# Application Report **TDA4VMid VPAC ISP Tuning Overview**

# TEXAS INSTRUMENTS

# ABSTRACT

This application report explains the high-level flow of tuning tool usage and transferring the tuned parameters to the SDK.

# **Table of Contents**

1 Introduction	2
2 Getting Started	2
2.1 Hardware Requirements	2
2.2 Software Requirements	2
2.3 Lab Requirements	2
3 IQ Tuning Prerequisites	
3.1 Requirements for Tuning on Target	
3.2 Requirements for Tuning on Simulator	
4 VPAC Overview	
4.1 Block Diagram	
4.2 IP Overview	
5 Dataflow	6
6 IQ Tuning Process	6
6.1 Expectations of the Tool	
7 Imaging Software Architecture	7
7.1 Auto White Balance	7
7.2 AutoExposure	7
7.3 Dynamic Camera Configuration (DCC) Algorithm	7
8 Computing Tuning Parameters	7
9 Testing Tuning Parameters	
9.1 On Target Platform – Compile Time Update	
9.2 On Target Platform – Run Time Update From File System	9
10 Live Tuning	9
10.1 Supported Features	9
11 Revision History	

# **List of Figures**

Figure 4-1. VPAC Logical Block Diagram	3
Figure 4-2. VISS Logical Block Diagram	4
Figure 5-1. VISS Dataflow	6
Figure 10-1. IP Address Entry in Live Tuning Bar	9
Figure 10-2. RAW File Save Button in Live Tuning Bar	9
Figure 10-3. YUV File Save Button in Live Tuning Bar	9
Figure 10-4. Exposure Control (Auto Mode) in Live Tuning Window	10
Figure 10-5. Exposure Control (Manual Mode) in Live Tuning Window	. 10
Figure 10-6. White Balance Control (Auto) in Live Tuning Window	11
Figure 10-7. White Balance Control (Manual) in Live Tuning Window	11
Figure 10-8. Sensor Register Read/Write Control in Live Tuning Window	. 12
Figure 10-9. Current Plugin DCC Update Control in Live Tuning Window	. 12
Figure 10-10. Multi Plugin DCC Update Control in Live Tuning Window	12



# Trademarks

All trademarks are the property of their respective owners.

# 1 Introduction

The camera imager is a sensor that converts light into a digital signal, which can be used to represent an image. However, the image generated by an image sensor is in a format not suitable to be consumed directly by humans or machines. The sensor output has many defects that must be fixed to make the image visually appealing or compatible with computer vision or deep-learning systems. This is achieved by an Image Signal Processor (ISP).

Jacinto 7 processors come equipped with hardware accelerated ISP called Vision Preprocessing ACcelerator (VPAC). VPAC is responsible for end-to-end image processing and is designed to be software configurable to support a wide range of image sensors.

Every image sensor has a different response to light. Therefore, the VPAC parameters must be computed uniquely for every sensor. This process is called Image Quality (IQ) tuning. Texas Instruments (TI) provides a PC based tool that allows customers to accomplish IQ tuning with a user friendly interface.

Tuned parameters thus computed must be written to VPAC Memory Mapped Registers (MMRs). This is achieved through changing the Dynamic Camera Configuration (DCC) software library in imaging component of J7 Software Development Kit https://www.ti.com/tool/download/PROCESSOR-SDK-RTOS-J721E.

# **2 Getting Started**

# 2.1 Hardware Requirements

- Jacinto 7 EVM and all the related accessories + daughterboards
- Camera module to be tuned
- Ubuntu 18.04 PC to build PROCESSOR SDK RTOS J721E
- Windows 10 PC for DCC Tuning Tool
- Micro-SD Card & Reader
- Compatible monitor for live preview

# 2.2 Software Requirements

- PSDK RTOS release (v7.1 or later)
- DCC Tuning Tool (v2.03 or later)

# 2.3 Lab Requirements

- Lightbox
  - With calibrated light sources, ranging from 2700K 6500K (at least)
  - More light sources can be included if wider range of color temperatures is required
- XRite Color Checker Chart
- Different chart sizes maybe needed depending on the lab setup
- Chroma meter
- Tripods or mounts as needed to hold the camera securely
- Ethernet router and cables



# **3 IQ Tuning Prerequisites**

# 3.1 Requirements for Tuning on Target

- PROCESSOR SDK RTOS J721E built in TARGET mode and firmware flashed on the micro-SD card. For more information, see the Processor SDK Getting Started Page.
- Sensor driver done and integrated with the SDK
- Camera demo app (for example, vx\_app\_single\_cam) fully functional
- Live Preview
  - Ability to save RAW and YUV images. To know more about RAW image requirements, see the plugin guides in the tuning tool documentation.
  - Ability to get IP address and get pinged from a Windows PC
  - Sensor data sheet and specs like WDR companding parameters must be available.
- Auto Exposure must be functional, or the imager must be controllable in manual mode.
- It is critical to get well exposed images for calibration. Under-exposure, over-exposure or excessive noise can result in poor tuning results.
- Sensor driver must ensure that gain/integration time is correctly translated to sensor registers.

# 3.2 Requirements for Tuning on Simulator

- PSDK RTOS built in TARGET mode
- Ability to run tiovx conformance test on the Ubuntu PC
- RAW images to be used for tuning. To know more about RAW image requirements, see the plugin guides in the tuning tool documentation.
- Sensor data sheet and specs like WDR companding parameters must be available.
- Lens data sheet and specs like F-number and distortion model must be available.

# 4 VPAC Overview

# 4.1 Block Diagram



Figure 4-1. VPAC Logical Block Diagram

# 4.2 IP Overview

# 4.2.1 Lens Distortion Correction (LDC)

# 4.2.1.1 Features

- Mesh based distortion correction. Uses back mapping LUT
- Formats Supported
  - YUV422
  - YUV420
  - Up to 12 bit input and output
- Bicubic and Bilinear Interpolation



- Programmable mesh table
- Support extendible to almost every lens, including Fish Eye Lenses

# 4.2.2 Bilateral Noise Filter (BNF)

## 4.2.2.1 Features

- YUV domain noise filter
- Two modes of operation:
  - Bilateral
  - Generic

# 4.2.3 Multi Scalar (MSC)

# Features:

- 10 programmable resizers
- Multi-thread/Multi-scaling operations (up to 2 inputs and 10 outputs)
- Scaling ratio between 1x and 0.25x

# 4.2.4 Vision Imaging Subsystem (VISS)



# Figure 4-2. VISS Logical Block Diagram

#### 4.2.5 Subblocks

# 4.2.5.1 Raw Front End (RAWFE)

Includes HW accelerators for the following functions:

#### 4.2.5.1.1 WDR Decompanding

Supports Piece Wise Linear (PWL) and Look Up Table (LUT) based decompanding.

Black Level Subtraction can be done as part of decompanding, if LUT method is chosen.

#### 4.2.5.1.2 WDR Merge

Support for up to 3-exposure merge. Not currently supported by Tuning Tool.

# 4.2.5.1.3 Defect Pixel Correction (DPC)

On The Fly (OTF) and Look Up Table (LUT)-based defect pixel correction.

Eight correction methods supported by LUT based correction.

# 4.2.5.1.4 Lens Shading Correction (LSC)

Corrects for the lens fall of light at the corners of the lens.

Programmable gain map. MxN down-sampled to fit in internal memory

Independent gain maps for 4 color channels.

#### 4.2.5.1.5 Hardware 3A (H3A)

Generates statistics like sum/min/max/sum\_of\_sq on a given input image

• Tuning Tool supports sum mode only. min, max and sum\_of\_sq not supported.

Statistics are used by algorithms like Auto Exposure, Auto White Balance.

Frame is divided into MxN sub-windows and statistics are calculated for each sub-window.

Configurable parameters for ROI selection, and so forth to enable modes like center-metering

#### 4.2.5.1.6 White Balance (WB)

Applies WB gains computed by AWB algorithm.

If NSF4 is enabled, WB gains will be applied in NAF4 and gains here must be set to unity.

#### 4.2.5.2 NSF4 – 4th Generation Noise Filter

Raw domain spatial noise filter supporting generic 2x2 raw color patterns and adaptive noise level thresholds.

#### 4.2.5.3 Global and Local Brightness Contrast Enhancement (GLBCE)

Based on iridix7

Raw format

12-bit to 16-bit input, 16-bit output

If the bit-depth of raw data is higher than 16, decompanding LUT must be generated in a way that the image is global tone mapped to 16-bits. Tuning tool helps with that.

# 4.2.5.4 Flexible Color Processing (FCP)

- Flexible CFA: Accepts any 2x2 CFA as input and generates up to 4 full resolution color planes
- · Flexible Color Conversion: Includes sub-blocks for:
  - CCM Color Correction Matrix
  - Gamma
  - Color space conversion
    - RGB to YUV
    - RGB to Saturation
- Chroma down-sampling





#### Figure 5-1. VISS Dataflow

For most use cases, the data flow of VISS is as following.

- 1. RAWFE decompands the sensor raw image, removes the black level, and recompresses the companded WDR raw sensor images into 16-bit gamma corrected format with a LUT.
- 2. RAWFE may apply DPC and LSC, if necessary.
- 3. NSF4 filters the noise in raw image and applies white balance gains in gamma corrected format.
- 4. GLBCE compresses the image dynamic range to 12-bit.
- 5. FCP performs demosaicing, color linearization and correction, display gamma correction, RGB to YUV conversion, and edge sharpening.

In general, tuning shall be done in the above order of the data flow.

# **6 IQ Tuning Process**

#### 6.1 Expectations of the Tool

Optimal image quality from a camera system is a function of the sensor, lens and ISP all tuned for best possible quality. TI provided tool and software are designed to tune TI ISP only. Before starting ISP tuning, it is strongly recommended that image sensor has been configured to get output best possible quality RAW images. To get optimal sensor configuration, please contact the sensor vendor. TI SDK includes sample configuration and tuning for some image sensors, but this is not guaranteed to be production ready.

#### 6.1.1 User Expertise

- Tuning tool and Jacinto software try to abstract low level details of VPAC so that you can get started rapidly on TI SoCs. However, for best results, an understanding of VPAC usage model will be beneficial.
- Tuning tool is not a replacement for imaging expertise. Users must have a thorough understanding of IQ tuning and the challenges involved. Users must fully understand the dataflow and the role of every tunable IP in the image pipe.

#### 6.1.2 Ease of Use

- TI has designed the tools and software to be as user friendly as possible. However, this is a not a single click automatic tuning tool. Some procedures must be followed and TI tools used in a specific way, as per tool documentation.
- In some cases, you may need additional tools for steps not covered by TI tool.

#### 6.1.3 Troubleshooting

#### Correctly root-causing IQ issues is critical

- · IQ degradation may come from sensor, lens, ISP or software defects.
- Tuning Tool can help only with ISP issues.
- Users need to root-cause the issue to ISP and then to a sub-block in ISP.



· Once the issue has been root-caused correctly, the specific ISP block can be tuned using the tool.

# 7 Imaging Software Architecture

Imaging component in PSDK RTOS (<PSDK RTOS\_root>/imaging) includes sensor drivers, IQ tuning and target side algorithms for image quality enhancement. The supported algorithms are:

# 7.1 Auto White Balance

Analyzes VISS H3A output and computes color gains for the ambient lighting conditions. Communicates the color gains and estimates color temperature to OpenVX VISS node, which in turn:

- · Writes the color gains to appropriate MMRs.
- Uses the color temperature to select the appropriate CCM for color tuning.

Some sensors (AR0233) support a special mode where AWB gains are written to the sensor pre-HDR merge. This results in lower motion artifacts. This mode can be enabled in the sensor driver's *AR0233\_GetWBPrgFxn* API. By default, it is OFF, for example, AWB gains are applied in VISS.

# 7.2 AutoExposure

Analyzes VISS H3A output and computes sensor exposure time and analog gain to ensure correctly exposed images. The computed parameters are sent to:

- Sensor driver Translates to sensor register values for configuring the sensor over I2C.
- OpenVX VISS node uses the information to pick the right photospace for plugins like NSF4 and EE.

# 7.3 Dynamic Camera Configuration (DCC) Algorithm

IQ tuning database for a given sensor consists of:

- Plugin Params Every tunable block (hardware or software) is modeled as a plugin. Each plugin can be tuned independently. Some examples of HW plugins are NSF4, GLBCE etc. Auto White Balance algorithm is a software plugin.
- Photospace Params Some plugins have only 1 set of parameters that stay constant through the lifecycle of the camera. WDR PWL is an example of such a plugin. However, there are plugins like NSF4, Color Fidelity, and so forth, which have a different set of parameters for different lighting conditions. The lighting conditions can be divided into photospaces. Each photospace can be tuned separately and the tuning parameters can be saved in the tuning database.

Tuning database can be queried at runtime to get the optimal parameters for a given combination of sensor ID, plugin and lighting condition.

#### DCC algorithm is responsible for parsing the database and returning the tuning parameters.

# 8 Computing Tuning Parameters

The process of computing tuning parameters is called IQ Tuning. This process can be facilitated through the tuning tool. You are expected to start from a default tuning profile, which can be reused from one of the existing sensors in the SDK. In the beginning, image quality is expected to be low. You can start tuning plugin by plugin and experience an improvement in image quality. Typically, the sequence of plugins to be tuned is same as the order in which they are processed on the hardware. For more information, see Section 5.

# 9 Testing Tuning Parameters

PSDK RTOS comes with a set of utility functions in the folder *PSDK RTOS\_root*/vision\_apps/utils/iss. The file app\_iss\_common.c includes DCC databases of all the sensors as C-style arrays. There are three OpenVX nodes that support DCC tuning:

- VISS
- AEWB
- LDC

7



There is a unique DCC parameter array for each node-sensor combination. The application (app\_single\_cam) creates an OpenVX graph connecting all the nodes. While creating the nodes, it passes the DCC buffer for the node as follows:

- Queries ISS utility to get the size of DCC database for a given sensor-node combination.
- Allocates the memory and creates a user data object

Passes that buffer pointer to ISS utility that copies the tuning database to the object.

This is repeated for all the DCC enabled nodes. When the node is created, Create method of the node uses the DCC buffer to initialize internal structures which are used later for configuring VPAC drivers.

DCC database is read from the header files in the folder

<PSDK RTOS\_root>/imaging/sensor\_drv/include

There are several options to update DCC database as described in the following sections.

# 9.1 On Target Platform – Compile Time Update

Application app\_single\_cam starts with DCC database from sensor driver header file. This can be overridden at runtime by reading DCC binary files from file system on the SD card. app\_single\_cam has a runtime menu as shown below

```
" =======""
" Demo : Single Camera w/ 2A"
" ===============""
" p: Print performance statistics"
" s: Save Sensor RAW, VISS Output and H3A output images to File System"
""
" e: Export performance statistics"
" u: Update DCC from File System"
" x: Exit"
" Enter Choice: "
```

Option 'u' instructs the application to find the DCC binary files in the folder:

/opt/vision\_apps/dcc/<sensor\_name>/<wdr\_mode>

Where

- sensor\_name = string containing the sensor name as defined in sensor driver. For e.g. IMX390 name is "IMX390-UB953\_D3"
- wdr\_mode = "wdr" or "linear"

Before giving the command 'u', make sure that this folder exists and contains the DCC files in binary format. If this folder contains partial DCC database, then only that part will be overridden. Example:

- User launches app\_single\_cam
- User selects sensor IMX390. Application selects WDR mode by default
   Application code can be modified to change the default to linear, if needed
- Applications reads default DCC profile from sensor driver and starts streaming
- User copies RGB2RGB and AWB into the folder

#### /opt/vision\_apps/dcc/IMX390-UB953\_D3/wdr/

- User presses 'u'
- Applications updates RGB2RGB and AWB tuning, leaving rest of the tuning same as initial tuning read from sensor driver. The effect of new tuning parameters can be seen immediately.



# 9.2 On Target Platform – Run Time Update From File System

At runtime, you can use DCC tool's live update function to override create time parameters. This results in an immediate effect of new parameters without the need for any code changes or restarting the application. The effect is similar to previous section, but the binary files are sent over Ethernet connection instead of being read from SD card.

For more details, see section Section 10.

# **10 Live Tuning**

Live tuning features allow tuning tool to interact with target system at runtime over Ethernet interface.

Prerequisites

- J7 EVM connected to network using Ethernet.
- PC and EVM on the same network. PC must be able to ping the EVM.
  - VPN connection can interfere with this. Please disconnect PC from VPN before trying live tuning.
  - EVM IP address can be found using "ifconfig" on EVM Linux console.
- app\_single\_cam application must be running on the EVM.

After the app\_single\_cam application is launched, the connection can be established by entering EVM IP address in the tool window.



Figure 10-1. IP Address Entry in Live Tuning Bar

# **10.1 Supported Features**

#### 10.1.1 RAW Capture

Saves a raw image from the output of capture node. This image is the output of image sensor and is completely unprocessed by J7 ISP. The width, height and bit-depth will be determined by the sensor driver.

File	Options	Plug-ins	Execute	Help				
F	2	୧ ୧	۹ 🛛 🐐	6	2	192.168.1.80	<b>I</b> III III III III III IIII IIII IIII	0
Came	ra Informa	ation & Con	trol 🖻				$\mathbf{U}$	

Figure 10-2. RAW File Save Button in Live Tuning Bar

#### 10.1.2 YUV Capture

Saves a YUV image from the output of VISS node. This image is the output of ISP, before LDC or scalar. The width, height and bit-depth will be determined by VISS node.



Figure 10-3. YUV File Save Button in Live Tuning Bar

# 10.1.3 AutoExposure Control

#### Supported features

· Read exposure and gain computed by AE algorithm



Figure 10-4. Exposure Control (Auto Mode) in Live Tuning Window

• Bypass AE by setting mode = Manual. Set manual exp and gain values, subject to sensor driver limits.



Figure 10-5. Exposure Control (Manual Mode) in Live Tuning Window



#### 10.1.4 AutoWhiteBalance Control

#### **Supported features**

Read exposure and gain computed by AWB algorithm



Figure 10-6. White Balance Control (Auto) in Live Tuning Window

 Bypass AWB by setting mode = Manual. Set manual gains and color temperature, subject to sensor/VISS driver limits.



Figure 10-7. White Balance Control (Manual) in Live Tuning Window

#### 10.1.5 Sensor Register Read/Write

Read from or write to image sensor registers. Switch to memory view tab.

File Options Plug-ins Execute Help	
📓 🍳 🍳 🔍 🐫 ઇ 隆 🍡	192.168.1.80 📕 🏧 🜔 📀
Camera Information & Control	Ø
Memory View Algorithm View	
0x3012	
/alue in Dec /alue in Hea 32 0x20	
Plugin output	8
.oaded communication DLL:itCommun.dll Sending the Camera Control Params: 192.168.1.80 .oaded communication DLL:itCommun.dll Successfull Vransfered Camera Control data	Â
Loaded communication DLL:itCommun.dll	



#### 10.1.6 Live DCC Update

#### 10.1.6.1 Update Single Plugin

Sends DCC binary of the plugin currently open in tool GUI to the EVM. Instant impact on the image quality can be seen on the display connected to the EVM.



#### Figure 10-9. Current Plugin DCC Update Control in Live Tuning Window

The change is valid only until the app\_single\_cam is running. As soon as the app is closed the changes are lost.

#### 10.1.6.2 Update Multiple Plugins

Prompts you to select several DCC binary files and sends all of them to the EVM.



#### Figure 10-10. Multi Plugin DCC Update Control in Live Tuning Window

The change is valid only until the app\_single\_cam is running. As soon as the app is closed the changes are lost.

#### **11 Revision History**

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

С	hanges from Revision * (December 2020) to Revision A (January 2021)	Page
•	Updated the numbering format for tables, figures and cross-references throughout the document	2
•	Updates were made in Section 2.1.	2



Revision His
--------------

•	Updates were made in Section 3.1.	3
•	Update was made in Section 5	. 6
		-

# IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265 Copyright © 2022, Texas Instruments Incorporated