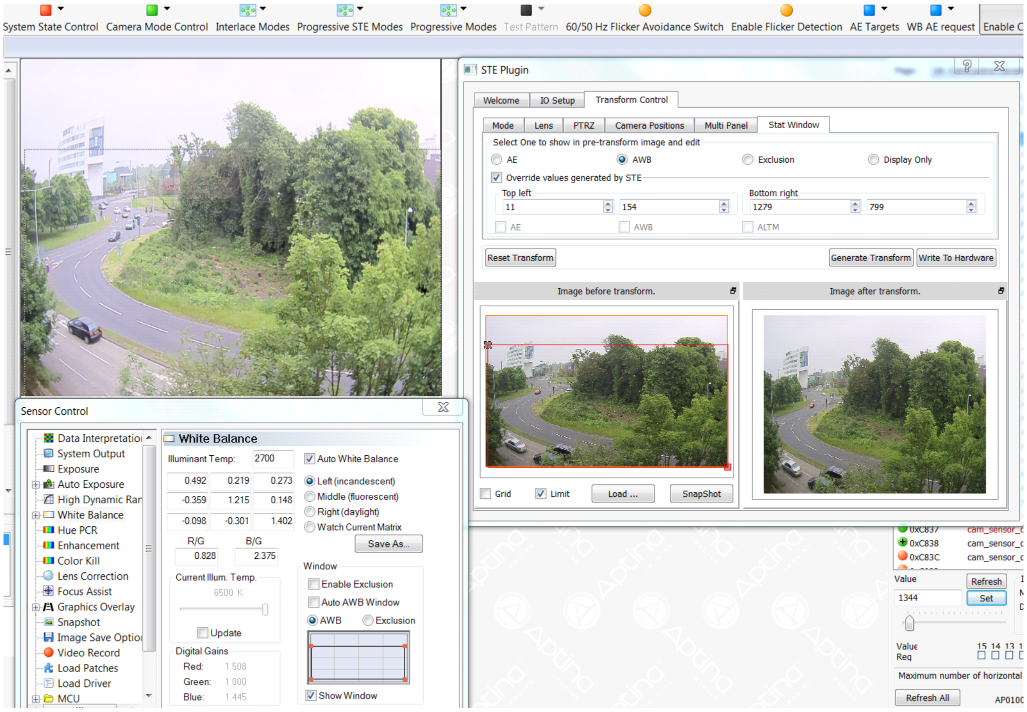
Top left: 11, 154

Bottom right: 1279, 799

□ AE □ AWB □ ALTM

□ Grid ☑ Limit

Press "Write to Hardware" and enable AWB exclusion window in DevWare to look at selected AWB window on the live video



STE Plugin: Select One to show in pre-transform image and edit: ○ AE ● AWB ○ Exclusion ○ Display Only; ☑ Override values generated by STE; Top left 11, 154; Bottom right 1279, 799; □ AE □ AWB □ ALTM; □ Grid ☑ Limit

White Balance

Illuminant Temp: 2700

☑ Auto White Balance

0.492	0.219	0.273
-0.359	1.215	0.148
-0.098	-0.301	1.402

● Left (incandescent)
○ Middle (fluorescent)
○ Right (daylight)
○ Watch Current Matrix

R/G	B/G
0.828	2.375

Current Illum. Temp.: 6500 K

□ Update

Digital Gains

Red: 1.508
Green: 1.000
Blue: 1.445

Window

□ Enable Exclusion
□ Auto AWB Window
● AWB ○ Exclusion
☑ Show Window

0xC837 cam_sensor_c
0xC838 cam_sensor_c
0xC83C cam_sensor_c

Value: 1344

Value Req 15 14 13 1

Maximum number of horizontal

APPENDIX A – GLOSSARY OF TERMS

Table 41. ABBREVIATIONS

Term	Description
AE	Auto Exposure
ALTM	Adaptive Local Tone Mapping
algo	Algorithm
AWB	Auto White Balance
CC	Companion Chip
CCI	Camera Control Interface
CCIM	Camera Control Interface Master
CCM	Color Correction Matrix
CIE	International Commission on Illumination (usually abbreviated CIE for its French name, Commission internationale de l'éclairage)
CWF	Cool White Fluorescent
DAC	Digital Analog Conversion
DC	Defect Correction
DCG	Dual Conversion Gain
DCNR	Defect Correction & Noise Reduction
DLO	Digital Lateral Overflow
DMA	Direct Memory Access
DSP	Digital Signal Processor
EEPROM	Electrically Erasable Programmable Read−Only Memory
EOF	End Of Frame
EV	Exposure Value
FAE	Field Application Engineer
FCFG	Flash Configuration
FD	Flicker Detect
FOV	Field Of View
fps	Frame Per Second
GND	Ground
GPIO	General Purpose Input Output
GUI	Graphical User Interface
HCI	Host Command Interface
HDR	High Dynamic Range
HiSPi	High Speed Serial Pixel Interface
I^2C	Two−wire serial interface
IO	Input Output
IR	Infra−Red
IRCF	IR Cut−off Filter
IRE	IRE (Institute of Radio Engineers) is a means of measuring brightness as a relative percentage of total brightness
ISP	Image Signal Processor
JEDEC	Joint Electron Device Engineering Council
KB	Kilobytes

Table 41. ABBREVIATIONS (continued)

Term	Description
LCD	Liquid Crystal Display
LSB	Least Significant Bit
MB	Megabytes
MSB	Most Significant Bit
MC	Motion Compensation
NVM	Non Volatile Memory
PCB	Printed Circuit Board
PCR	Preferred Color Reproduction
PLL	Phase Locked Loop
OTPM	One Time Programmable Memory
PGA	Positional Gain Adjustment
PLL	Phase Lock Loop
RMS	Root Mean Square
SDR	Standard Dynamic Range
SNR	Signal to Noise Ratio
SOF	Start of Frame
SPI	Serial Peripheral Interface
STE	Spatial Transform Engine
XSDAT	XML Sensor Data File
XML	Extensible Markup Language

PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:
Literature Distribution Center for ON Semiconductor
19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA
Phone: 303−675−2175 or 800−344−3860 Toll Free USA/Canada
Fax: 303−675−2176 or 800−344−3867 Toll Free USA/Canada
Email: orderlit@onsemi.com

N. American Technical Support: 800−282−9855 Toll Free USA/Canada
Europe, Middle East and Africa Technical Support:
Phone: 421 33 790 2910

ON Semiconductor Website: **www.onsemi.com**

Order Literature: http://www.onsemi.com/orderlit

For additional information, please contact your local Sales Representative