Pixel Formats – Area Scan

Looking for Line Scan or 3D Pixel Formats? Visit our **Line Scan Pixel Format (https://support.thinklucid.com/knowledgebase/pixel-formats-line-scan/)** and our **3D Pixel Format (https://support.thinklucid.com/knowledgebase/pixel-formats-3d/)** pages for more information.

Introduction

Understanding pixel formats allow you to understand how images are put together. More specifically, understanding these pixel formats will help you if you need to extract specific information from the pixels in the image that you receive from the camera.

Jump to:

BayerRG12Packed
BayerRG16
BayerRG24

Dual Bayer RG formats

DualBayerRG8
DualBayerRG10
DualBayerRG10p
DualBayerRG10Packed
DualBayerRG12
DualBayerRG12p
DualBayerRG12Packed
DualBayerRG16

Polarized pixel formats

PolarizedAngles_0d_45d_90d_135d_Mono8

PolarizedStokes_S0_S2_S1_S3_Mono8

PolarizedDolpAolp_Mono8
PolarizedDolpAolp_Mono12p
PolarizedDolp_Mono8
PolarizedDolp_Mono12p
PolarizedAolp_Mono8
PolarizedAolp_Mono12p

Basic mono pixel formats

Mono8
Mono10
Mono10p
Mono10Packed
Mono12
Mono12p
Mono12Packed
Mono16

YCbCr pixel formats: basic and subsampled

YCbCr – Basic
YCbCr- Subsampled

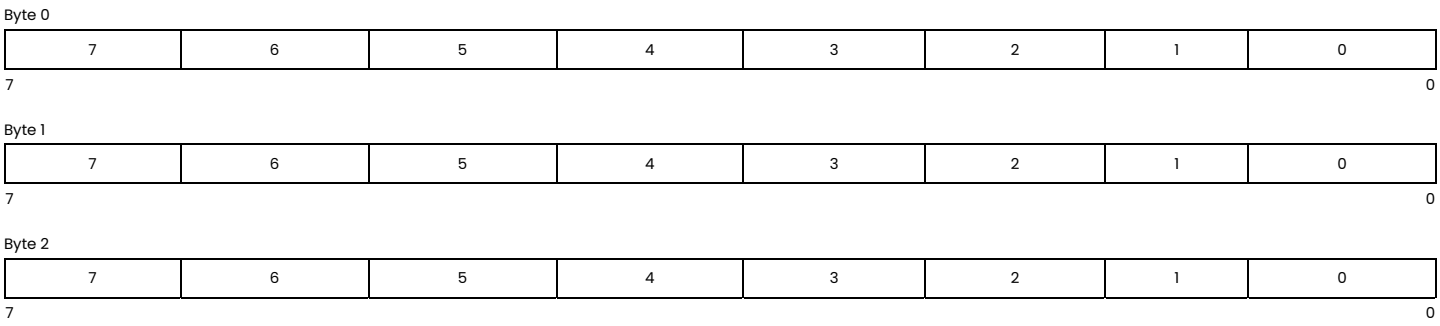
QOI pixel formats

Description

Color-processed pixel formats

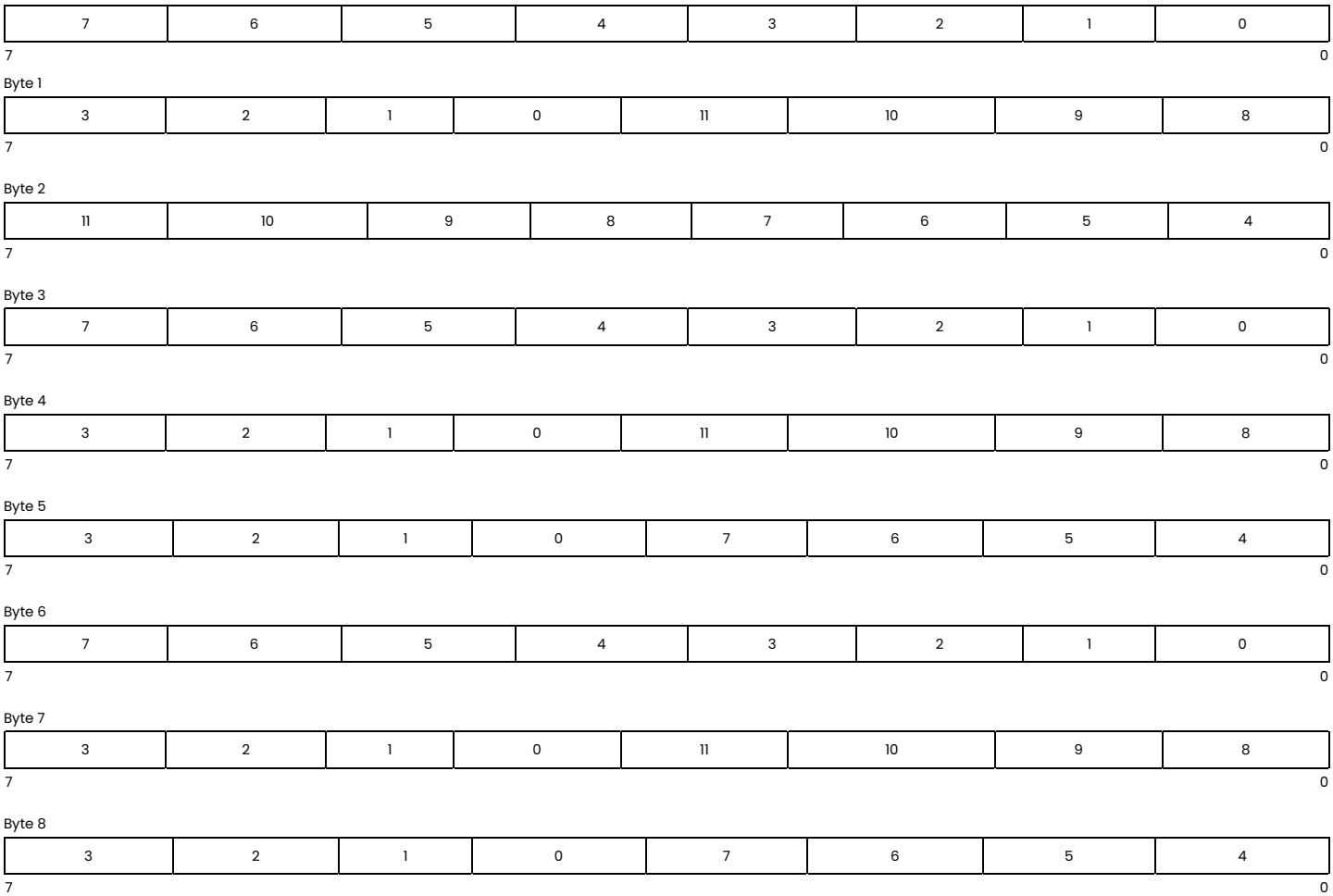
FOR YCbCr and YUV, see the separate sections for these formats.

RGB8



RGB12p

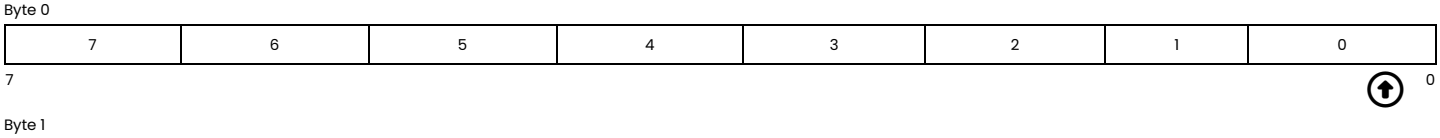
Byte 0


7 6 5 4 3 2 1 0
7 0
Byte 1
3 2 1 0 11 10 9 8
7 0
Byte 2
11 10 9 8 7 6 5 4
7 0
Byte 3
7 6 5 4 3 2 1 0
7 0
Byte 4
3 2 1 0 11 10 9 8
7 0
Byte 5
3 2 1 0 7 6 5 4
7 0
Byte 6
7 6 5 4 3 2 1 0
7 0
Byte 7
3 2 1 0 11 10 9 8
7 0
Byte 8
3 2 1 0 7 6 5 4
7 0

RGB16


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

RGB24


Byte 0
7 6 5 4 3 2 1 0
7 0
Byte 1

15	14	13	12	11	10	9	8
7							0

Byte 2

23	22	21	20	19	18	17	16
7							0

Byte 3

7	6	5	4	3	2	1	0
7							0

Byte 4

15	14	13	12	11	10	9	8
7							0

Byte 5

23	22	21	20	19	18	17	16
7							0

Byte 6

7	6	5	4	3	2	1	0
7							0

Byte 7

15	14	13	12	11	10	9	8
7							0

Byte 8

23	22	21	20	19	18	17	16
7							0

BGR8

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

7	6	5	4	3	2	1	0
7							0

Byte 2

7	6	5	4	3	2	1	0
7							0

BGR12p

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

3	2	1	0	11	10	9	8
7							0

Byte 2

11	10	9	8	7	6	5	4
7							0

Byte 3

7	6	5	4	3	2	1	0
7							0

Byte 4

3	2	1	0	11	10	9	8
7							0

Byte 5

11	10	9	8	7	6	5	4
7							0

Byte 6

7	6	5	4	3	2	1	0
7							0

Byte 7

3	2	1	0	11	10	9	8
7							0

Byte 8

11	10	9	8	7	6	5	4
7							0

BGR16

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

15	14	13	12	11	10	9	8
7							0

Byte 2

7	6	5	4	3	2	1	0
7							0

Byte 3

15	14	13	12	11	10	9	8
7							0

Byte 4

7	6	5	4	3	2	1	0
7							0

Byte 5

15	14	13	12	11	10	9	8
7							0

BGR24

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

15	14	13	12	11	10	9	8
7							0

Byte 2

23	22	21	20	19	18	17	16
7							0

Byte 3

7	6	5	4	3	2	1	0
7							0

Byte 4

15	14	13	12	11	10	9	8
7							0

Byte 5

23	22	21	20	19	18	17	16
7							0

Byte 6

7	6	5	4	3	2	1	0

7 0

Byte 7

15	14	13	12	11	10	9	8

7 0

Byte 8

23	22	21	20	19	18	17	16

7 0

Bayer RG formats

The BayerRG pixel formats are repetitions of a 2×2 grid comprising a top row containing one red (R) and one green (G) filter, and a bottom row containing one green (G) and one blue (B) filter.

Bayer formats are raw pixel formats.

BayerRG8

Byte 0

7	6	5	4	3	2	1	0

7 0

Byte 1

7	6	5	4	3	2	1	0

7 0

Byte 2

7	6	5	4	3	2	1	0

7 0

Byte 3

7	6	5	4	3	2	1	0

7 0

Byte n

7	6	5	4	3	2	1	0

7 0

Byte n+1

7	6	5	4	3	2	1	0

7 0

Byte n+2

7	6	5	4	3	2	1	0

7 0

Byte n+3

7	6	5	4	3	2	1	0

7 0

BayerRG10

Byte 0

7	6	5	4	3	2	1	0

7 0

Byte 1

pad	pad	pad	pad	pad	pad	9	8

7 0

Byte 2

7	6	5	4	3	2	1	0

7 0

Byte 3

pad	pad	pad	pad	pad	pad	9	8

7 0

Byte 2n

7	6	5	4	3	2	1	0

7 0

Byte 2n+1

pad	pad	pad	pad	pad	pad	9	8

7 0

Byte 2n+2

7	6	5	4	3	2	1	0

7 0

Byte 2n+3

pad	pad	pad	pad	pad	pad	9	8

7 0

BayerRG10p

Byte 0

7	6	5	4	3	2	1	0

7 0

Byte 1

5	4	3	2	1	0	9	8

7 0

Byte 2

3	2	1	0	9	8	7	6

7 0

Byte 3

1	0	9	8	7	6	5	4

7 0

Byte 4

9	8	7	6	5	4	3	2

7 0

Byte 1.25n

7	6	5	4	3	2	1	0

7 0

Byte 1.25n+1

5	4	3	2	1	0	9	8

7 0

Byte 1.25n+2

3	2	1	0	9	8	7	6

7 0

Byte 1.25n+3

1	0	9	8	7	6	5	4

7 0

Byte 1.25n+4

9	8	7	6	5	4	3	2

7 0

BayerRG10Packed

Byte 0

9	8	7	6	5	4	3	2

7 0

Byte 1

7	6	5	4	3	2	1	0
pad	pad	1	0	pad	pad	1	0

Byte 2

7	6	5	4	3	2	1	0
9	8	7	6	5	4	3	2

Byte 1.5n

7	6	5	4	3	2	1	0
9	8	7	6	5	4	3	2

Byte 1.5n+1

7	6	5	4	3	2	1	0
pad	pad	1	0	pad	pad	1	0

Byte 1.5n+2

7	6	5	4	3	2	1	0
9	8	7	6	5	4	3	2

BayerRG12

Byte 0

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0

Byte 1

7	6	5	4	3	2	1	0
pad	pad	pad	pad	11	10	9	8

Byte 2

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0

Byte 3

7	6	5	4	3	2	1	0
pad	pad	pad	pad	11	10	9	8

Byte 2n

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0

Byte 2n+1

7	6	5	4	3	2	1	0
pad	pad	pad	pad	11	10	9	8

Byte 2n+2

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0

Byte 2n+3

7	6	5	4	3	2	1	0
pad	pad	pad	pad	11	10	9	8

BayerRG12p

Byte 0

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0

Byte 1

7	6	5	4	3	2	1	0
3	2	1	0	11	10	9	8

Byte 2

7	6	5	4	3	2	1	0
11	10	9	8	7	6	5	4

Byte 1.5n

7	6	5	4	3	2	1	0

7 0

Byte 1.5n+1

3	2	1	0	11	10	9	8

7 0

Byte 1.5n+2

11	10	9	8	7	6	5	4

7 0

BayerRG12Packed

Byte 0

11	10	9	8	7	6	5	4

7 0

Byte 1

3	2	1	0	3	2	1	0

7 0

Byte 2

11	10	9	8	7	6	5	4

7 0

Byte 1.5n

11	10	9	8	7	6	5	4

7 0

Byte 1.5n+1

3	2	1	0	3	2	1	0

7 0

Byte 1.5n+2

11	10	9	8	7	6	5	4

7 0

BayerRG16

Byte 0

7	6	5	4	3	2	1	0

7 0

Byte 1

15	14	13	12	11	10	9	8

7 0

Byte 2

7	6	5	4	3	2	1	0

7 0

Byte 3

15	14	13	12	11	10	9	8

7 0

Byte 2n

7	6	5	4	3	2	1	0

7 0

Byte 2n+1

15	14	13	12	11	10	9	8

7 0

Byte 2n+2

7	6	5	4	3	2	1	0

7 0

Byte 2n+3

15	14	13	12	11	10	9	8

7 0

BayerRG24

Byte 0

7	6	5	4	3	2	1	0

7 0

Byte 1

15	14	13	12	11	10	9	8

7 0

Byte 2

23	22	21	20	19	18	17	16

7 0

Byte 3

7	6	5	4	3	2	1	0

7 0

Byte 4

15	14	13	12	11	10	9	8

7 0

Byte 5

23	22	21	20	19	18	17	16

7 0

Byte 3n

7	6	5	4	3	2	1	0

7 0

Byte 3n+1

15	14	13	12	11	10	9	8

7 0

Byte 3n+2

23	22	21	20	19	18	17	16

7 0

Byte 3n+3

7	6	5	4	3	2	1	0

7 0

Byte 3n+4

15	14	13	12	11	10	9	8

7 0

Byte 3n+5

23	22	21	20	19	18	17	16

7 0

Dual Bayer formats

The Dual Bayer pixel formats are for the dual extended-head cameras.

In the diagram below, Imager 1 pixels are shown with a regular border around each bit, and Imager 2 pixels are shown with a bold line around each bit.

Imager 1:

1

Imager 2:

1

DualBayerRG8

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

7	6	5	4	3	2	1	0
7							0

Byte 2

7	6	5	4	3	2	1	0
7							0

Byte 3

7	6	5	4	3	2	1	0
7							0

Byte 2n

7	6	5	4	3	2	1	0
7							0

Byte 2n+1

7	6	5	4	3	2	1	0
7							0

Byte 2n+2

7	6	5	4	3	2	1	0
7							0

Byte 2n+3

7	6	5	4	3	2	1	0
7							0

DualBayerRG10

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 2

7	6	5	4	3	2	1	0
7							0

Byte 3

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 4

7	6	5	4	3	2	1	0
7							0

Byte 5

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 6

7	6	5	4	3	2	1	0
7							0

Byte 7

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 4n

7	6	5	4	3	2	1	0
7							0

Byte 4n+1

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 4n+2

7	6	5	4	3	2	1	0
7							0

Byte 4n+3

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 4n+4

7	6	5	4	3	2	1	0
7							0

Byte 4n+5

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 4n+6

7	6	5	4	3	2	1	0
7							0

Byte 4n+8

pad	pad	pad	pad	pad	pad	9	8
7							0

DualBayerRG10p

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

5	4	3	2	1	0	9	8
7							0

Byte 2

3	2	1	0	9	8	7	6
7							0

Byte 3

1	0	9	8	7	6	5	4
7							0

Byte 4

9	8	7	6	5	4	3	2
7							0

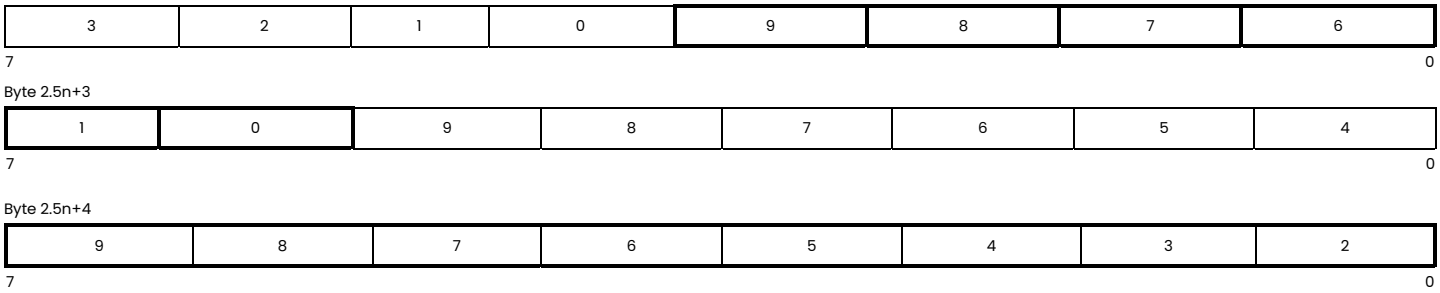
Byte 2.5n

7	6	5	4	3	2	1	0
7							0

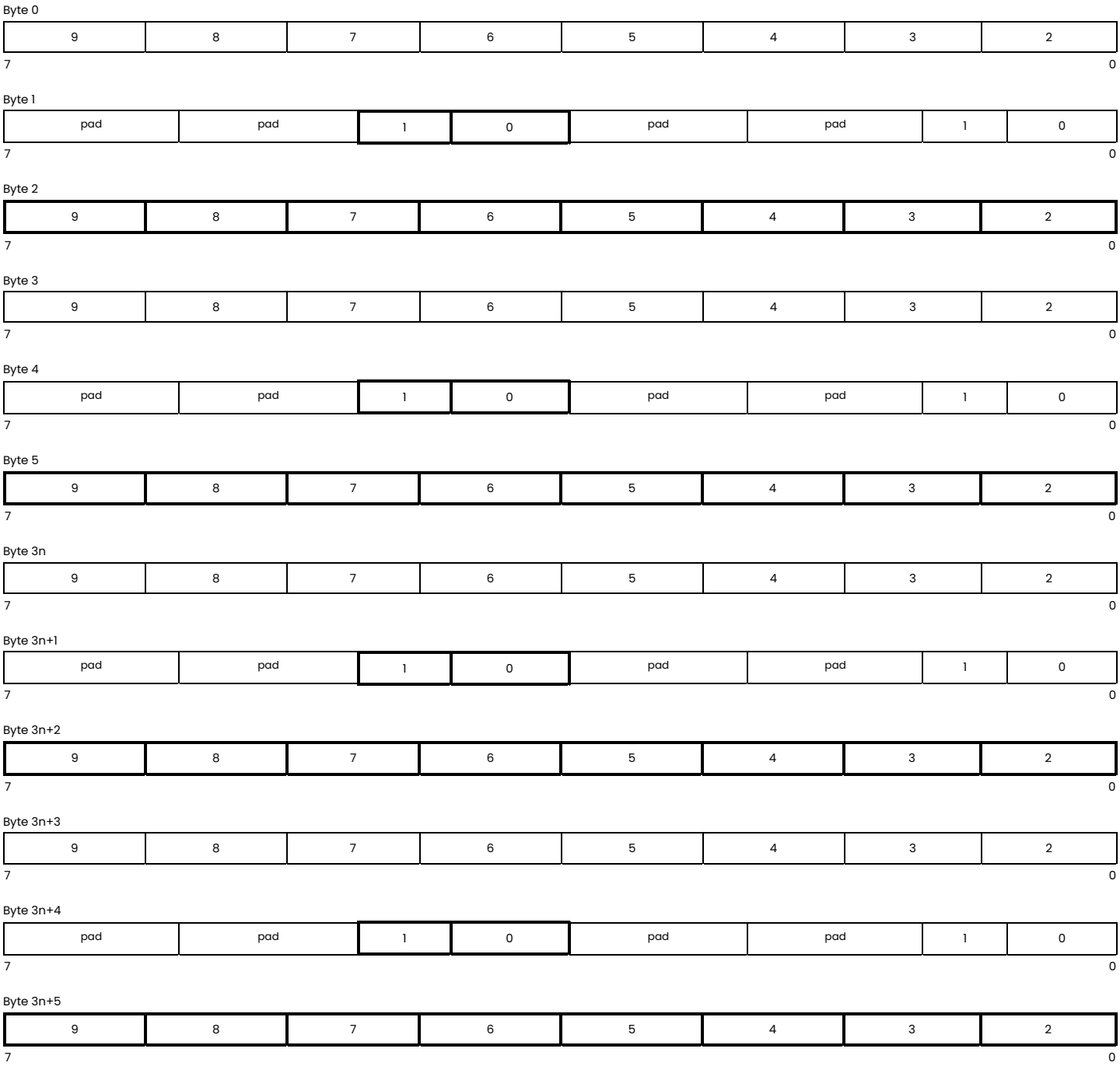
Byte 2.5n+1

5	4	3	2	1	0	9	8
7							0

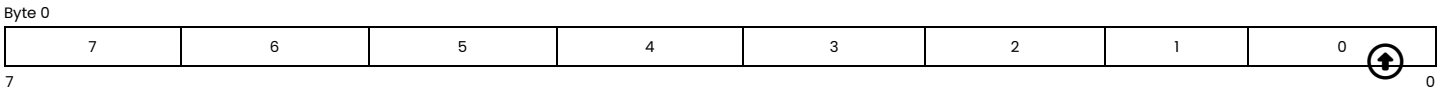
Byte 2.5n+2



DualBayerRG10Packed



DualBayerRG12



pad	pad	pad	pad	11	10	9	8
7							0

Byte 2

7	6	5	4	3	2	1	0
7							0

Byte 3

pad	pad	pad	pad	11	10	9	8
7							0

Byte 4

7	6	5	4	3	2	1	0
7							0

Byte 5

pad	pad	pad	pad	11	10	9	8
7							0

Byte 6

7	6	5	4	3	2	1	0
7							0

Byte 7

pad	pad	pad	pad	11	10	9	8
7							0

Byte 4n

7	6	5	4	3	2	1	0
7							0

Byte 4n+1

pad	pad	pad	pad	11	10	9	8
7							0

Byte 4n+2

7	6	5	4	3	2	1	0
7							0

Byte 4n+3

pad	pad	pad	pad	11	10	9	8
7							0

Byte 4n+4

7	6	5	4	3	2	1	0
7							0

Byte 4n+5

pad	pad	pad	pad	11	10	9	8
7							0

Byte 4n+6

7	6	5	4	3	2	1	0
7							0

Byte 4n+7

pad	pad	pad	pad	11	10	9	8
7							0

DualBayerRG12p

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

3	2	1	0	11	10	9	8

7 0

Byte 2

11	10	9	8	7	6	5	4

7 0

Byte 3

7	6	5	4	3	2	1	0

7 0

Byte 4

3	2	1	0	11	10	9	8

7 0

Byte 5

11	10	9	8	7	6	5	4

7 0

Byte 3n

7	6	5	4	3	2	1	0

7 0

Byte 3n+1

3	2	1	0	11	10	9	8

7 0

Byte 3n+2

11	10	9	8	7	6	5	4

7 0

Byte 3n+3

7	6	5	4	3	2	1	0

7 0

Byte 3n+4

3	2	1	0	11	10	9	8

7 0

Byte 3n+5

11	10	9	8	7	6	5	4

7 0

DualBayerRG12Packed

Byte 0

11	10	9	8	7	6	5	4

7 0

Byte 1

3	2	1	0	3	2	1	0

7 0

Byte 2

11	10	9	8	7	6	5	4

7 0

Byte 3

11	10	9	8	7	6	5	4

7 0

Byte 4

3	2	1	0	3	2	1	0

7 0

Byte 5

11	10	9	8	7	6	5	4

7 0

Byte 3n

11	10	9	8	7	6	5	4

7 0

Byte 3n+1

3	2	1	0	3	2	1	0

7 0

Byte 3n+2

11	10	9	8	7	6	5	4

7 0

Byte 3n+3

11	10	9	8	7	6	5	4

7 0

Byte 3n+4

3	2	1	0	3	2	1	0

7 0

Byte 3n+5

11	10	9	8	7	6	5	4

7 0

DualBayerRG16

Byte 0

7	6	5	4	3	2	1	0

7 0

Byte 1

15	14	13	12	11	10	9	8

7 0

Byte 2

7	6	5	4	3	2	1	0

7 0

Byte 3

15	14	13	12	11	10	9	8

7 0

Byte 4

7	6	5	4	3	2	1	0

7 0

Byte 5

15	14	13	12	11	10	9	8

7 0

Byte 6

7	6	5	4	3	2	1	0

7 0

Byte 7

15	14	13	12	11	10	9	8

7 0

Byte 4n

7	6	5	4	3	2	1	0

7 0

Byte 4n+1

15	14	13	12	11	10	9	8

7 0

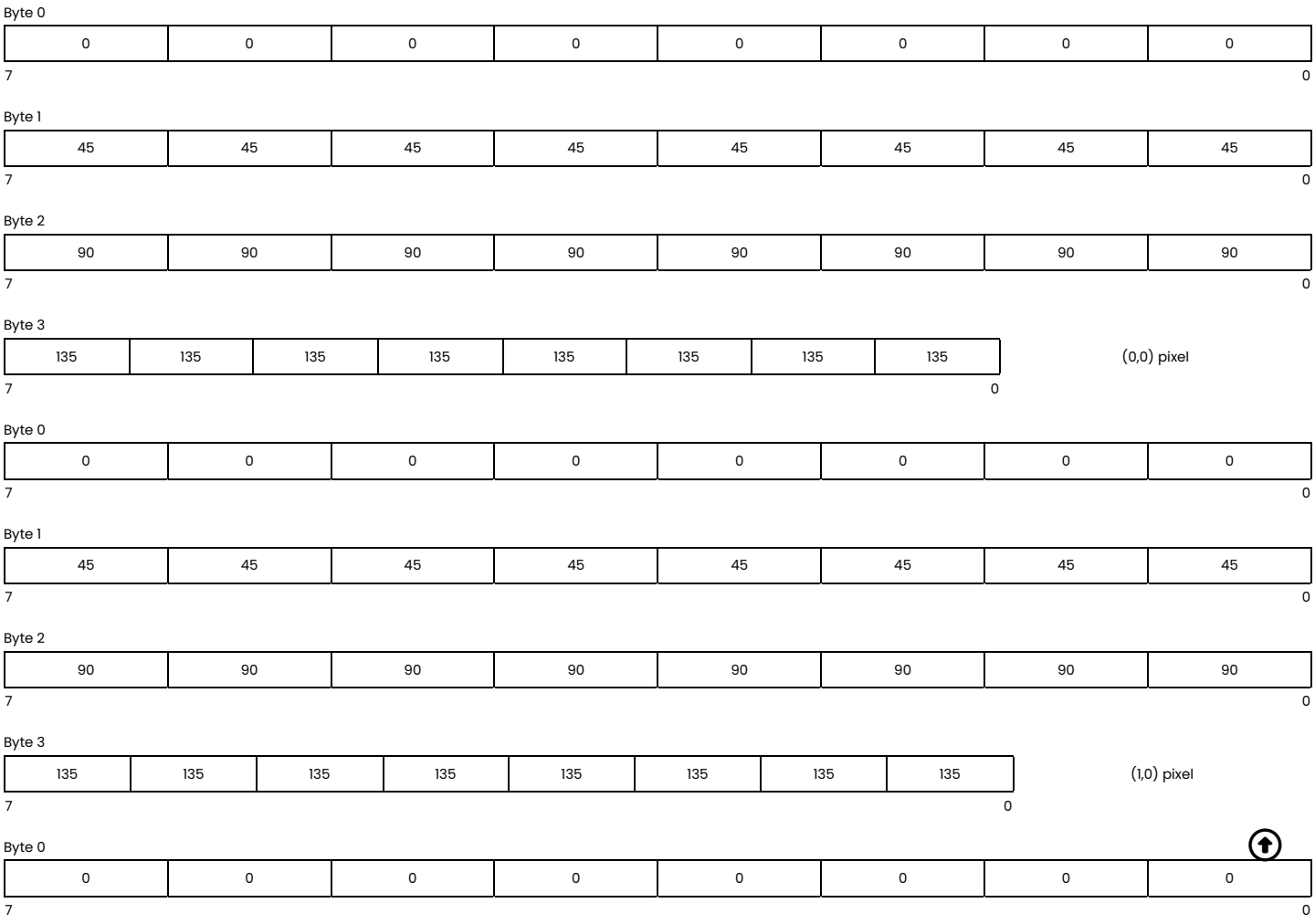
Byte 4n+2

Polarized pixel formats

These pixel formats are for LUCID's polarized-light cameras.

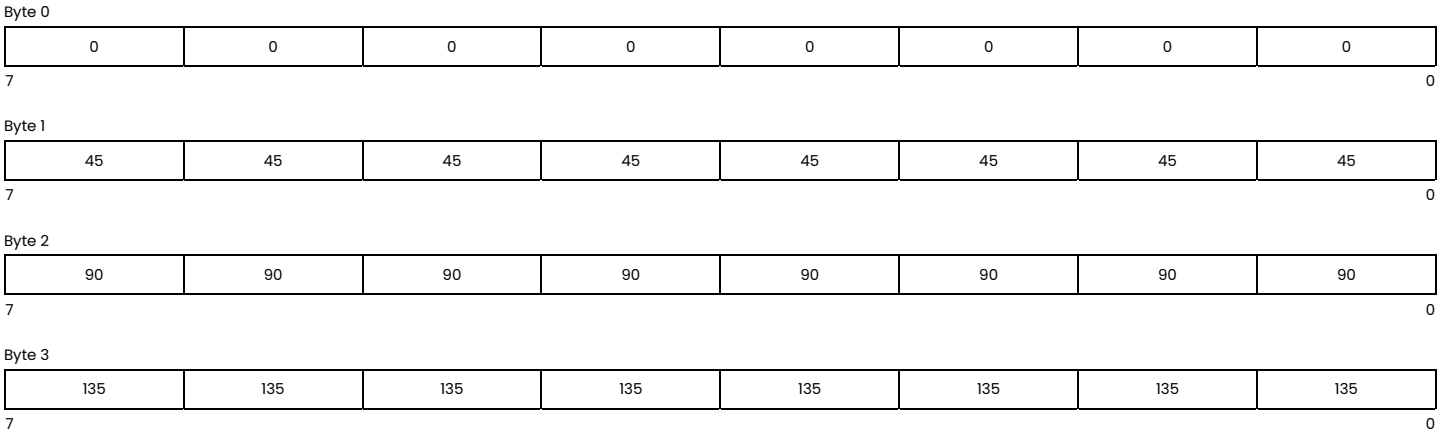
PolarizedAngles_0d_45d_90d_135d_RGB8

32 bits per pixel



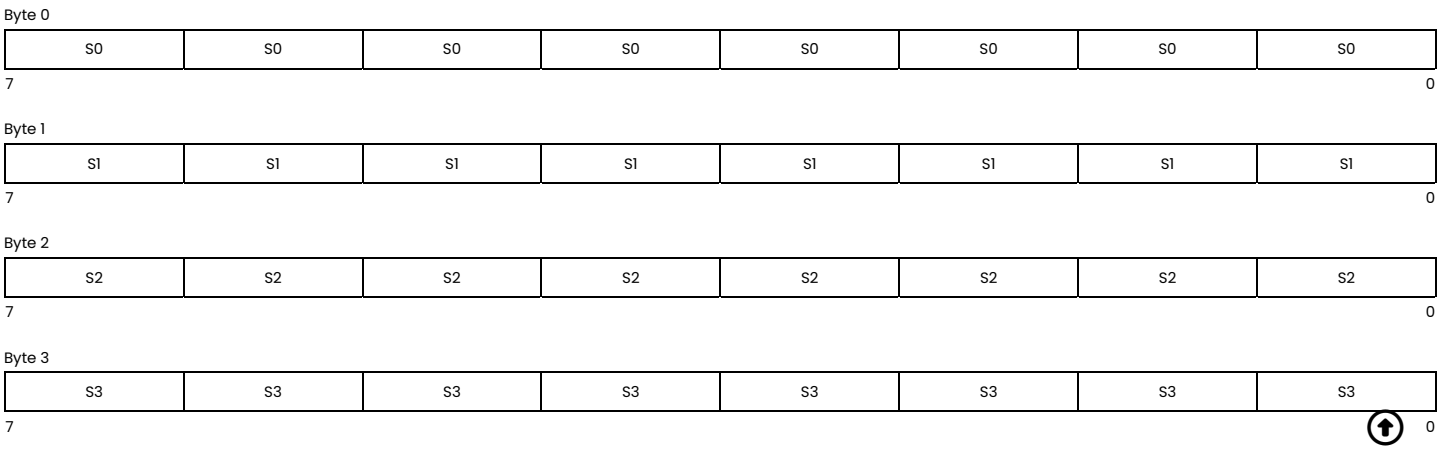
PolarizedAngles_0d_45d_90d_135d_Mono8

32 bits per pixel



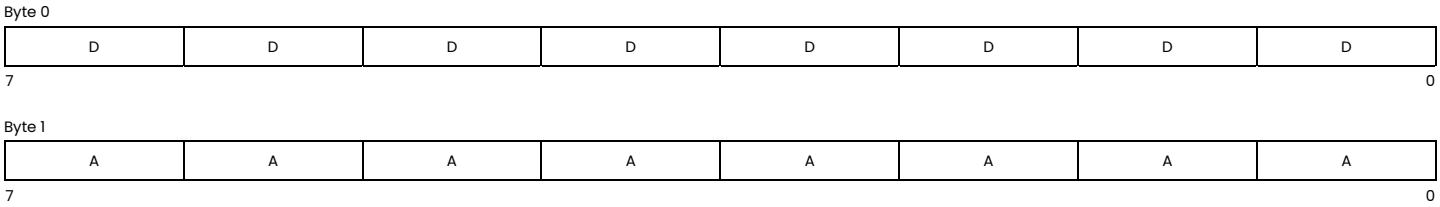
PolarizedStokes_S0_S2_S1_S3_Mono8

32 bits per pixel



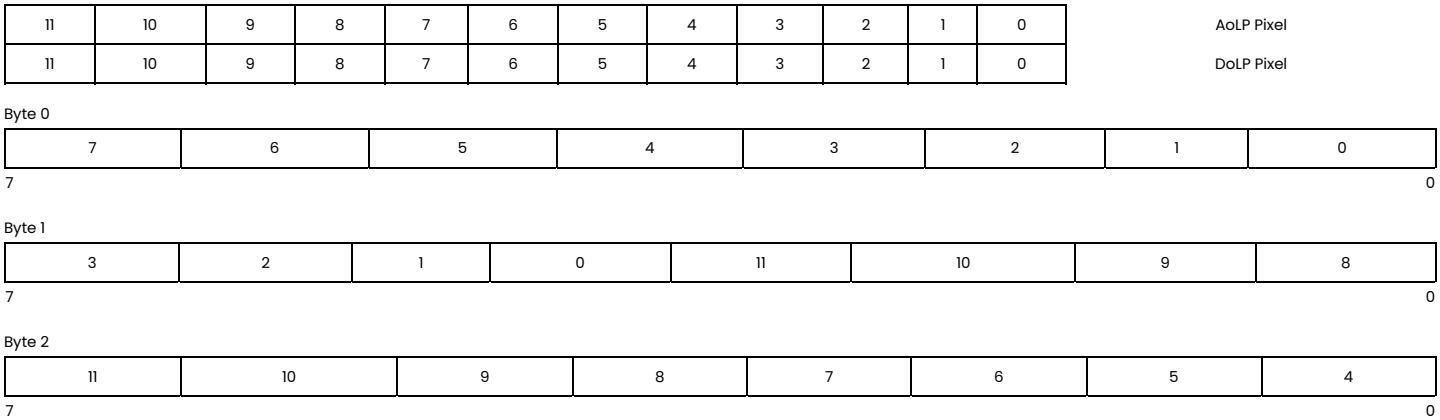
PolarizedDolpAolp_Mono8

16 bits per pixel



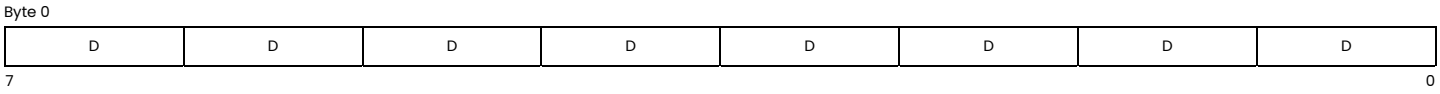
PolarizedDolpAolp_Mono12p

24 bits per pixel



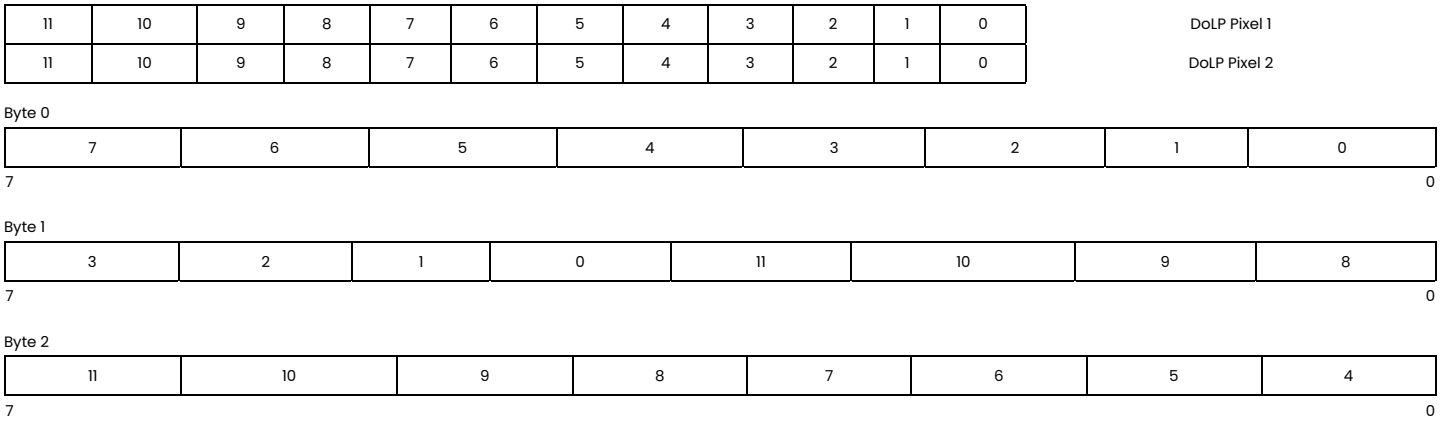
PolarizedDolP_Mono8

8 bits per pixel



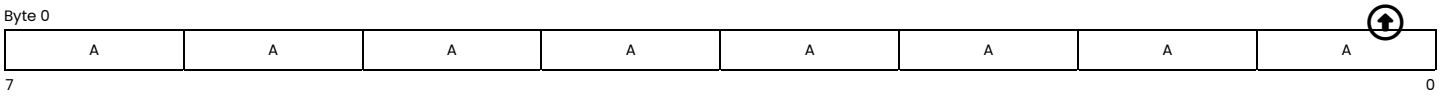
PolarizedDolP_Mono12p

12 bits per pixel



PolarizedAoLP_Mono8

8 bits per pixel



PolarizedAoLP_Mono12p

12 bits per pixel

11	10	9	8	7	6	5	4	3	2	1	0	AoLP Pixel 1
11	10	9	8	7	6	5	4	3	2	1	0	AoLP Pixel 2

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

3	2	1	0	11	10	9	8
7							0

Byte 2

11	10	9	8	7	6	5	4
7							0

Mono pixel formats

Mono pixel formats supported by LUCID cameras are shown below.

Mono8

8 bits per pixel

Byte 0

7	6	5	4	3	2	1	0
7							0

Mono10

10 bits per pixel

9	8	7	6	5	4	3	2	1	0	Pixel 1
9	8	7	6	5	4	3	2	1	0	Pixel 2

Byte 0

7	6	5	4	3	2	1	0
7							0

Byte 1

pad	pad	pad	pad	pad	pad	9	8
7							0

Byte 2

7	6	5	4	3	2	1	0
7							0

Byte 3

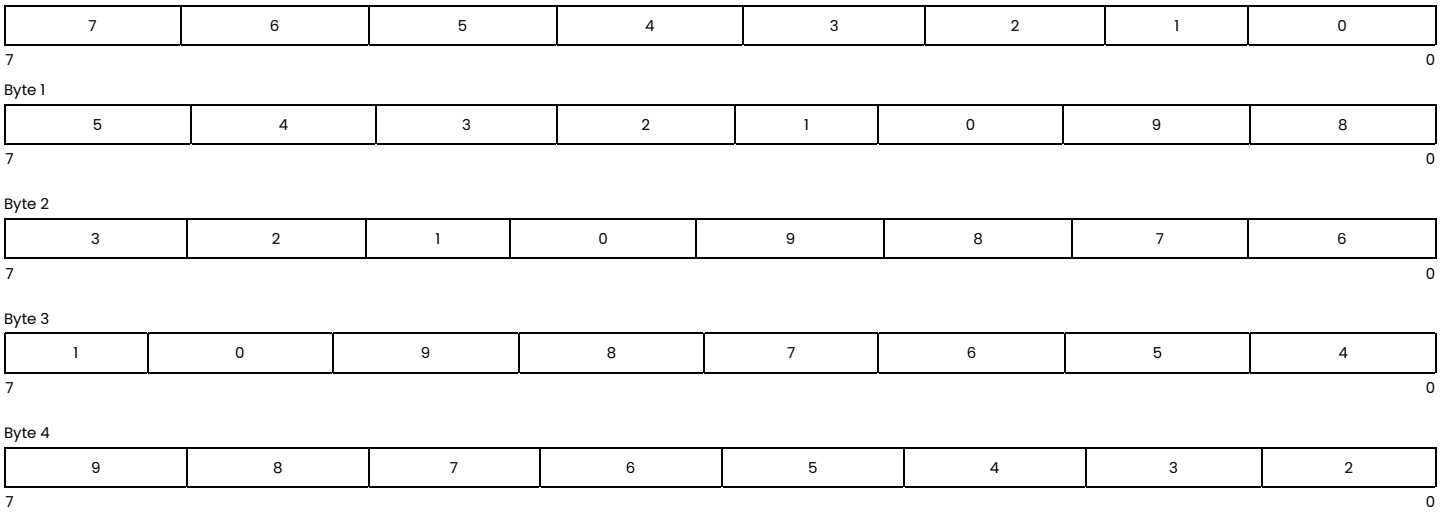
pad	pad	pad	pad	pad	pad	9	8
7							0

Mono10p

10 bits per pixel

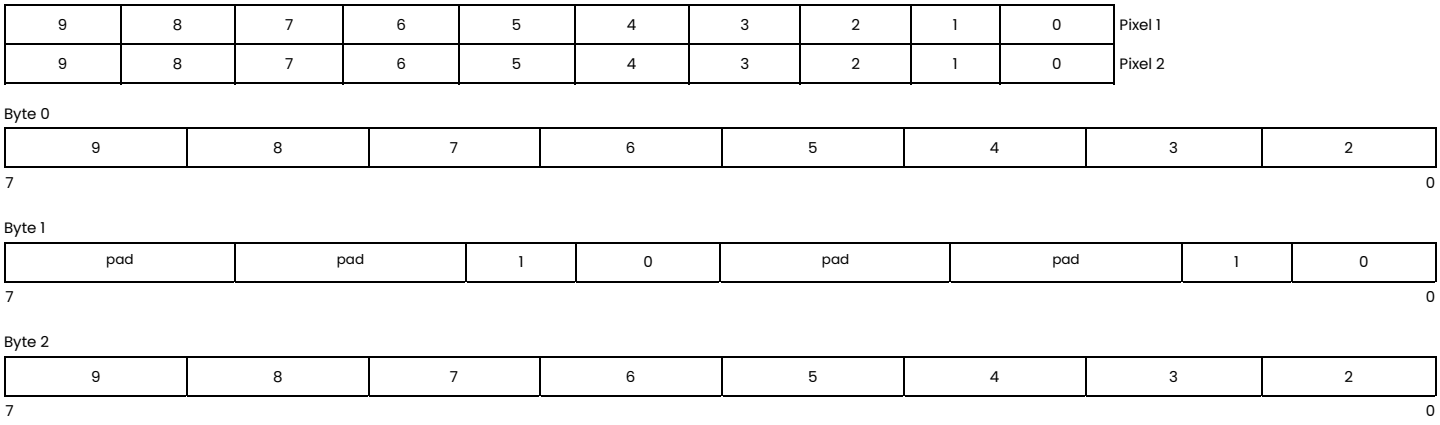
9	8	7	6	5	4	3	2	1	0	Pixel 1
9	8	7	6	5	4	3	2	1	0	Pixel 2
9	8	7	6	5	4	3	2	1	0	Pixel 3
9	8	7	6	5	4	3	2	1	0	Pixel 4

Byte 0



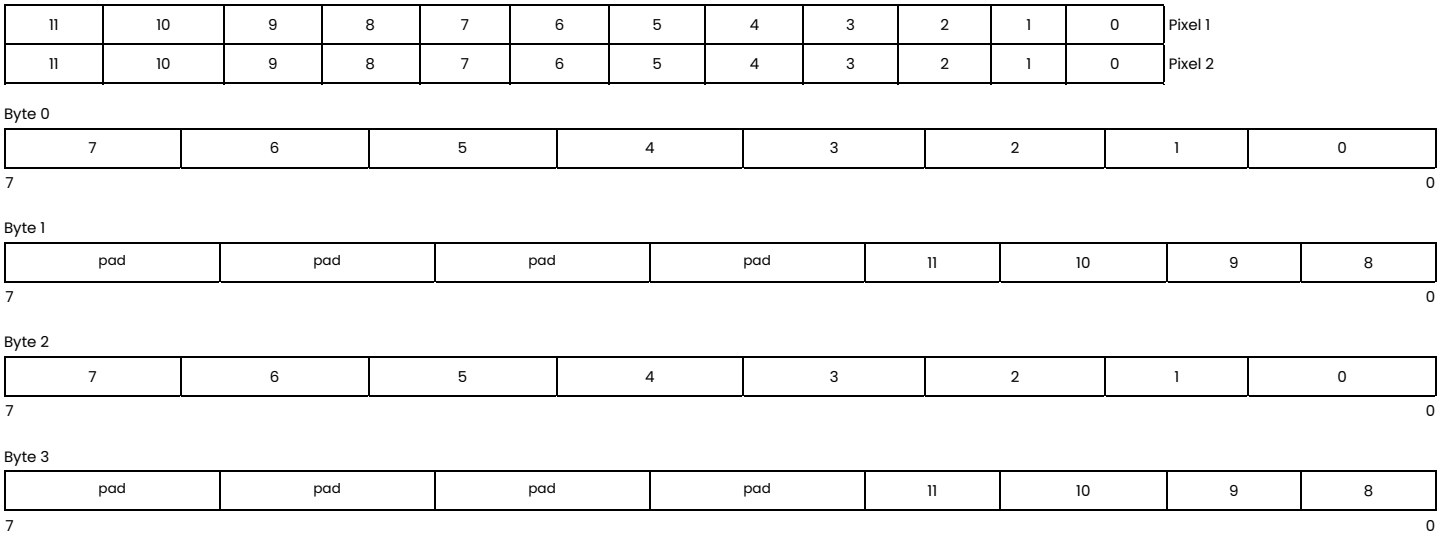
Mono10Packed

10 bits per pixel



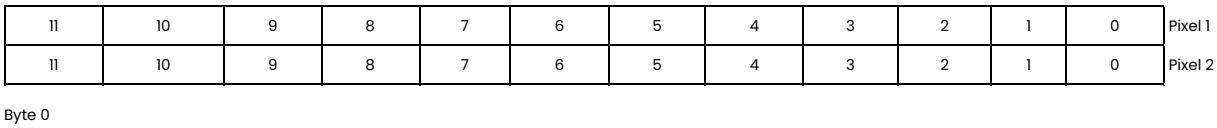
Mono12

12 bits per pixel

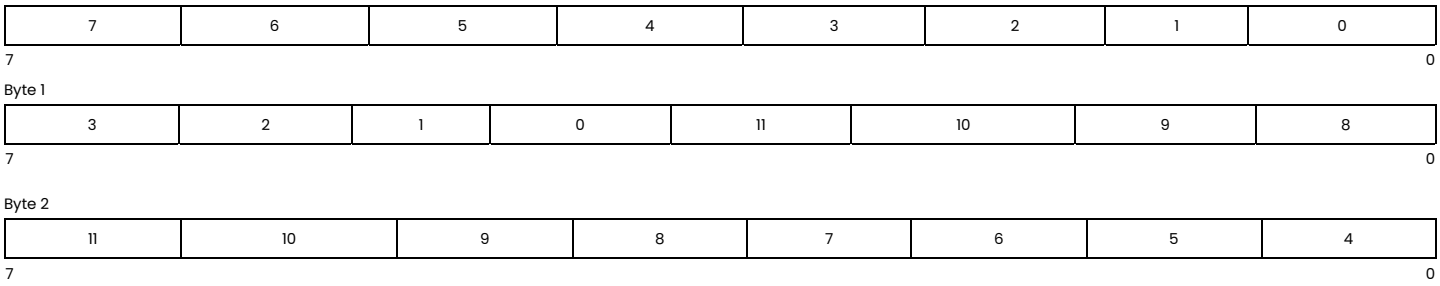


Mono12p

12 bits per pixel

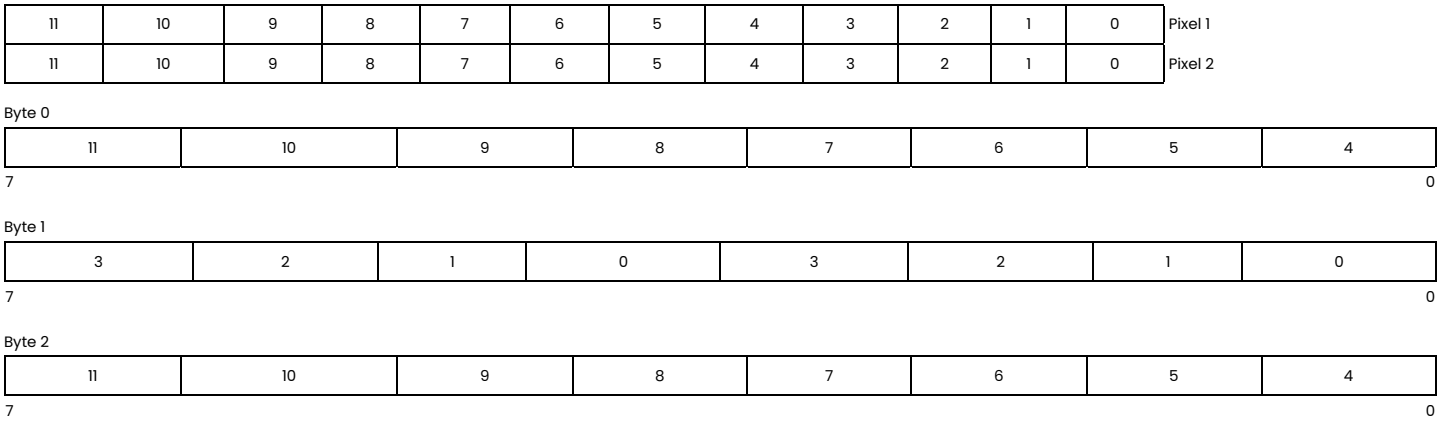


Byte 0



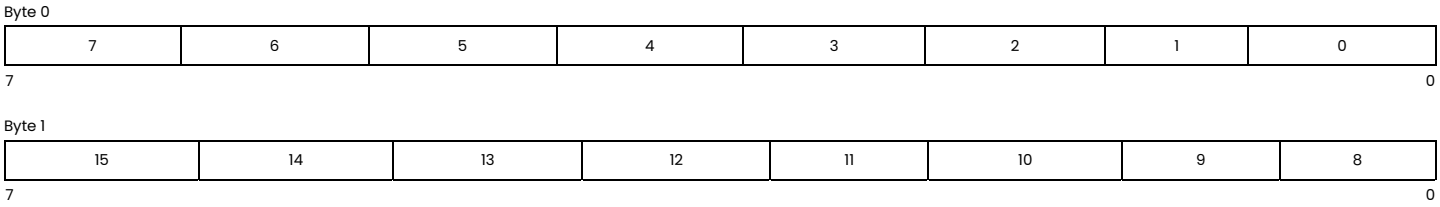
Mono12Packed

12 bits per pixel



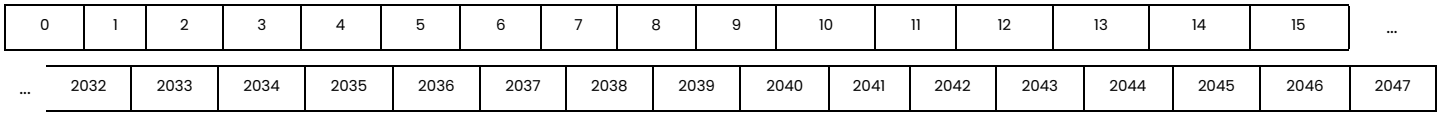
Mono16

16 bits per pixel



Linescan ROI

Linescan ROI

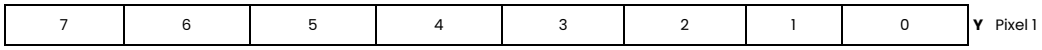


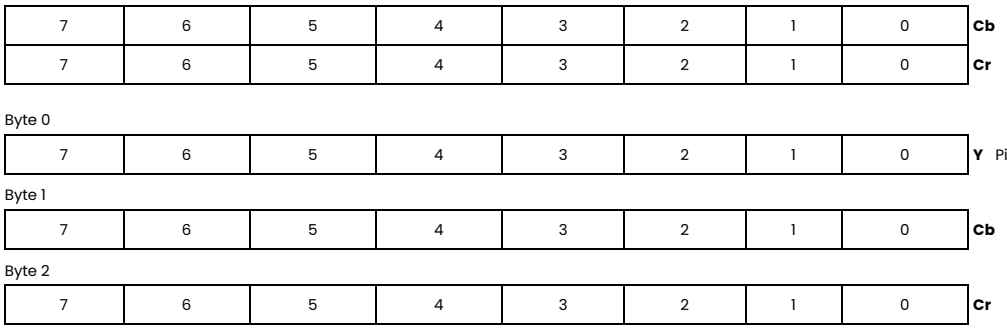
YCbCr Pixel Formats: Basic and subsampled

YCbCr Pixel Formats – Basic

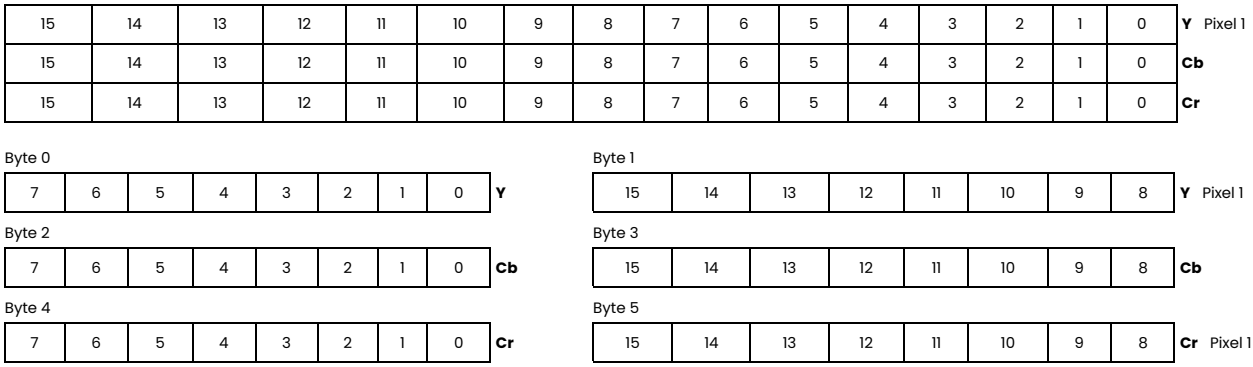
All basic (non-subsampled) YCbCr pixel formats capture one luma (Y) component and two chroma components (Cr and Cb). These basic pixel formats are also known as YCbCr444 formats. Note: For YUV8, YUV16, and YUV24, substitute U for Cb (and V for Cr) in the diagrams below.

YCbCr8

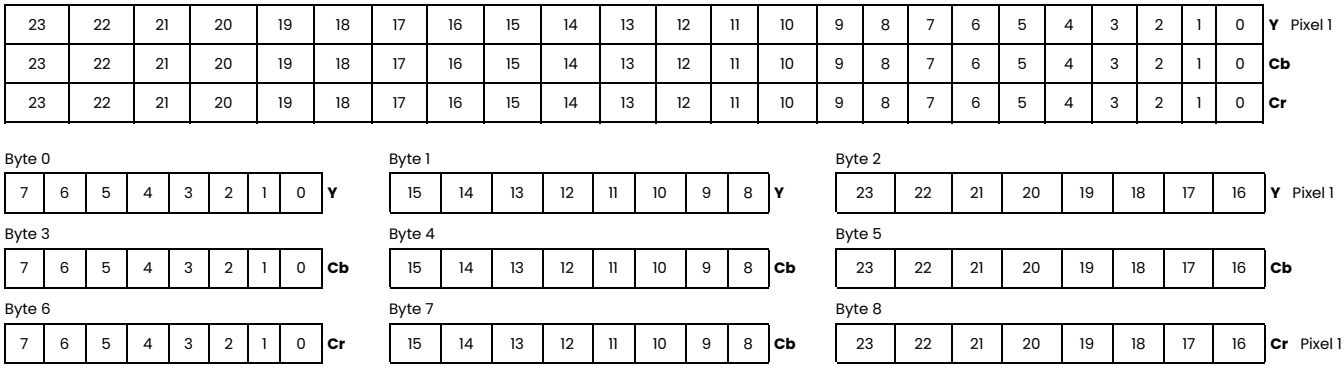




YCbCr16



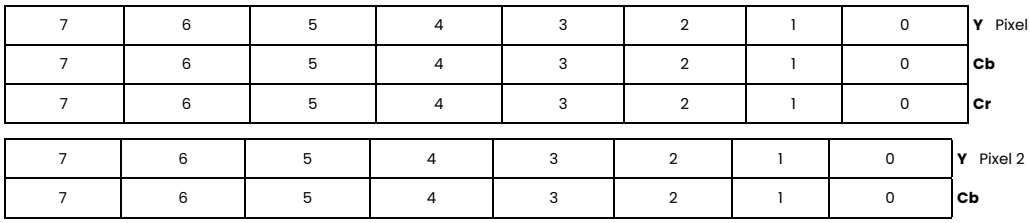
YCbCr24



YCbCr Pixel Formats – Subsampled

In subsampled pixel formats, the luma value in each pixel is sampled (read), but the chroma values are not. The purpose of subsampling is to reduce the amount of data. “422” subsampling means that that two samples each of Cb and Cr is recorded for every four samples of luma. Similarly, “411” subsampling means that that one sample each of Cb and Cr is recorded for every four samples of luma. In each subsampling scheme, Y1, Y2, Y3, and Y4 are luma values from pixels 1 to 4 respectively. Cb and Cr are averages calculated from the available pixels. YCbCr422_8_CbYCrY means that “422” subsampling is being used with 8 bits per channel, and the order of bytes is Y, Cb, Y, Cr. Note: The patterns shown below applies to YCbCr formats. To see the equivalent YUV formats, substitute U for Cb (and V for Cr).

YCbCr422_8_CbYCrY



7	6	5	4	3	2	1	0	**Cr**

Byte 0

7	6	5	4	3	2	1	0	**Cb** Pixel 1

Byte 1

7	6	5	4	3	2	1	0	**Y**

Byte 2

7	6	5	4	3	2	1	0	**Cr**

Byte 3

7	6	5	4	3	2	1	0	**Y**

YCbCr422_16_CbYCrY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 2
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

Byte 0

7	6	5	4	3	2	1	0	**Cb**

Byte 1

15	14	13	12	11	10	9	8	**Cb** Pixel 1

Byte 2

7	6	5	4	3	2	1	0	**Y**

Byte 3

15	14	13	12	11	10	9	8	**Y**

Byte 4

7	6	5	4	3	2	1	0	**Cr**

Byte 5

15	14	13	12	11	10	9	8	**Cr** Pixel 1

Byte 6

7	6	5	4	3	2	1	0	**Y**

Byte 7

15	14	13	12	11	10	9	8	**Y** Pixel 1

YCbCr422_24_CbYCrY

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 1
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 2
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

Byte 0

7	6	5	4	3	2	1	0	**Cb**

Byte 1

15	14	13	12	11	10	9	8	**Cb**

Byte 2

23	22	21	20	19	18	17	16	**Cb** Pixel 1

Byte 3

7	6	5	4	3	2	1	0	**Y**

Byte 4

15	14	13	12	11	10	9	8	**Y**

Byte 5

23	22	21	20	19	18	17	16	**Y**

Byte 6

7	6	5	4	3	2	1	0	**Cr**

Byte 7

15	14	13	12	11	10	9	8	**Cr**

Byte 8

23	22	21	20	19	18	17	16	**Cr** Pixel 1

Byte 9

7	6	5	4	3	2	1	0	**Y**

Byte 10

15	14	13	12	11	10	9	8	**Y**

Byte 11

23	22	21	20	19	18	17	16	**Y** Pixel 1

YCbCr411_8_CbYYCrYY

7	6	5	4	3	2	1	0	**Y** Pixel 1
7	6	5	4	3	2	1	0	**Cb**
7	6	5	4	3	2	1	0	**Cr**

7	6	5	4	3	2	1	0	**Y** Pixel 2
7	6	5	4	3	2	1	0	**Cb**

7	6	5	4	3	2	1	0	**Cr**

7	6	5	4	3	2	1	0	**Y** Pixel 3
7	6	5	4	3	2	1	0	**Cb**
7	6	5	4	3	2	1	0	**Cr**

7	6	5	4	3	2	1	0	**Y** Pixel 4
7	6	5	4	3	2	1	0	**Cb**
7	6	5	4	3	2	1	0	**Cr**

Byte 0

7	6	5	4	3	2	1	0	**Cb** Pixel 1

Byte 1

7	6	5	4	3	2	1	0	**Y1**

Byte 2

7	6	5	4	3	2	1	0	**Y2**

Byte 3

7	6	5	4	3	2	1	0	**Cr**

Byte 4

7	6	5	4	3	2	1	0	**Y3**

Byte 5

7	6	5	4	3	2	1	0	**Y4**

YCbCr411_16_CbYYCrYY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 2
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 3
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 4
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cb**
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Cr**

Byte 0

7	6	5	4	3	2	1	0	**Cb**

Byte 1

15	14	13	12	11	10	9	8	**Cb** Pixel 1

Byte 2

7	6	5	4	3	2	1	0	**Y1**

Byte 3

15	14	13	12	11	10	9	8	**Y1**

Byte 4

7	6	5	4	3	2	1	0	**Y2**

Byte 5

15	14	13	12	11	10	9	8	**Y2** Pixel 1

Byte 6

7	6	5	4	3	2	1	0	**Cr**

Byte 7

15	14	13	12	11	10	9	8	**Cr** Pixel 1

Byte 8

7	6	5	4	3	2	1	0	**Y3**

Byte 9

15	14	13	12	11	10	9	8	**Y3** Pixel 1

Byte 10

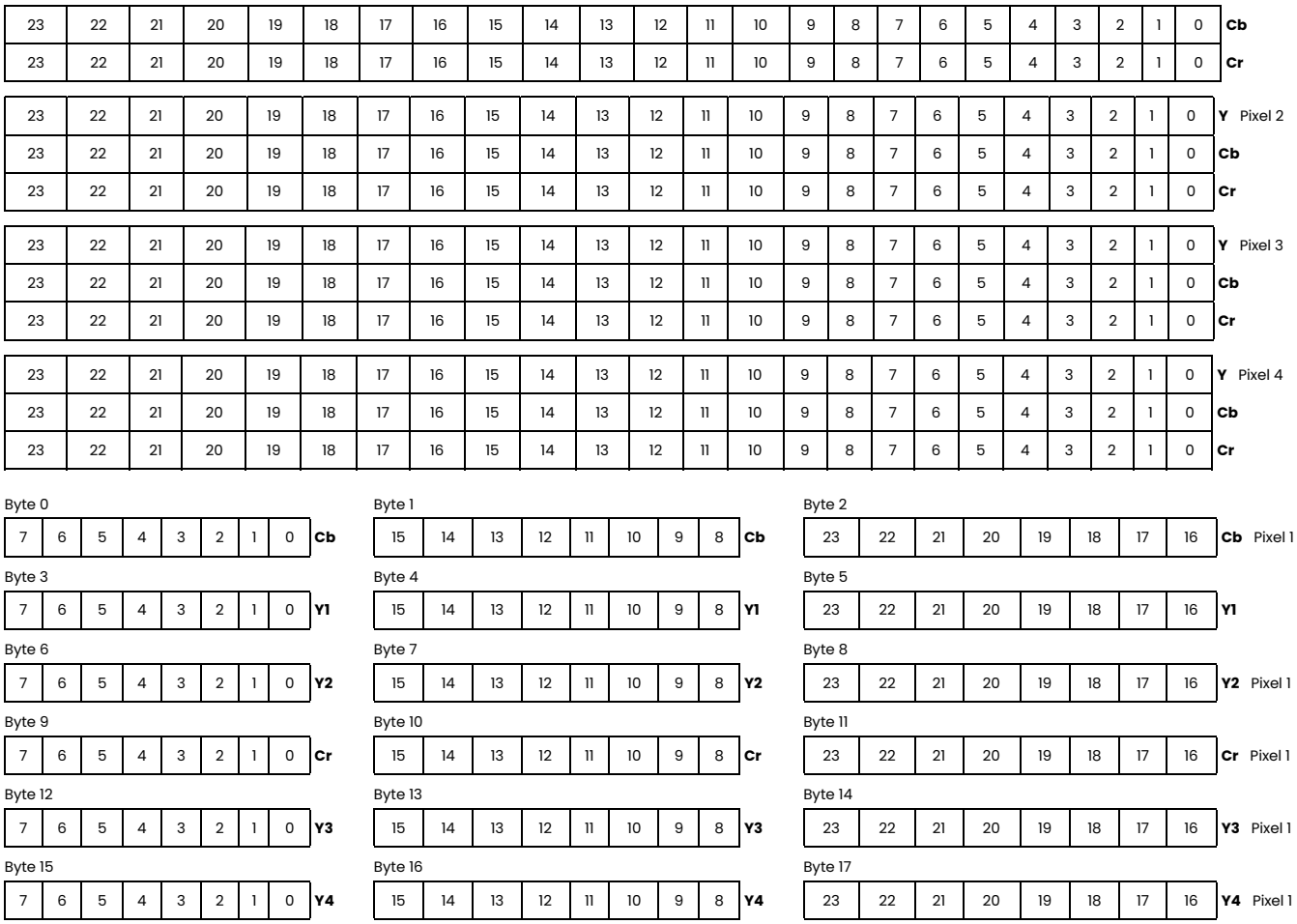
7	6	5	4	3	2	1	0	**Y4**

Byte 11

15	14	13	12	11	10	9	8	**Y4** Pixel 1

YCbCr411_24_CbYYCrYY

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**Y** Pixel 1



QOI Pixel Formats

Quite OK Image Format (QOI) is a lossless image compression method implemented in LUCID's Triton cameras. For information on how to use QOI pixel formats, see the knowledge base Using QOI with LUCID Cameras (https://support.thinklucid.com/knowledgebase/using_qoi_with_lucid_cameras/).

Examples of QOI pixel formats:

- QOI_Mono8
- QOI_BayerRG8
- QOI_RGB8
- QOI_BGR8
- QOI_YCbCr8
- QOI_YCbCr8_CbYCr

Was this helpful?

Yes No

Related Articles

Using GigE Vision RDMA Streaming Protocol in GigE Vision 3.0 (https://support.thinklucid.com/knowledgebase/using-gige-

Pixel Formats – 3D (https://support.thinklucid.com/knowledgebase/pixel-formats-

vision-rdma-streaming-protocol-in-gige-vision-3-0/)

- Pixel Formats – Line Scan (https://support.thinklucid.com/knowledgebase/pixel-formats-line-scan/)
- Using Multiple Helios Cameras Simultaneously (https://support.thinklucid.com/knowledgebase/using-multiple-helios-2-cameras-simultaneously/)

3d/)

- Using QOI with LUCID Cameras (https://support.thinklucid.com/knowledgebase/using_qoi_with_
- Shading Correction Control (Atlas10) (https://support.thinklucid.com/knowledgebase/shading-correction-control-atlas10/)

Didn't find your answer? @ Contact Us

Using QOI with LUCID Cameras (https://support.thinklucid.com/knowledgebase/using_qoi_with_lucid_cameras/)

Pixel Formats – Line Scan (https://support.thinklucid.com/knowledgebase/pixel-formats-line-scan/)

Looking to purchase our cameras?
Visit the LUCID Webstore at thinklucid.com (https://thinklucid.com)