Guide to Writing Drivers for GMSL or FPD-Link ADKs

This document contains proprietary information to Teledyne FLIR, LLC. This information may be maintained in confidence and used only in a manner consistent with the document and any executed Non-Disclosure Agreement and may not be disclosed to any third parties without Teledyne FLIR’s written consent.

EAR Controlled: EAR99

These commodities, technology or software are subject to the U.S. Export Administration Regulations (EAR). Diversion contrary to U.S. law is prohibited.

EAR Controlled – See Cover Page
PROPRIETARY – Teledyne FLIR, LLC
Table of Contents

1 Introduction .................................................................................................................. 3
  1.1 Revision History ........................................................................................................ 3
  1.2 SCOPE ........................................................................................................................ 3
  1.3 BACKGROUND .......................................................................................................... 3

2 Existing Drivers ............................................................................................................. 3
  2.1 GSML1 for NVIDIA Drive PX2 .................................................................................. 3
  2.2 GMSL2 for NVIDIA Drive Xavier ............................................................................. 4
  2.3 GMSL2 for NVIDIA Jetson AGX Xavier – JetPack 4.3 ............................................ 4
  2.4 FPD-Link III for Jetson AGX Xavier – JetPack 4.6 .................................................. 4

3 Hardware Description .................................................................................................. 4
  3.1 Basic Architecture .................................................................................................... 5
  3.2 Chip Details and Pinouts ......................................................................................... 5
  3.3 Power over Coax (PoC) .......................................................................................... 6
  3.4 16bit images ........................................................................................................... 6

4 Existing Documentation and References ..................................................................... 6
  4.1 Boson Engineering Data Sheet .............................................................................. 6
  4.2 ADK Getting Started Guide .................................................................................. 6
  4.3 Boson SDK ............................................................................................................ 6
  4.4 Boson SDK Documentation .................................................................................. 7
  4.5 Boson FFC NUC Control Application Note ............................................................. 8
1 Introduction

1.1 Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1/7/2022</td>
<td>Initial Release</td>
</tr>
</tbody>
</table>

1.2 SCOPE

This document serves to provide the necessary information to implement a custom driver for the GMSL or FPD-Link version of the Automotive Development Kit (ADK) thermal camera. This is NOT a step by step guide. Prior knowledge of programming, MIPI, GMSL/FPD-Link, device drivers, and your host computing platform are needed to understand this guide and implement your own driver.

1.3 BACKGROUND

The FLIR ADK is FLIR’s Boson camera in an IP67 enclosure. It provides a Long Wave Inferred Image (LWIR) that is 640x512 pixels. It senses thermal radiation with wavelengths of 8-14um. The output is either a 8bit or 16bit monochrome image.

The ADK comes in three different interfaces: USB output, GMSL output, FPD-Link and GMSL output with a converter for GMSL to Ethernet. The ADK also has four different lens options.

<table>
<thead>
<tr>
<th>Horizontal FOV</th>
<th>Vertical FOV</th>
<th>F#</th>
<th>Focal Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>19.2</td>
<td>F1.1</td>
<td>18mm</td>
</tr>
<tr>
<td>32</td>
<td>25.6</td>
<td>F1.0</td>
<td>14.0mm</td>
</tr>
<tr>
<td>50</td>
<td>40</td>
<td>F1.0</td>
<td>9.2mm</td>
</tr>
<tr>
<td>75</td>
<td>60</td>
<td>F1.0</td>
<td>6.2mm</td>
</tr>
</tbody>
</table>

The output format of the camera is controlled by a separate ‘personality’ board that is integrated into the ADK. The GMSL and FPD-Link personality boards are specific to the ADK and are not covered in other ADK or Boson Documentation.

2 Existing Drivers

There are several existing drivers.

2.1 GSML1 for NVIDIA Drive PX2
GMSL1 for the NVIDIA Drive PX2 line of computers. This driver is modifications to the existing img_cc application. While the PX2 line is obsolete, this driver does have ‘boson*.script’ which contains SerDes register settings for configuring the link. You can find this driver at: https://flir.box.com/s/hnjfmjdsv1xlqevoz8gyj26v127f7i8j

2.2 GMSL2 for NVIDIA Drive Xavier

Very similar to the Driver PX2 driver.
https://flir.box.com/s/siws9s3tzesnd0kwfk2dtui7vs4brlpwi

2.3 GMSL2 for NVIDIA Jetson AGX Xavier – JetPack 4.3

This driver works with JetPack 4.3 on the Jetson AGX Xavier with a D3 SerDes card (https://www.d3engineering.com/product/designcore-nvidia-jetson-serdes-card/). There is a sample python application for streaming video to the screen. This driver is compiled directly into the kernel. The installation process replaces the kernel on your machine. You can find the compiled driver here: https://flir.box.com/s/jm6hiof1j7xsen042qxyiuwtq08xjhou

The source can be found here:
https://flir.box.com/s/mumcxqullpfj7x5lnilnn20qk2g32ucn. Note that ADK specific source is embedded in a combination of existing source files and ADK (Boson) specific files.

Sample python application: https://github.com/FLIR/Xavier_OpenCV_Sample

Note this driver only provides 8bit imagery.

2.4 FPD-Link III for Jetson AGX Xavier – JetPack 4.6

The driver works with JetPack 4.6 on the Jetson AGX Xavier with the D3 Engineering SerDes card (https://www.d3engineering.com/product/designcore-nvidia-jetson-serdes-card/). It will stream to the screen using a gstlaunch command. The driver is a modified kernel and 3 modules (one for the deserializer, one for the serializer, one for the camera). The source of this driver is the NVIDIA kernel and patch files that clearly show the changes made to implement the drive. These patch files are probably your best starting point to implement any driver of your own (GMSL or FPD-Link).

3 Hardware Description
3.1 Basic Architecture

The Boson Camera provides an 8bit wide parallel CMOS video output and a UART command and control interface. The parallel CMOS is feed into a GMSL or FPD-Link serializer chip. Since the backchannel on both GMSL and FPD-Link is I2C, an I2C to UART Expander is used to talk with the Boson Camera. External sync (not required) is from a GPIO pin on the serializer. The window heater and camera reset are controller from GPIO pins on the I2C to UART Expander.

![Figure 1 – GSML/FPD-Link ADK Block Diagram](image)

3.2 Chip Details and Pinouts

<table>
<thead>
<tr>
<th>Function</th>
<th>GMSL2</th>
<th>FPD-Link III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serializer</td>
<td>MAX9295A</td>
<td>DS90UB933-Q1</td>
</tr>
<tr>
<td>Serializer HW I2C Addr</td>
<td>0x84</td>
<td>0x5A/0xB4 (7bit/8bit)</td>
</tr>
<tr>
<td>External Sync Pin</td>
<td>MFP7 – pin31</td>
<td>GPO(1) – pin 16</td>
</tr>
<tr>
<td>I2C to Uart Bridge</td>
<td>MAX3108</td>
<td>MAX3108</td>
</tr>
<tr>
<td>Bridge HW I2C Addr</td>
<td>0xD8</td>
<td>0xD8</td>
</tr>
<tr>
<td>Boson Reset Pin</td>
<td>GPIO0 – pin C4</td>
<td>GPIO0 – pin C4</td>
</tr>
<tr>
<td>Heater Enable Pin</td>
<td>GPIO3 – pin D4</td>
<td>GPIO4 – pin D4</td>
</tr>
</tbody>
</table>

The I2C address listed here are the hardware addresses. It is highly likely that your deserializer configuration will alias these values to different addresses.

The GMSL1 ADK is the same hardware as the GSML2 ADK but bootstrapped to boot into GMSL1. This is a hardware configuration done during manufacturing and not accessible by the user. It is possible to switch GMSL modes after power up but establishing a connection in the bootstrap GMSL mode and reconfiguring registers for the other mode.

**EAR Controlled – See Cover Page**

**PROPRIETARY – Teledyne FLIR, LLC**
3.3 Power over Coax (PoC)

Power for the SerDes ADK is provided by power over coax. The PoC filter design is critical to providing enough power to the camera. In many cases running the heater and shutter at the same time will draw too much power. Teledyne FLIR has a PoC worksheet that is available under NDA. Alternatively a PoC injector circuit is described in the ADK Getting Started Guide.

3.4 16bit images

Since the input to the serializer is 12 bits wide, it is not possible to directly transfer 16bit images. To transfer 16bit images, we produce a ‘double wide’ frame that is 1280x512. The first pixel of the 1280 frame is the upper byte of the 640 image. The second pixel of the 1280 frame is the lower byte of the 640 image. These bytes will have to be reassembled on the computer end of the link. There is a more in depth description of this functionality in the ADK Getting Started Guide.

4 Existing Documentation and References

4.1 Boson Engineering Data Sheet

The Boson Engineering Data Sheet has detailed descriptions of the features and functionality of the camera. It can be found on the documents tab of the Boson support page (https://www.flir.com/support/products/boson/#Documents) or the current version (at time of this document) at https://flir.netx.net/file/asset/15754/original/attachment

4.2 ADK Getting Started Guide

The ADK Getting Started Guide contains valuable information about the ADK https://flir.box.com/s/on5l2wfq6rihxhpfeuefplsl4vhs2jz

4.3 Boson SDK

The SDK provides a software (C, C#, Python) interface to command and control the camera. The package is designed to work directly with a UART. However the package is designed with the UART driver in a separate folder. The driver can easily be modified for new architectures such as I2C. The UART driver is in the FSLP_Files or EmulatorFiles folder. The
FPDLink driver contains an already modified version of the SDK, however the low level I2C read/write commands may be different on your system.

Boson SDK: https://flir.box.com/s/g8x22crd9qw1h8u5h1zuat0hql3oy6z2

The version of the SDK included with the FPD-Link driver has already been modified to work over I2C on the NVIDIA Jetson AGX Xavier. You can launch this version of the SDK by starting Python3, importing pyClientUserStartup, and running a command. Note: the first command issued sometimes returns an error despite being executed by the camera.

```python
>>> import pyClientUserStartup as vid
[sudo] password for nvidia:
Python serial over I2C FSLP load
>>> vid.cam.bosonRunFFC()
```

When writing to the I2C to UART chip, it highly recommend to disable the UART, write a whole command packet to the chip and then enable the UART. This results in the packet being sent to the camera in as short a time as possible. In some cases longer commands may take too long to write over I2C and the camera will time out. Example code is below.

```python
def writeBosonCmd(values):
    # sends a series of bytes to Boson's uart port
    # first turns off the I2C to UART transmitter,
    # loads the buffer and enables transmitter so all
    # the byte arrive in a timely manner
    Max2108addr = 0xD8
    board.WriteI2C(Max2108addr, 0x09, 0x02)
    #text = ""
    for val in values:
        board.WriteI2C(Max2108addr, 0x00, val)
        #text += hex(val) + " "
    board.WriteI2C(Max2108addr, 0x09, 0x00)
```

4.4 Boson SDK Documentation

Describes the functions in the SDK as well as the information need to implement your own SDK.

https://flir.netx.net/file/asset/12950/original/attachment
If implementing from this document, it is recommended to test commands that send data to and from the camera as well as commands without data.

bosonRunFFC() – command without data. The camera should make an audible ‘click’ when this command is issued.
bosonGetCameraSN() – command receiving data. This command gets the camera serial number.
telemetrySetLocation() – sends data to the camera. This command sets if the telemetry line is at the top or bottom of the frame. (A separate command enables/disable telemetry).

4.5 Boson FFC NUC Control Application Note

The Boson camera inside the ADK contains a mechanical shutter. The shutter provides a Flat Field Correction (FFC) to the image. During the FFC, the shutter puts a fin a metal in the optical path and provides a single point calibration on every pixel. This process provides a better quality images but during the process the camera does not provide any new imagery.

By default the camera is in automatic shutter mode where the shutter will run every 5 minutes or if the focal plane array has change temperature my 1°C. For automotive applications we strongly suggest placing the camera into manual shutter mode. In the mode the camera will only run the shutter at boot and when commanded. The user can then select when the FFC executes (for example at a stop sign or red traffic light when the vehicle is not moving.) The telemetry line contains a field that changes when the camera requests an FFC.

https://flir.box.com/s/lvv9109hpkdh4qryo970fiuzlpehz8t