U8500 Linux Imaging Sub-System Software Architecture Specification

Purpose	Presents the U8500 Linux Imaging sub-system software architecture specification. Details the available functionalities and SW architecture at the STELP level.
Document Status	Approved
Date	August, 2010
Document Version	V1.0

Legal Information

Disclaimer

The contents of this document are subject to change without prior notice. ST-Ericsson makes no representation or warranty of any nature whatsoever (neither expressed nor implied) with respect to the matters addressed in this document, including but not limited to warranties of merchantability or fitness for a particular purpose, interpretability or interoperability or, against infringement of third party intellectual property rights, and in no event shall ST-Ericsson be liable to any party for any direct, indirect, incidental and or consequential damages and or loss whatsoever (including but not limited to monetary losses or loss of data), that might arise from the use of this document or the information in it.

ST-Ericsson and the ST-Ericsson logo are trademarks of the ST-Ericsson group of companies or used under a license from STMicroelectronics NV or Telefonaktiebolaget LM Ericsson.

All other names are the property of their respective owners.

Trademark List

All trademarks and registered trademarks are the property of their respective owners.
In addition to the generic statement above, please fill in all trademarks and registered trademarks that could be identified. See examples below.

Microsoft® Microsoft is a registered trademark of Microsoft Corporation in the United States and/or other countries.

OBEX™ OBEX is a trademark of Infrared Data Association.

Version	Release Date	Writer	Updates
V0.0	June29, 2010	Eric Auger, PY Taloud	This document is mainly derived from Mont-Blanc SAS. Updates from this SAS will also be added as and when available.
V0.1	July 19, 2010	Preetika BHASIN	Creation of first draft.
V1.0	August 16, 2010	Preetika BHASIN	Updated with review comments

Table 1 Document History

Contents

1 References

Category	Title	Version	Editors
Standard	/1/ I2C Bus Specification	2.1	Philips Semiconductor
Standard	/2/ SMIA 1.0 Part 2, CCP2 Specification, June 2004	1.0	SMIA
Standard	/3/ SMIA 1.0 Part1, Functional Specification, June 2004	1.0	SMIA
Standard	/4/ SMIA 1.0, Introduction and overview, June 2004	1.0	SMIA
SMIA	/5/ SMIA Application Note: Video Timing	0.972	SMIA
Standard	/6/ ISO/IEC 10918-1, Information Technology-Digital compression and coding of continuous-tone still images, requirements and guidelines, JPEG Standard	1993	ISO/IEC
Standard	/7/ JPEG File Interchange Format (JFIF), version 1.02, Sept 1992	Sept 1992	E. Hamilton
Standard	/8/ Japan Electronic Industry Development Association, Digital Still Camera Image File Format Standard (Exchangeable Image File Format for digital still cameras: EXIF), version 2.1, June 1998		
Standard	/9/ Japan Electronic Industry Development Association, Design Rule for Camera File System, v1.0, JEIDA-49-2-1998, DCF standard	Dec 1998	
IQ	/10/ Generic Image Quality Tuning Procedure	Dec 2006	ST-Ericsson
OMX	/11/ OpenMax IL Spec 1.1.2	1.1.2	Khronos
OMX	/12/ OpenMAX IL Imaging extension	Nov 2008	OMXIL Imaging TSG
OMX	/13/ Extended Callback Events	Feb 6, 2009	Khronos
HW	/14/ STn8500 SIA Hardware Architecture Specification	V1.8	ST-Ericsson
Video	/15/ STn8500 SVA HW Overview	DV0.3	ST-Ericsson
Linux	/16/ ST Allocation API Specification	DV0.2	ST-Ericsson
Linux	/17/ ST MMIO API Specification	1.0	ST-Ericsson
RM	/18/ OMX IL RM Architecture Note For MontBlanc	DV0.4	ST-Ericsson
B2R2	/19/ Mont-Blanc B2R2 Image/Video Post-Processing	V0.2	ST-Ericsson

	Application Note		
Power	/20/ Mont Blanc Power Tree	V0.6	ST-Ericsson
MM Overview	/21/ Mont Blanc Multimedia System Description	V0.95	ST-Ericsson
Video	/22/ Video Sub-System SAS	DV0.5	ST-Ericsson
eSRAM	/23/ eSRAM Application Note	0.94	ST-Ericsson
Flash	/24/ Flash API Specification	0.3	ST-Ericsson
OMX	/25/ OMX IL Application Note 0318 (“Race Conditions”)	Oct 9, 2008	Khronos
Camera	/26/ MBL Camera Footprint xls sheet	0.1	ST-Ericsson
JPEG	/27/ TIFF 6 Specification	Rev 6	
RM	/28/ Video/Camera Visual Resource Management Slide Set (ppt)	April 2010	ST-Ericsson
Camera	/29/ U8500 Imaging Android Addendum	0.1	ST-Ericsson
Symbian	/30/ Symbian Foundation Extensions	1.2	Symbian Foundation

Table 2 References

2 Acronyms and Terms

2.1 Glossary

Term	Definition
3A	AEC + AWB + AF
AE[C]	Auto Exposure Control
AF	Auto-focus
APE	Application Processor Engine
API	Application Programming Interface
A:R	Aspect Ratio
[A]WB	[Auto-]White-Balance
B_2R_2	Blitter Blender Rotate Resize
BML	Bayer Memory Load
BMS	Bayer Memory Store
BPP	Bit Per Pixel
CAF	Continuous Auto-Focus
CCD	Charged Couple Device
CCI	Camera Control Interface, I2C type interface
CCP	Compact Camera Port (serial data link)
CDG	Camera Data Grabber
CE	Color Engine
CFAI	Color filter Array Interpolation
CM	Component Manager
CRM	ISP8500 Clock and Reset Manager
DB	Database (Resource Management)
DD	Device Driver
DIS	Digital Image Stabilization
DOF	Depth Of Field
DPCM	Differential Pulse Code Modulation
DPOF	Digital Print Order Format
DPS	Dynamic Power Switching
DSA	Direct Screen Access
DVFS	Dynamic Voltage and Frequency Scaling
DZ	Digital Zoom (aka. Digizoom)
EBD	Empty Buffer Done
EDOF	Extended Depth of Field
EMC	Electro Magnetic Compatibility
ENS	Execution Network Server
eSRAM	Embedded SRAM
ETB	Empty This Buffer
EV	Exposure Value
FBD	Fill Buffer Done
FD	Face Detection
FFOV	Full Field Of View
FSP	False Synchronization Protection
FT	Face Tracking
FTB	FillThisBuffer
FW	Firmware (DSP software)
GPIO	General Purpose Input/Output
GPS	General Purpose Scaler
HAL	Hardware Abstraction Layer
HCR	Hardware Configuration Repository
HCS	Hill Climbing Search
HPLED	High Power Led
HSEM	Hardware Semaphore
I2C	Inter Integrated Circuit Serial Bus (industry standard 2-wire serial protocol)

[I]LED	Indicator Light Emitting Diode
IPP	Image Preprocessor Pipeline
ISL	Intelligent Status Line
ISP	Image Signal Processor
ISR	Interrupt Service Routine
IT	Interrupt
Itld	Interleaved
IV	Image/Video (vocable applied to an OMX Processor)
JPEG	Joint Photographic Experts Group
LED	Light Emitting Diode
LVDS	Low Voltage Differential Signaling
MB	Macroblock
MCDE	Multi Controller Display Engine
MECC	Motion Estimation & Compensation HW unit
MM	Multimedia
MS	Multimedia Sharing application
MTF	Modulation Transfer Function
ND	Neutral Density [filter]
NMF	Nomadik Multiprocessing Framework
OMX	OpenMax
OPB	Other Port Domain
OpenWFC	Khronos API for 2D composition
OT	Output Timing Domain and object tracking
OTF	On-The-Fly
OZ	Optical Zoom
PFF	PIF FIFO FULL
PHY	Physical Interface
PLL	Phase Locked Loop
PRCMU	Power Reser and Clock Management Unit
PRM	Power Resource Manager
PRY	YUV / RGB Packer (ISP block)
QoS	Quality Of Service
RE	Reconstruction Engine
REC	Red Eye Correction
RED	Red Eye Detection
RER	Red Eye Reduction
RLE	run-length-encoding
RME	Resource Management Engine
RML	RGB Memory Load
ROFS	Read Only File System
ROI	Region Of Interest
Rx	Receiver
SDG	Sensor Data Grabber
SGA	Nomadik Smart Graphic Accelerator
SMIA	Standard Mobile Imaging Architecture
SMPS	switch mode power supply
SVA	Nomadik Smart Video Accelerator
SWIS	See What I See
TBD	To Be Done
UC	Use Case
(G)UI	(Graphical) User Interface
VCM	Voice Coil Motor
VE	Video Engine
VF	Viewfinder
VFPN	Vertical Fixed Pattern
VPB	Video Port Domain
VPP	Video Post-Processor HW unit
VT	Video Telephony or also video timing (domain) as opposed to output timing (domain)

WOI	Window Of Interest
ZOI	Zone Of Interest

Table 3 Glossary

2.2 Prevalent Resolutions

The prevalent resolutions are listed below along with their aspect ratio. Common video graphic aspect ratios are 4:3 (1.33:1) and 16:9 (1.78:1) whereas still aspect ratios are 4:3 and 16:9.

Acronym	Resolution	Aspect Ratio
Sub-QCIF	128x96	4 :3
QQVGA	160x120	4 :3
QCIF	176x144	11 :9
QVGA	320x240	4 :3
CIF	352x288	11 :9
HVGA	320x480	
nHD	640x360	16 :9
qHD	960x540	16 :9
VGA	640x480	4 :3
NTSC	720x480	4:3
PAL	720x576	4:3
QSXGA	2560x2048	
QWVGA	400x224	
SVGA	800x600	
XGA	1024x768	4:3
1.3 Mpel	1280x960	4:3
720p (HD)	1280x720	16::9
1080p (full HD)	1920x1080	16:9
UXGA (2Mpel)	1600x1200	4:3
1080p (Full HD)	1920x1080	16:9
3.15 Mpel	2048x1536	4:3
5 Mpel	2592x1944	4:3
	2592x1456	16:9
8 Mpel	3264x2448	4:3
	3264x1832	16:9
12 Mpel	4000x3000 or 4096x3072	4:3
	4000x2248	16:9
18 Mpel	5024x3768	4:3
20 Mpel	5280x3960	4:3

Table 4 Prevalent Resolutions

2.3 Prevalent Formats

The image formats referred to in this document are listed below.

Buffer Type	OMX_COLOR_FORMATTYPE	
Raw 8	OMX_COLOR_FormatRawBayer8bit	8 bit raw bayer pattern camera format
Raw 12	OMX_COLOR_FormatRawBayer12bit	12 bit raw bayer pattern camera format
YUV 422 interleaved raster	OMX_COLOR_FormatCbYCrY	Raster format, UY0VY1 (ie. CbYCrY), each component taking 8 bits
YUV420 Packed Planar	OMX_COLOR_FormatYUV420PackedPlanar	3 separate planes for Y, U, V packed in the same buffer to constitute a slice. Each slice contains a plane of Y, U and V. Also refered to as I420 (blue chroma first). or YV12 (red chroma first).
YUV420 Planar	OMX_COLOR_FormatYUV420Planar	Same as packed planar except that each plane is transferred in its entirety (Not Supported) I420 ot YV12.
YUV420 MB	OMX_COLOR_FormatYUV420MBPackedSemiPlanar	ST Proprietary Macro-block tiled format
ARGB 32bits	OMX_COLOR_Format32bitARGB8888	24 bits per pixel ABGR format with colors stored as Alpha 31:24, Blue 23:16, Green 15:8, and Red 7:0.
RGB 24bits	OMX_COLOR_Format24bitRGB888	24 bits per pixel RGB format with colors stored as Red 23:16, Green 15:8, and Blue 7:0.
RGB16bits	OMX_COLOR_Format16bitRGB565	6 bits per pixel RGB format with colors stored as Red 15:11, Green 10:5, and Blue 4:0.
YUV 420 SP NV21	OMX_COLOR_FormatYUV420SemiPlanarNV21	NV 21 format with Y in the first plane and V and U in that order in the second plane

Table 5 Prevalent Video/Camera Formats

2.4 Conventions or Legends used in the document

2.4.1 Open Points

All points highlighted in blue are under discussion.

2.4.2 Extensions

SHAI 1.2 are Symbian Foundation extensions.
Then there are also some vendor specific OMX extensions which are introduced in this document.

2.4.3 Colors in Tables

OMX_IndexConfigxxx	Description	Khronos	Extensions
Supported			
Not supported			

2.4.4 Drawing Conventions

The drawing conventions for Omx components which are followed in the remaining sections of this document are described here.

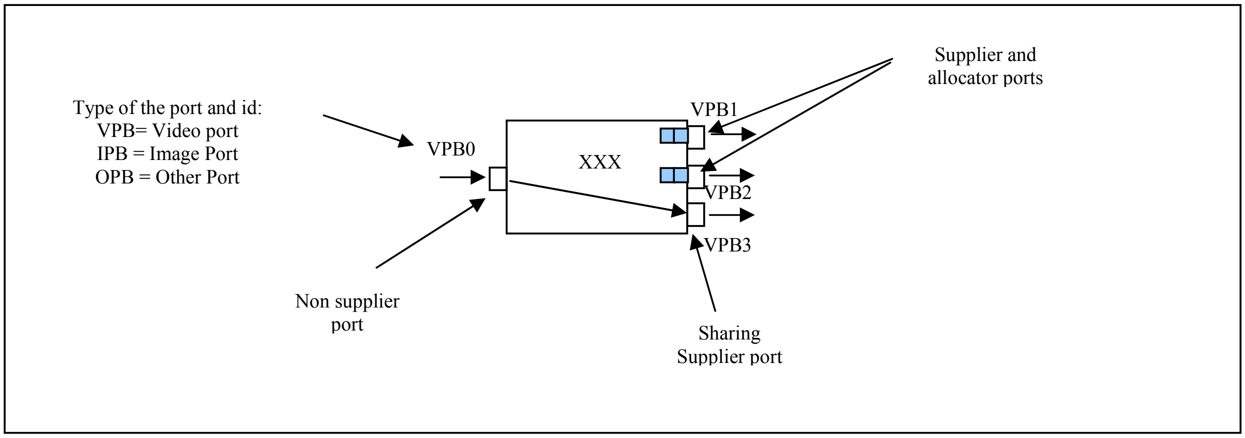


Figure 1. Drawing Conventions

3 Introduction

This document presents the overall camera sub-system SW architecture for the Mont-Blanc Linux platform. It especially focusses on the OpenMax components and the below layers.

By camera, it is meant the functional system composed of

- the image reconstruction pipeline (embedded ISP8500),
- the sensor module: sensor, actuators (focus, mechanical shutter, iris, ND filter, …)
- the flash ICs (Xenon, high power led, privacy indicators).

4 Platform Overview

The STn8500 processor platform powers multimedia computers to access more content anywhere and to enjoy connected services instantly, by means of a unique association of state-of-the-art 3G modem, high- performance smart multimedia combined with the latest ARM® Cortex-A9 dual core and advanced power management technology.

The key features of the STn8500 mobile processor are:

- ARM® Cortex-A9 SMP dual core with Neon extension up to 600MHz, providing general-purpose high-performance computing power
- A smart video accelerator for HDTV 720p video encoding and decoding in real-time, together with HDMI v1.3 support,
- A smart imaging accelerator, providing 12Mpixel image snapshots with DSC-like quality, low-latency 260MPixel/s capture speed,
- A smart graphics accelerator for life-like animated UI and immersive gaming, which complies with OVG v1.0, OGL-ES v1.2 and v2.0,
- A smart audio framework supporting any speech/audio decoders and encoders with versatile mixing and effects combination, while enabling 100h extended playback time,
- On-chip ROM and eSRAM memory devices, including a 6 Mbit buffer

The ARM® Cortex-A9 SMP dual processor incorporates two instances that implements the ARM® Cortex architecture v7. It supports the Thumb-2 instruction set, TrustZone® computing, and Neon SIMD extensions.

The STn8500 has two main functional domains:

- the MODEM sub-system
- the APE sub-system

The CortexA9 ARM Dual Core is part of the APE sub-system.

It has a separate power supply interface. The resources shared between APE and MODEM are in the VSAFE domain. The DDR memory controller and the power manager are shared resources in the VSAFE domain.

The STw4500 is the power management IC delivering all the core power supplies needed by the STn8500 SOC.

This flexible imaging engine provides real-time programmable image processing, and features:

- Optimized support for CMOS sensor, up to 12 Mpixel resolution (RGB raw Bayer input)
- For camera modules, unlimited resolution (YUV input with JPEG compression)
- On-the-fly image reconstruction up to 200 Mpixel/s
- On-the-fly image capture speed up to 260 Mpixel/s
- Fast sensor capture datapath to memory up to 3x800Mbps
- Leading-edge shot-to-shot performances (8 Mpixel in less than 1/4s)
- Auto-focus control, auto exposure control
- Automatic white balancing, contrast enhancement, brightness control
- Noise reduction, gamma correction, image sharpening
- Lens actuator control
- Flashgun plus white Led support
- Optical and digital zoom functions
- Special effects

- Image stabilization with configurable region of interest
- Ultra-low power implementation

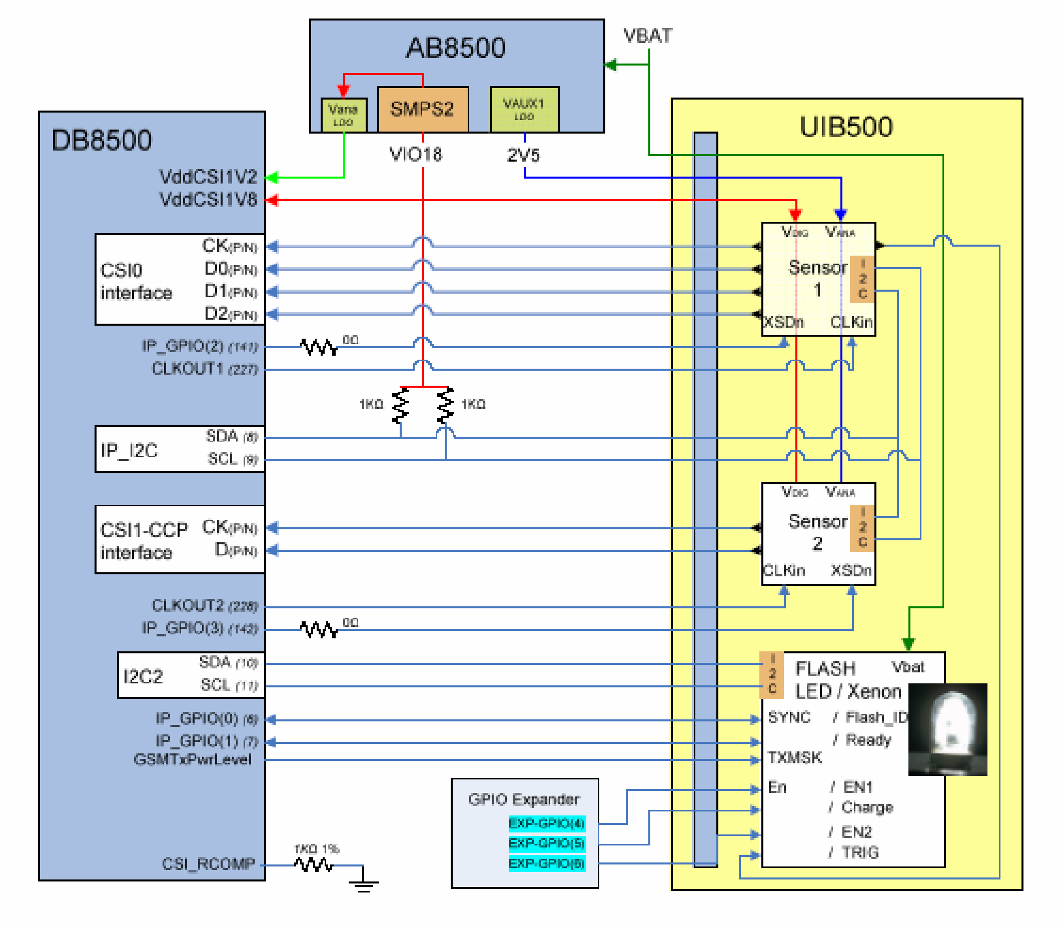
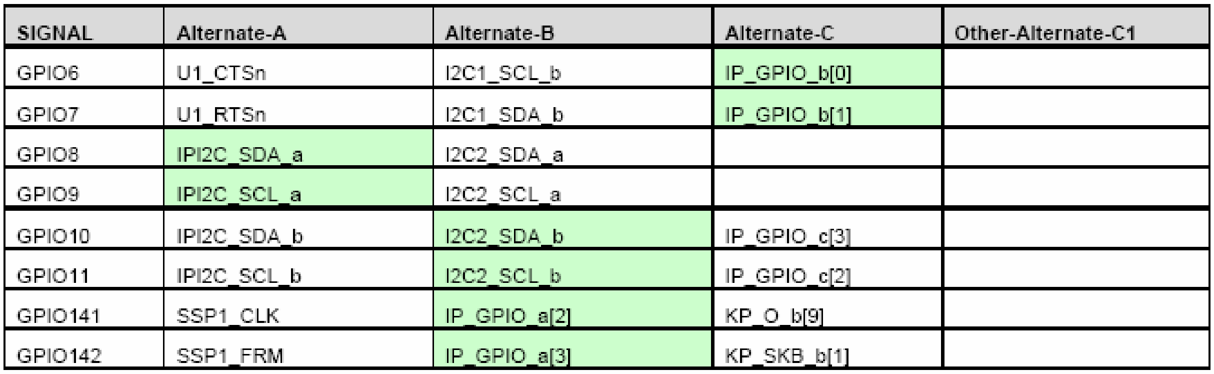


Figure 2. HREFP Camera Sensors Interface (excerpt of the Hardware Design Specification)



SIGNAL	Alternate-A	Alternate-B	Alternate-C	Other-Alternate-C1
GPIO6	U1_CTSn	I2C1_SCL_b	IP_GPIO_b[0]	
GPIO7	U1_RTSn	I2C1_SDA_b	IP_GPIO_b[1]	
GPIO8	IPI2C_SDA_a	I2C2_SDA_a		
GPIO9	IPI2C_SCL_a	I2C2_SCL_a		
GPIO10	IPI2C_SDA_b	I2C2_SDA_b	IP_GPIO_c[3]	
GPIO11	IPI2C_SCL_b	I2C2_SCL_b	IP_GPIO_c[2]	
GPIO141	SSP1_CLK	IP_GPIO_a[2]	KP_O_b[9]	
GPIO142	SSP1_FRM	IP_GPIO_a[3]	KP_SKB_b[1]	

Figure 3. HREFP Camera GPIO Muxing (excerpt of the Hardware Design Specification)

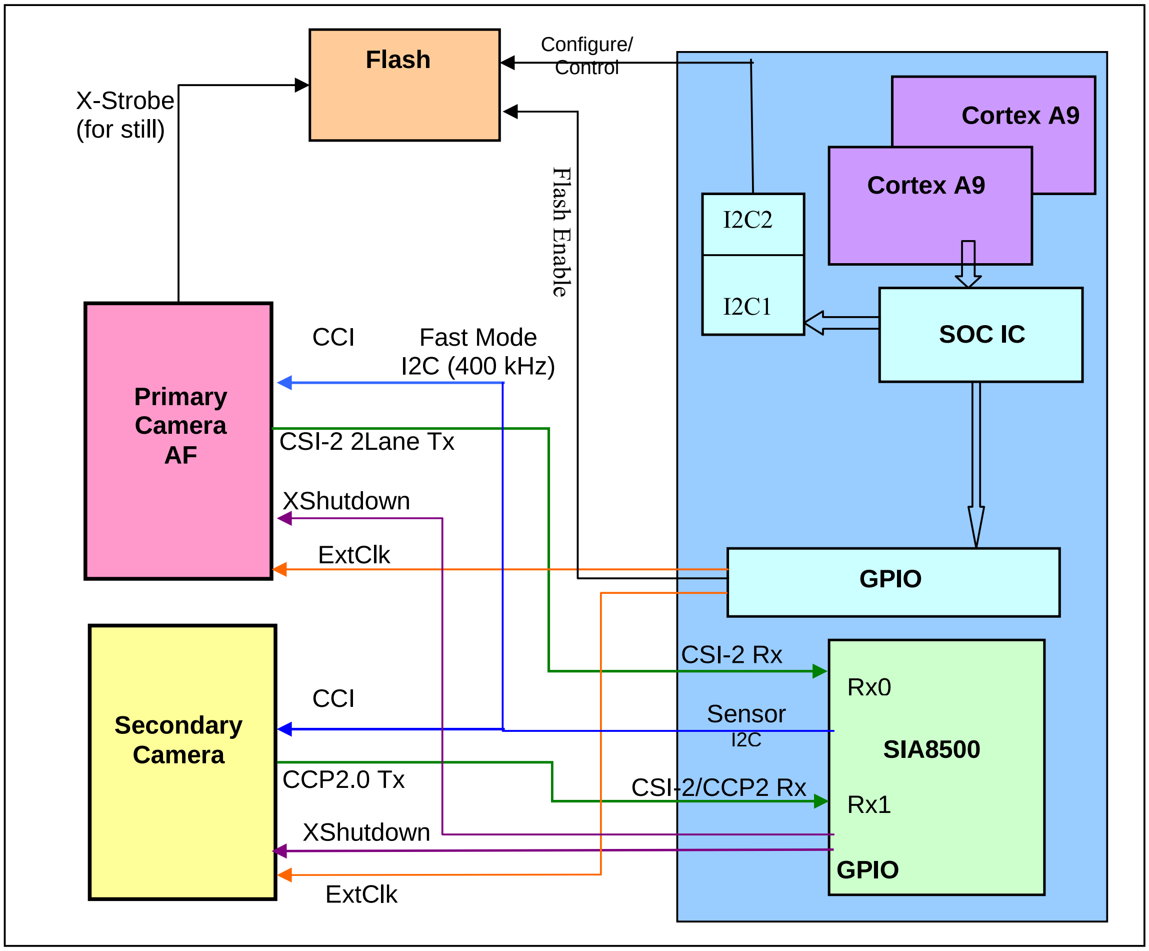


Figure 4. Platform Overview

768 KB embedded shared SRAM memory and 128 KB on chip ROM for boot and security.

Platform configurations are program or product specific. Hence for complete details of product specification please refer to customer specific documents/addendums.

5 HW Overview

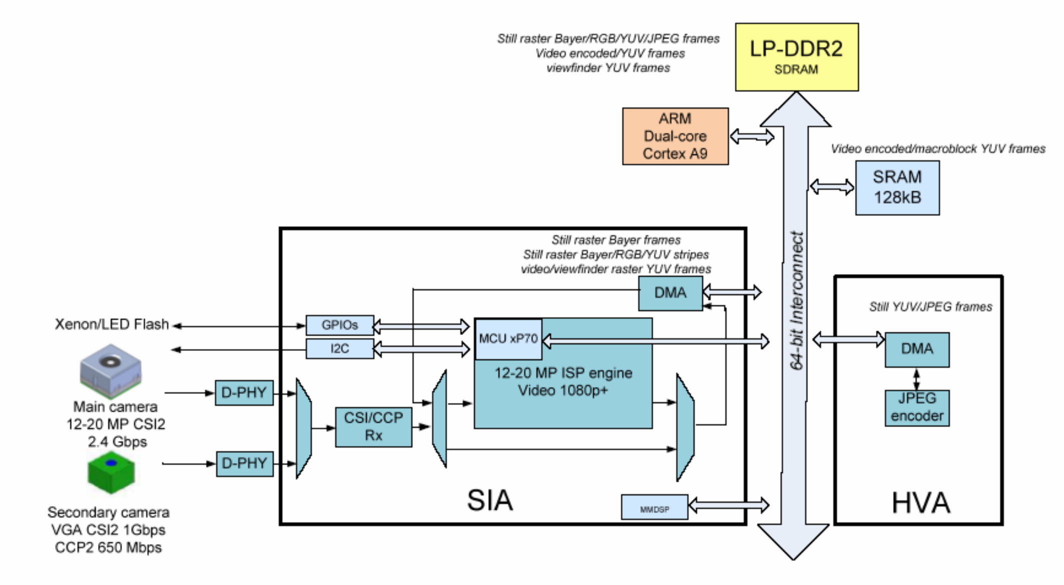
This chapter provides an overview of the Smart Imaging Accelerator HW IP.

For introduction on

- the SVA (Smart Video Accelerator),
- the B_2R_2 (Blitter Blender Rotate Resize),
- the MCDE ,

please refer to /15/ and /19/ .

This chapter provides important definitions that are mandatory for further reading this document.



The SIA (Smart Imaging Accelerator) is composed of the following main components:

- MMDSP+ sub-system, embedding a DSP executing DSP firmware.
- Image pre-processor (IPP) implementing front-end camera logic and YUV/raw-data grab (without image reconstruction).
- Image signal processor (ISP8500), implementing raw-bayer to RGB/YUV image reconstruction.
- Stabilization hardware (STBx V/H) hardware (ROT L/C)
- Direct memory access engine (DMA), optimizing data transfers for hardware modules.
- 2x STBusPlug (STP), generating optimized STBus accesses on main SOC interconnect.

5.1 MMDSP+ sub-system

The MMDSP+ sub-system is a complete, self-contained DSP-based system, featuring its own memory hierarchy (I$, D$+TCM), its own set of peripherals (DMA, interrupt controller, timers, GPIO, I2C, …), its own debug logic and interface, and its own test hardware.

The SIA is a co-processor of the STW8500's main CPU. As such, it behaves as a slave: the SIA is initialized, configured and controlled by the STW8500's main CPU. The STW8500's main CPU is named HOST in the context of the SIA.

The MMDSP firmware implements at least the following.

- The slave part of the communication with the HOST.
- The allocation/de-allocation of SIA hardware resources (Timers, …).
- The control abstraction of SIA hardware resources (Timers, …).
- Digital signal processing functions (typically for imaging but not necessarily)

5.2 IPP

The IPP IP is pictured in the following figure.

The Image Pipeline Processor (IPP) integrates two pixel pipes supporting two kinds of camera platform architectures

- The ISP implements full image reconstruction on raw-bayer data output by a sensor module.
- The YUV/JPEG Pipeline implements a reduced set of functions on YUV data output by a camera module or an external ISP co-processor; image reconstruction is handled either by the module itself (if it embeds an ISP) or by the ISP co-processor. Possibly, JPEG can be grabbed through the YUV/JPEG Pipeline, in which case no processing is done on the data.

The ISP and YUV/JPEG Pipeline cannot run concurrently. The SIA500 does not support two camera modules providing pixel data at the very same time.

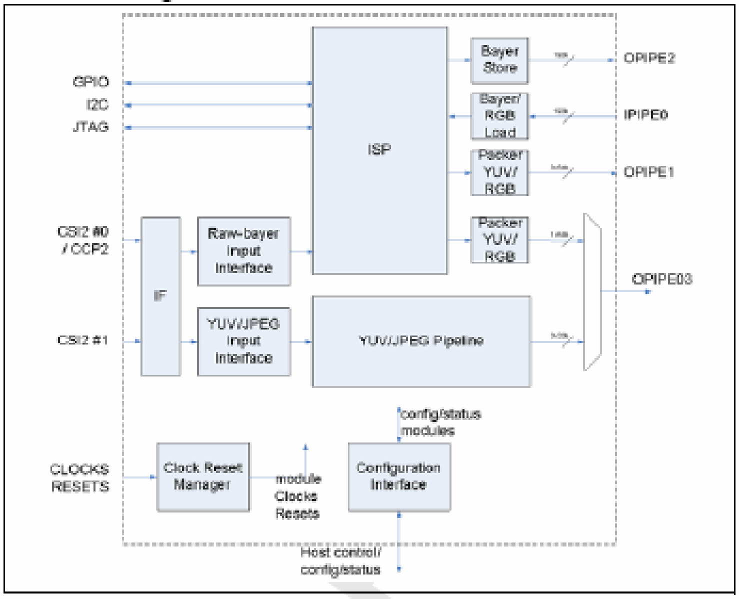


Figure 5. Image Pre-Processing block

5.3 ISP Pipeline

The Image Signal Processor, referred to as ISP in the rest of this document, includes:

- a raw-bayer to RGB/YUV pixel pipe with:
  - a twin pixel pipe handling raw-bayer image reconstruction
  - a demosaic engine
  - two parallel color restoration pixel pipes for concurrent grabs of high and low resolution frames.

- 32-bit MCU with dedicated Tighly Coupled Data Memory(TCDM) and an 8KB instruction L1 cache and dedicated 64KB local program memory, as well as a dedicated interrupt controller that:
  - controls both the camera module (in particular for house-keeping functions such as autoexposure and auto-focus), the ISP (and in particular auto-white-balance), and other external components such as the Flash and/or possibly the mechanical shutter and iris.
  - embeds a floating point unit to accelerate floating point operations.
- set of dedicated peripherals (timers, GPIOs, …).
- module to allow communication with the sensor.

The ISP can be seen as

- a common front-end, named the recovery engine (RE) which process data in raw bayer domain and
- 2 parallel back-ends named the color engines (CE) which process data in YUV domain. The rationale behind is to be able to output high resolution frame on the HR pipe concurrently with low-resolution frame on the LR pipe.

The boundary between both is the demosaicing stage which does the transformation from Raw Bayer to YUV.

Each color engine has its own crop/scaler.

The whole pipe is referred to as “pixel pipe”.

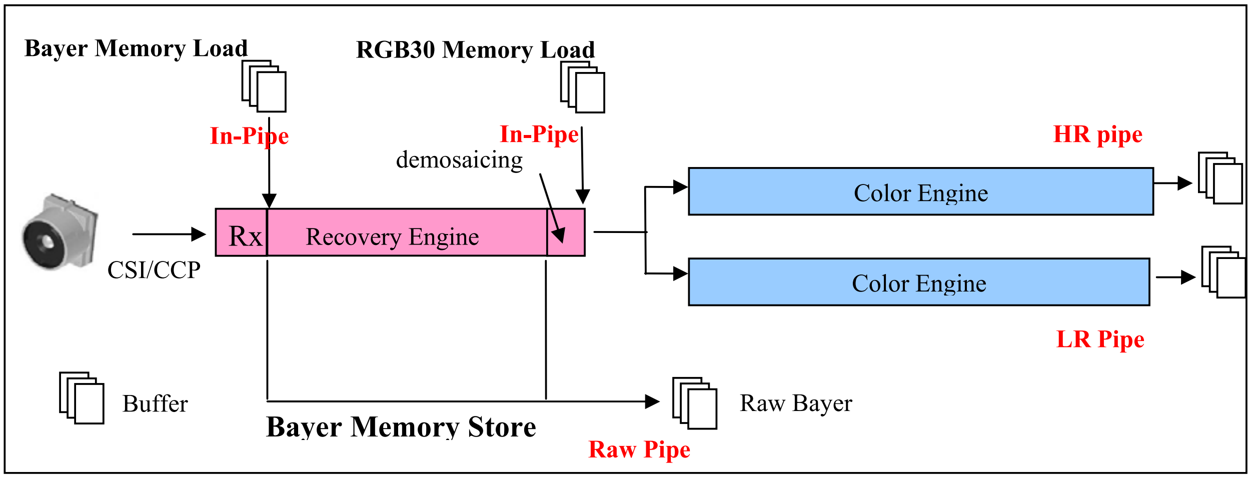


Figure 6. ISP Engine, Front-end and backends, bypass modes

The HW allows other bypass modes (not represented on the above picture) but those are not planned to be used.

The ISP HW is controlled by the MMDSP+ subsystem and a micro controller unit named XP70.

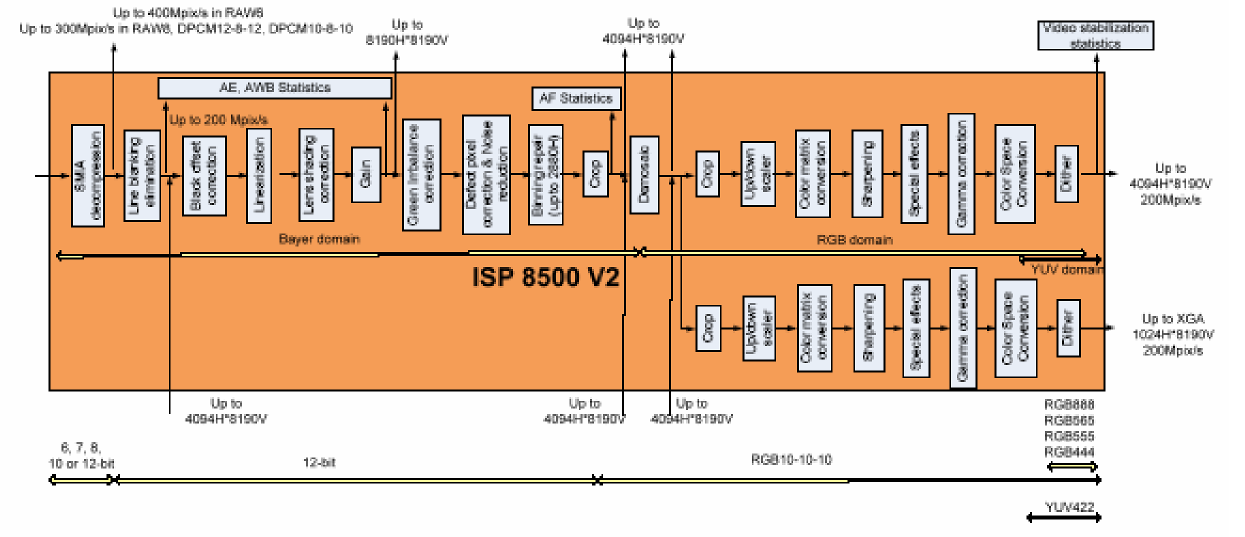


Figure 7. ISP 8500V2

5.3.1 Recovery Engine

The recovery engine is also referred to as front-end (FE) of the pixel pipe. It corresponds to the upstream part of the ISP pipe taking as input Raw Bayer data coming either from the sensor module or from memory and processing it into RGB. Includes the following Raw Bayer domain processings:

- Bayer smooth
- Black-level offsets
- gain and offset application (AWB),
- noise reduction (arctic),
- anti-vignetting
- interpolation from Raw Bayer to RGB (demosaising)
- barrel/pincushion distortion correction.

This upstream part of the HW pipe is shared by the multiple back-end (BE) pipes, also referred to as color engines (low quality, high quality, …), introduced in the next chapter.

5.3.2 Color Engine

The color engine is sometimes referred to as back-end (BE). It corresponds to the downstream part of the ISP pixel pipe, taking as input RGB data (output of the FE demosaicing/barrel distortion stages) and post-processing it in the RGB or YCbCr domain. This mainly includes

- cropping,
- scaling,
- RGB color matrix correction,
- Radial peaking (aka. Sharpening)
- Effects (aka. color tones),
- gamma correction
- overlay capability
- conversion to YCbCr
- dithering.

Out-pipe0 and output-pipe1 can run concurrently.

5.3.3 Bayer Memory Store

Out-Pipe2 dedicates to Raw Bayer output. It is also referred to as raw pipe. It actually corresponds to a bypass output mode of the recovery-engine which does not involve any color engine processings.

The recovery engine can output:

- after the HW interface receiver (Rx),
- before the demosaicing (CFAI).

This mode is also referred to as “Bayer Memory Store”, abbreviated BMS.

5.3.4 Bayer Memory Load and RGB Memory Load

It is also possible to feed the pixel pipe with Raw Bayer (uncompressed raw8/raw12) or RGB30 input buffers. This mode is referred to as “Bayer Memory Load” and “RGB Memory Load”, respectively abbreviated BML and RML.

The pixel pipe can be fed

- after the HW interface receiver (Rx), with Raw Bayer buffers,
- after the demosaicing (CFAI) with RGB30 buffers.

In-Pipe0 is the pipe being used for this bypass entry.

It must be noticed there is no support for compressed raw bayer input buffers. This is due to the fact the input is after the Rx block which performs HW decompression of raw bayer.

5.4 YUV/JPEG Pipeline

YUV/JPEG Pipeline implements a simple receiver for connecting the YUV or JPEG camera module (embedding an ISP), or raw-bayer camera through an external co-processor. It can receive YUV422 8-bit data or JPEG data. Alternatively, it can receive user-specific RAW data from either the parallel or CSI-2 interface.

6 SW Overview

The camera SW sub-system relies on 3 software stacks: the ARM stack, the MMDSP+ stack and the ISP8500 XP70 micro-controller stack. The SW breakdown is pictured below.

This breakdown is mainly showing Android SW stack as a reference.

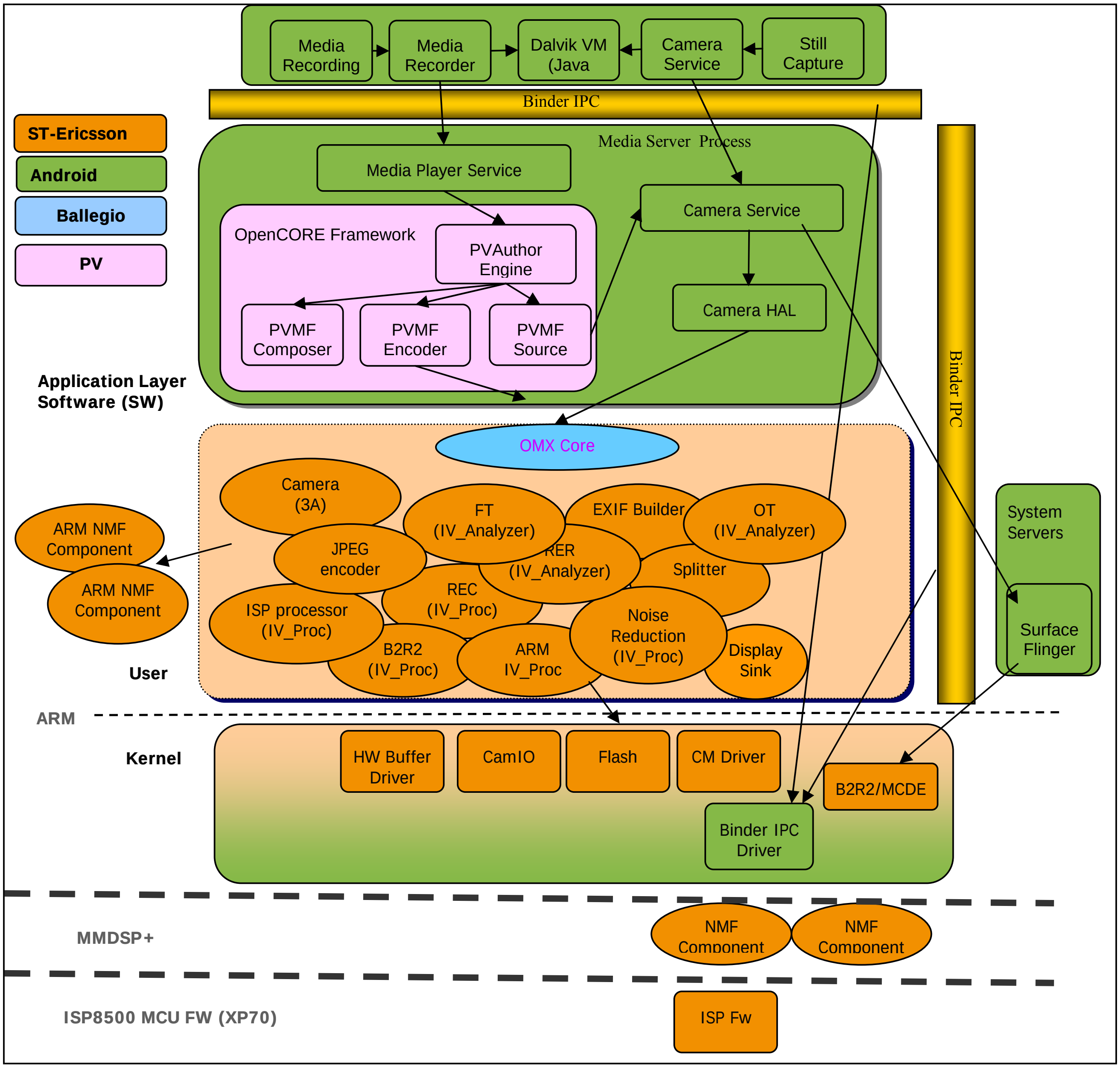


Figure 8. Imaging Sub-System SW Stack

From bottom up,

- the ISP FW finely controls the image reconstruction pipeline stages and also controls the sensor. The ISP FW runs on the xP70 MCU.

- The MMDSP firmware controls the SIA and SVA HW accelerators as a whole. It typically programs the DMA that extract/feed data from/into the SIA8500 HW pipes and the various HW blocks that are involved in the processings (stab, clock manager, …). This SIA MMDSP FW code is structured in functional units named NMF components (Nomadik Multi-Processing Framework). Among other things, NMF provides a framework for multi-processing unit development that alleviates asynchronous inter-CPU communications, marshalling and defines an execution model common to all NMF components (hybrid run-to-completion).

- Finally, the ARM executes the Android stack from highest application level (Camera application, Media Server Process) downto Linux device drivers running in priviledged mode. OMX components run on user side. Their ARM skeleton is called proxy and delegates most of the processings to asynchronous service providers, Linux device drivers, and NMF components, running either on ARM or SIA MMDSP.

NMF and OpenMax frameworks are the corner stones of ST-Ericsson camera adaptation.

ST-Ericsson OMX components are most often based on NMF components (either FW or ARM ones).

6.1 OpenMax IL Adaptation

This chapter details all the OMX components that are used by the camera sub-system.

OMX Component	NMF Dependency	HW Dependency	Description
Camera	SIA-NMF + ARM-NMF	SIA	abstracts one camera instance. Granted with a 3d port as opposed to std OMX component
B2R2 IV processor	No	B_2R_2	abstracts video processing (rotation, scaling, cropping, …). Performs memory to memory processings.
ARM IV processor	ARM-NMF	/	Performs rotation, scaling and format conversion
ISP processor	SIA-NMF	SIA	Performs image reconstruction from memory to memory using ISP Functionalities of SIA
REC IV Controlled processor	ARM-NMF	/	Implements Red Eye Correction on the input frame according to the red eye location provided through its input OPB port
Norcors Noise Reduction iv processor	ARM-NMF	/	Perform noise correction (chroma noise reduction)
Object Tracker Analyzer	ARM-NMF	/	Tracks a rectangle (object definitions) within the captured frames
RED IV analyzer	ARM-NMF	/	Detects the red eye location in an input frame and output those information on an OPB port
FT IV Analyzer	ARM-NMF	/	Detects faces on the input frame and output those information on an OPB port
JPEG Encoder	SVA-NMF	SVA	JPEG encodes an input frame
Splitter	/	/	1-2 splitter that broadcast an input buffer towards 2 output streams

Sequential Splitter	/	/	1-2 splitter that broadcast an input buffer towards 2 output streams. Priorities are defined on its output ports.
EXIF Builder	/	/	Merges the primary EXIF JPEG and its DCF thumbnail
Display Sink	/	B_2R_2 / MCDE	OMX IL sink that implements the connection with the display sub-system. Used for efficient video/still rendering (similar to DSA).

Table 6 OMX Components

7 Overview of OMX components

7.1 Extended Camera

This component is the main Camera OpenMax component which emits frames and statistics according to settings at its ports. It is one input and 3 output ports as shown in the figure below. The port definitions of this component are different from the 2 output port in Khronos definition /11/ .

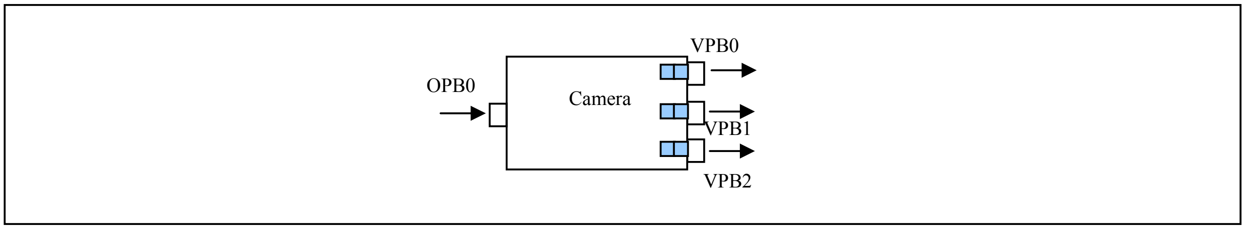


Figure 9. Camera

In a general scenario, the Omx Camera outputs viewfinder frames at VPB0, raw bayer frames at VPB1 i.e. still captures, and video capture frames at VPB2. The input port OPB0 is provided for synchronization with an audio clock but this could be disabled.

It abstracts one camera at one time. There is one implementation of this component per camera, one for the primary camera and another one for the secondary camera. There is a single component role associated with this component.

This component is developed by ST-Ericsson.

The camera component abstracts the sensor, its related actuators as well as the flash device.

The component enables to configure/control the camera and grab buffers from its 3 output data ports that fits well with the 8500 dual pipe architecture. 3rd port was added to support still image capture during video record and for analyzers during a still capture operation.

Depending on the operating mode, the camera component abstracts a part or the entirety of the ISP8500 pixel pipe, as described in the following chapters.

The camera components also relies on ST-Ericsson 3A component for 3A compution (AF, AEC, AWB).

The component has 2 modes of execution – Video and Still Mode. This is explained in the following sub-sections.

7.1.1 Video and Viewfinder Mode

In this mode, the camera component abstracts the whole dual pixel pipe and the 3A algorithms. In this mode, the VPB0 and VPB2 ports are actively contributing with VPB0 being used for preview and VPB2 being used for the video captured frames.

Features that can be controlled through the OMX Camera in this mode are:

- Drive Mode Config & Control,

- Flash & actuator settings,
- Recovery Engine Settings,
- Digital Zoom settings,
- Colour Engine Settings per output port (VPB0, VPB2).

Some of these features are applicable to a single port and some are applicable to all ports in a common way.

Computation latency is max 200 Mpel/s.

The camera component also outputs "video pack" extradata along with the pixel data at VPB0 and VPB2 ports. Those are defined in 25.

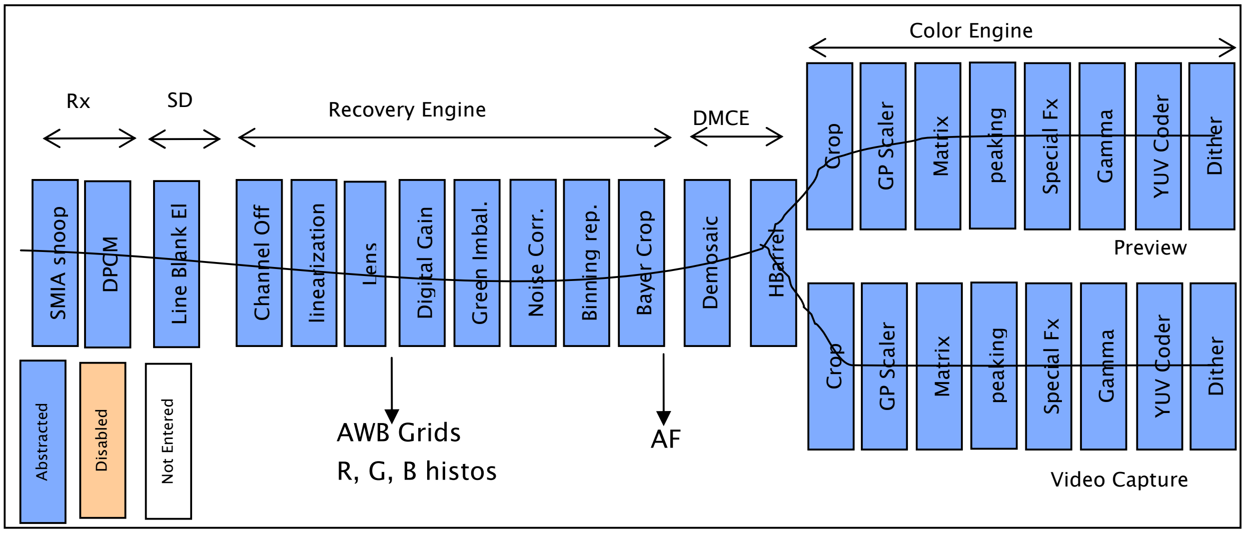


Figure 10. Pixel Pipe Stages abstracted by the camera OMX component in VF and video mode

7.1.2 Still Capture Mode

In the still capture mode, the camera OMX component still controls the sensor, the flash and all the actuators. But it outputs only Raw bayer frames at VPB1 port that are processed partially and hence named 'ideal' raw frames. These frames are stored in memory using the BMS HW. And later these frames are loaded back into the ISP pipe inside the ISP Processor using the BML HW. The reason for introducing such a split between the Omx Camera and the ISP Processor is that this allows flexibility to introduce new algorithms in the Raw bayer domain. For eg., incase you wish to pre-process the raw bayer data with some de-noising algorithms before processing it in the YUV domain, you can use this component split to achieve this.

Due to the above split the Omx Camera abstracts only a subset of the stages of the pixel pipe:

- The CSI2/CCP receiver
  - Preprocess the sensor raw data into the IDP 12-bit parallel format that is compatible with the pixel pipeline datapath
  - For SMIA sensor, extracts intelligent status lines used to automatically configures the pixel pipe backend
  - PCM/DPCM Bayer decompression
- The Sensor Data Pipe (SD Pipe) featuring optional line blanking elimination
- Channel offset correction
- Linearization

- Lens shading correction (LSC)
- Defect Pixel Correction (not the rest of the noise reduction)
- Bayer Memory Store before demosaicing

The grabbed data is output just before demosaicing, meaning the color engines are not entered at all and the output data has a Raw Bayer data format. There is a special mode where raw bayer is output without any treatment, ie. as it is output after the Rx.

In still capture scenario the Omx Camera cannot be used in a standalone mode. It has to be used in conjunction with the ISP Proc for processing raw bayer frames into the YUV domain.

The ISP IV Proc is connected to the VPB1 port of the OMX Camera.

During the still capture use case, viewfinder might be active with preview frames being sent at VPB0 port of the OMX camera.

The Camera OMX component abstracts AWB gain computation on the current grabbed frame (as opposed to the AWB computation in VF and video capture which are based on the n-2 frame statistics). Although the camera component computes the WB gain, it does not apply it.

Lens shading correction might be disabled for production test use cases. Then it is enabled inside the downstream ISP IV Proc.

To allow the remaining reconstruction inside the ISP Iv proc, the camera fills “still pack extradata” along with the output raw bayer data. Those extradata are described in 25.

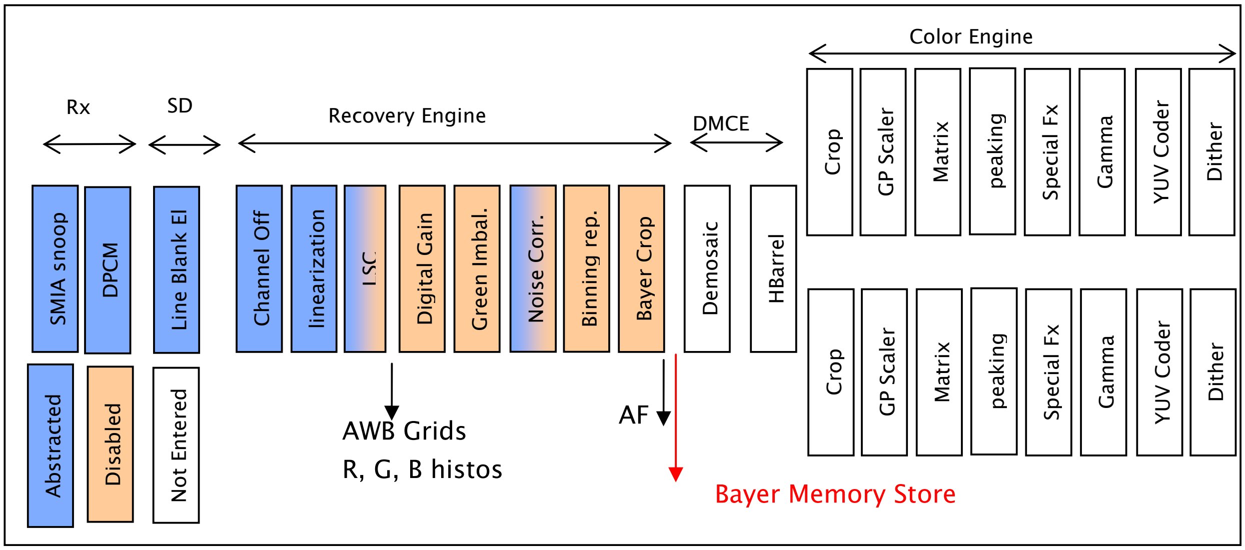


Figure 11. Pixel Pipe Stages abstracted by the camera OMX component in Still capture mode

In still mode, the camera only implements

- Drive Mode Config & Control
- Recovery Engine Settings

The rest of the camera OMX IL API are supported but some settings cannot be applied on the camera side (typically DZ, since the colour engine where the scaler stands is not entered) since not abstracted by this component for the still use case. In that case the settings are passed through extradata to the downstream ISP processors. This latter completes the image reconstruction (raw to yuv conversion), from memory to memory.

The fact recovery engine is entered limits the throughput to 200 Mpel/s. In case raw bayer would be output after Rx, 300 Mpel/s is reachable. But this mode is not supported today.

The camera relies on the ST-Ericsson 3A API, implemented on ARM side to compute automatic exposure control, automatic white balance control and auto-focus.

7.2 Extended IV processors

These IV processors are in general one input and two output port OMX components. These components are generally for memory to memory processings and provide the result of processings at its output ports potentially with different output formats and resolutions.

This definition of IV Processors is not in Khronos 1.1.2 /11/ . Hence it is called Extended IV processors.

7.2.1 ISP processor

This component is one of the IV Processors which is based on the SIA HW or ISP8500. It implements image memory to memory processing. The purpose is to enter the ISP pixel pipe in raw bayer, use raw and RGB processings offered by the ISP recovery engine and colour engines respectively and output data on one or both output data ports, in YUV formats potentially with different formats and resolutions.

This component is developed by ST-Ericsson.

As shown below, the ISP Proc supports one input and two output ports. VPB0 is for taking input frames. Frames are outputted at VPB1 and VPB2, with VPB1 emitting lower resolution frames if required.

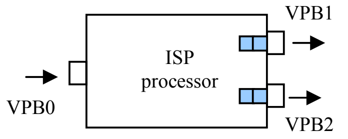


Figure 12. ISP IV Processor

As pictured below, the ISP abstracts the stages that were not abstracted by the fellow upstream “still” camera OMX component. Both components are complementary.

The ISP IV processor abstracts

- The Bayer Memory Load
- The Recovery Engine (RE)
- The Concurrent LR/HR engines and associated PRY’s

A pass within the ISP IV Proc is limited to 200 MPel/s throughput.

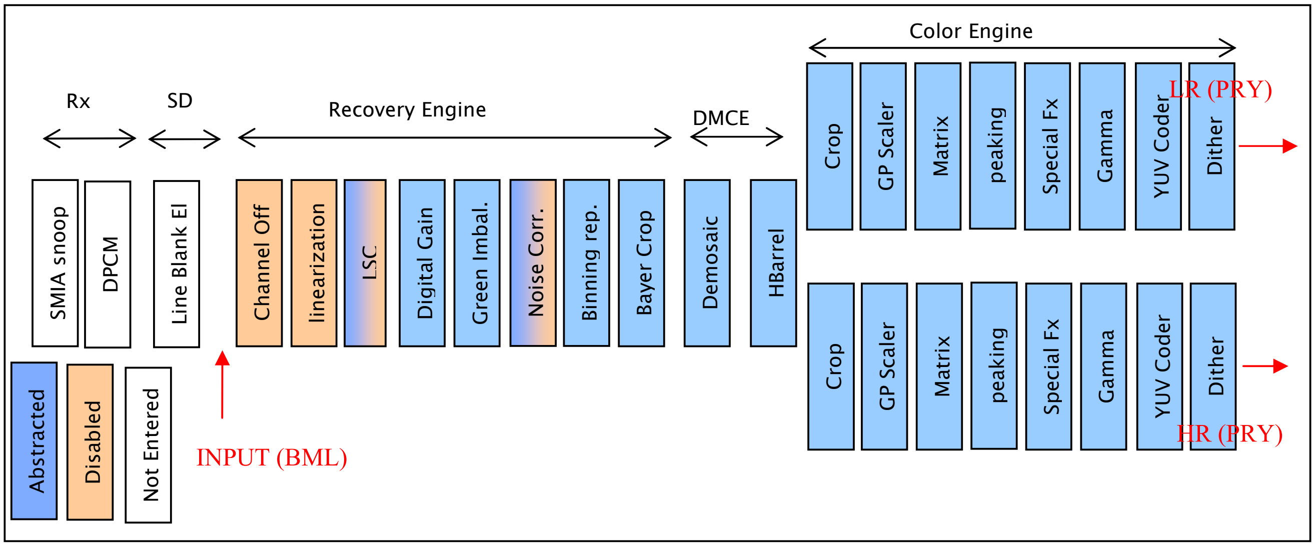


Figure 13. Concurrent output on LR and HR

As explained in the OMX Camera section, raw bayer frames that were stored in the memory by the OMX Camera and loaded from memory inside the ISP IV Proc back into the pixel pipe. These raw bayer frames are then fully reconstructed through the remaining Reconstruction engine and the color engines.

The 2 ports of the ISP IV Proc are mapped to the 2 pipes or 2 CE's of the ISP HW.

Settings applied in the ISP proc are provided through the 'still pack' extradata coming encapsulated inside the still buffers. The settings which are not applied at the OMX Camera are applied at the ISP Proc at its output ports i.e. saturation, effects, contrast, brightness, sharpness, YUV range, etc.

In case lens shading correction is disabled on camera side, it might be enabled on ISP processor side.

The ISP processor does not perform any 3A. However it is in charge of applying the WB gains computed by the camera source and provided in the 'still pack' extradata.

Please refer to 25 for complete description of this extradata exchange and how it is used by the components.

7.3 Standard IV Processors

7.3.1 B_2R_2 IV Processor

This is a standard OMX component that abstracts Image/Video memory to memory processings using the B2R2 HW. It abstracts all processings that the B2R2 HW can offer at OMX level. However it does not output to any display device HW interface which is done by the display sink OMX component described later in this main section.

It is a standard IV processor with one input and one output port.

This component is used at several places in Imaging dataflows where it is mandatory to do some processings on HW and B2R2 forms a good candidate for that. Such features could be resizing, rotation, format conversions, etc.

This component is developed by ST-Ericsson.

Input \ Output	OMX_COLOR_Format32bitARGB888
YUV422 itld	• Pixel accurate cropping (for stabilization) • Rotation (+90°, +180°, +270°) • Scaling • Mirroring
YUV420 raster (I420, OMX_COLOR_FormatYUV420PackedPlanar)	• Pixel accurate cropping (for stabilization) • Rotation (+90°, +180°, +270°) • Scaling • Mirroring

Table 7 B2R2 iv processor

The documentation of this OMX component is out of the scope of this document. Details can be found in /19/ .

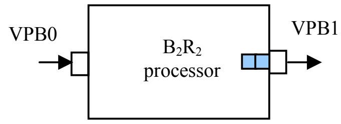


Figure 14. B_2R_2 IV Processor

The component does not implement buffer sharing.

7.3.2 ARM IV Processor

This component is a standard IV processor that is implemented on ARM. It does not rely on any HW or DSP.
Hence it has single input and single output port.

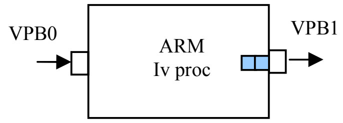


Figure 15. ARM IV Processor

It implements the following transformations and formats.

Input \ Output	YUV420 I420	YUV422 ITLD	NV21 YUV 420 SP OMX_COLOR_ FormatYUV420 SemiPlanarNV21	YUV420 MB
YUV420 (I420) OMX_COLOR_FormatYUV420PackedPlanar	*Not required*	Rotation Conversion scaling	*Not required*	Rotation Conversion scaling
YUV422 ITLD OMX_COLOR_FormatCbYCrY	Rotation conversion	*Not required*	Rotation Conversion scaling	*Not required*
YUV420 MB	conversion	*Not required*	*Not required*	*Not required*

Table 8 Supported Processings

It does not implement any buffer sharing.

7.3.3 Norcos IV Processor

This is a standard IV processor that is fully ARM based. It implements Norcos algorithm which is chroma noise reduction in the YUV domain. It is implemented upon YUV420 packed planar format.

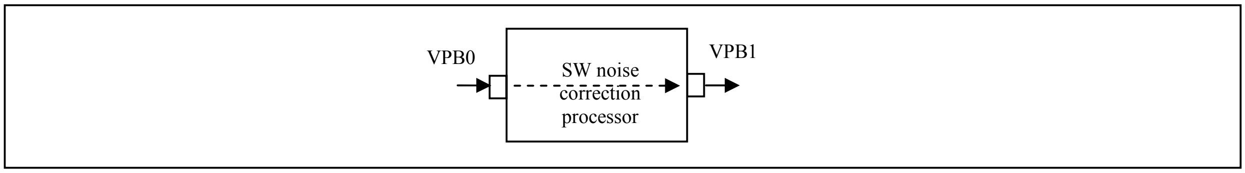


Figure 16. SW Noise Correction Processor

The processing is done in-place, ie. the component implements buffer sharing.

7.4 IV Analyzers

They are non standard OMX IL components that take as input an uncompressed image/video frame and output metadata information computed upon the content of the input frame. Those metadata can be output through an

- OPB output port (for red eye detection only)
- through OMX functional event (case for others)

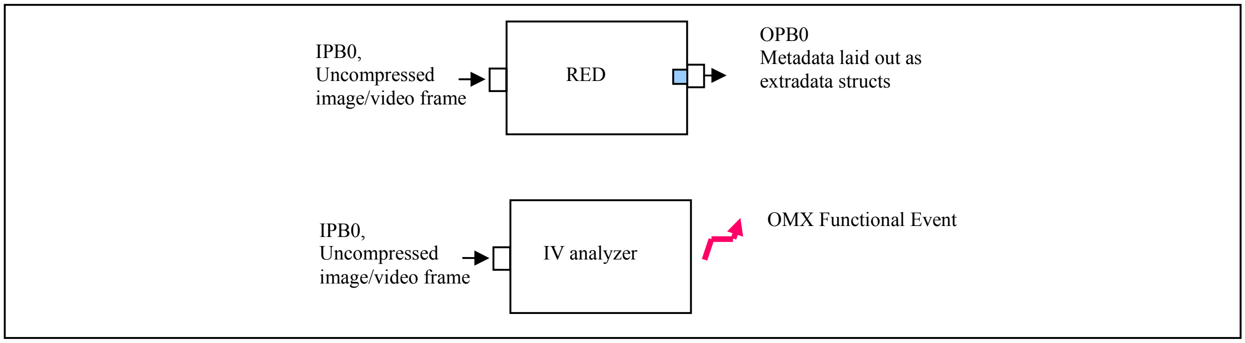


Figure 17. IV Analyzer

There are separate analyzers for:

- Red Eye Detection (RED)
- Face detection (FD)
- Object tracking (OT)

RED is used on still stream. FD and OT are used during still VF analysis. Please refer to still dataflow in 10.2.

7.4.1 Red Eye Reduction IV Analyzer

Red Eye Reduction analyzer is the one which has an output port and provides the anaysis information as metadata at its OPB0 buffer.

It is used in the still dataflows. Please refer to still dataflow in 10.2 for more details.

This analyzer detects and computes the red eye locations, if any. It is a full ARM based component.

7.4.2 Face Tracker IV Analyzer

The analyzer locates the faces, if any, in the input uncompressed frame, and assign them a priority according to a specific heuristic. It uses the OpenCV library. The face data ROI is provided as metadata.

7.4.3 Object Tracker IV Analyzer

This analyzer tracks a rectangle whose coordinates are provided as a configuration. The region of tracked object is provided as metadata. It relies on OpenCV library.

7.5 Extended IV Controlled Processor

This is a non-standard IV Processor which performs memory to memory processings of input image or video frames based on a set of control commands provided at an input port. This is not defined in the Khronos 1.1.2 /11/ .

Hence this component has two input ports and one output port. One input port i.e. OPB0 receives control data for controlling and synchronizing the processings of the video or image data provided on the second input port i.e. VPB0.

The output frames after processing are outputted at VPB1 port of the component.

There could be several roles defined for these components out of which one is described below.

7.5.1 Red Eye Correction IV Controlled Processor

REC OMX component is an IV controlled processor with a second input OPB port. This OPB port is fed with red eye location metadata computed by a separate Red Eye Detection analyzer (see 7.4.1).

This block applies desaturation on the red-eye areas. The processing is done in-place, ie. REC processor implements buffer sharing.

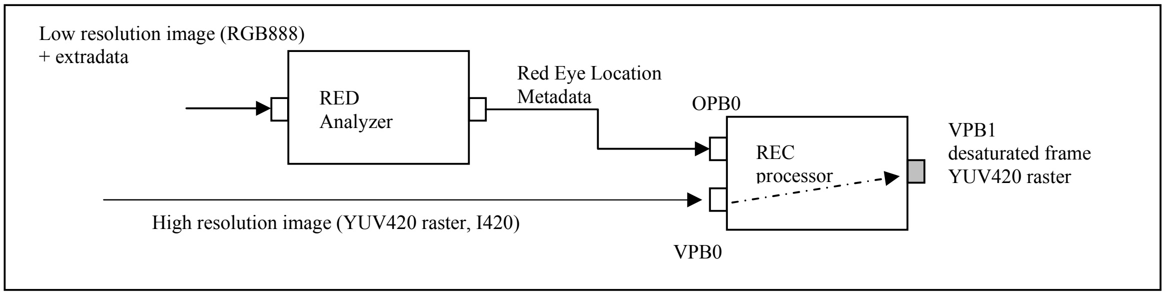


Figure 18. Red Eye Detection Analyzer and Red Eye Correction Processor

7.6 JPEG Encoder

This component is a standard OMX component, developped by ST-Ericsson. There are 2 flavours of this component:

- an HW accelerated one, which relies on SVA HW (supporting 420 MB input)
- an ARM based one, which relies on an ARM implementation (supporting YUV422 itld input)

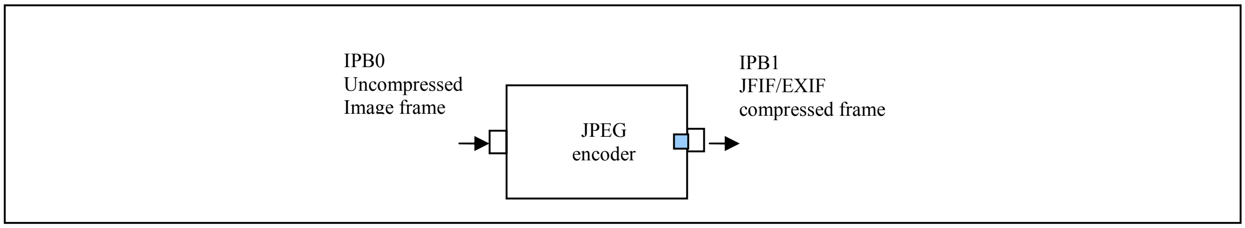


Figure 19. Jpeg Encoder

The ARM JPEG encoder is required since the SVA HW is not supporting YUV422 itld as input. This latter is required for DCF thumbnail encoding (YUV422 chroma sampling factor, QQVGA or QVGA). The EXIF JPEG thumbnail does not contain any APPn, nor COM, nor restart markers.

Both encoders are able to produce JFIF or EXIF compressed buffers. They supports baseline JPEG encoding, (YCbCr 4:2:0 input, one interleaved scan, no adaptative quantization).

Thumbnail insertion is not managed by the encoder itself but requires the instantiation of a downstream EXIF builder that merges the primary JPEG image with its DCF thumbnail. Please refer to 7.9

.

7.7 Video Splitter

This component is used to broadcast one video/still source towards two or more video/still outputs. Hence it acts like a broadcaster component. It sends input buffer to all the output ports at one time.

Its ports are non allocator ports. The component implements “Read-Only” buffer sharing between its input and output ports. Output ports are buffer suppliers.

The component does not implement any notion of priority between the different output ports, ie. the output ports are served immediately as opposed to in the next component.

The downstream components cannot perform in-place processings. As such the output tunnels must be read-only.

7.8 Sequential Video Splitter

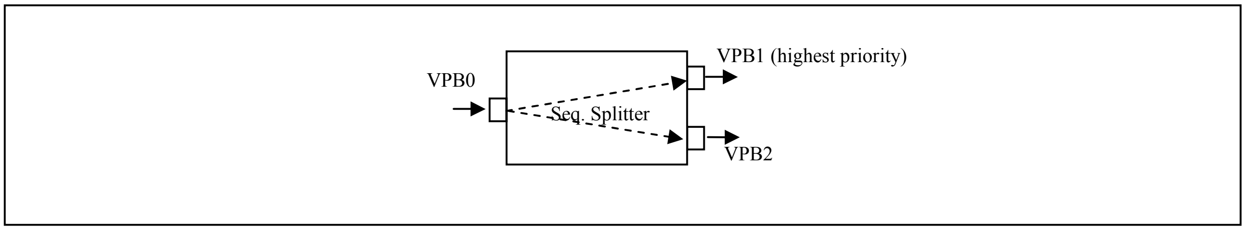


Figure 20. Sequential Splitter

Each input frame is propagated first to VPB1 and then, when the buffer is pushed back to VPB1 by the downstream component, it is served to VPB2. Hence the input buffer is propagated to all output ports one after another in increasing port number.

Output tunnels are not requested to be read-only since there is a guarantee that buffers have been used by downstream components before they are passed to other output ports.

7.9 EXIF Builder

This component is a non standard component developped by ST-Ericsson.

This Omx component is used to build EXIF images by merging the DCF thumbnail and the JPEG image into one stream. Hence this is also called an Image Mixer as it merges several input streams into a single stream.

It accepts thumbnail at IPB0 input port, JPEG at IPB1 input port and merges them into an EXIF image at IPB2 output port.

The EXIF Builder is an ARM based component.

7.10 Display Sink

This is a non standard sink component which displays the frames on screen.

This component enables efficient video/still “direct” rendering (similar to former “direct screen access” functionality). As such it is a gateway to the HW display sub-system.

Buffer supplier is output port of tunneling component. In our dataflows this would generally be the Omx Camera component.

It uses the Multimedia HW Buffer API to allocate physically contiguous buffers. It relies on the B2R2 user library for cropping, rotation, mirroring, scaling etc and performing HW accelerated display.

As such the implementation uses ARM code, no DSP FW, indirectly uses B_2R_2 HW and MCDE HW.

This component is developped by STE.

This component will mainly be used in OSI dataflow. In OS specific platforms, it might vary with the targeted platform. In Android, for example, Surface Flinger is used as the gateway to the display sub-system. Hence, the Display Sink is not used in any of the dataflow. Input buffers supposed to go to the Display Sink in OSI dataflows are sent to the Surface Flinger instead.

This component is derived from Bellagio FrameBufferDev component.
It provides capability for Input Cropping, Rotation, Mirror, Scale, Output position setting.

8 Linux Drivers

The OMX components listed in the previous chapter directly use the following ST-Ericsson Linux device drivers (more precisely their user-side wrappers).

Drivers	Description
Component Manager	enables user-side to interact with NMF components. Also provides some services linked to memory allocation (especially eSRAM).
CAMIO	Enables to initialize the HW interfaces (GPIO, I2C) , request/release power physical and logical resources through Power Resource Management (PRM) API.
Flash	Enables to configure & control the Flash, enables to listen to asynchronous flash events
Multimedia HW Buffer (aka Alloc API)	enables to allocate physically contiguous buffers and set their attributes
B_2R_2 Driver	Enables to configure and control the B2R2 HW IC

Table 9 Linux Drivers

8.1.1 Component Manager

The Component Manager API (further abbreviated CM API) brings a C user-side interface to access the underlying NMF framework services and especially enables to load NMF components from the file system, bind them together or from/to the host, control their state, mediate their interface calls.

It can somehow be seen as a gateway to all NMF components, running either on ARM user side, SIA or SVA domains.

It stands at the top of the so-called NMF stack.

The CM API is used by the OMX components that are implemented upon NMF components. OMX components that do not have dependency on NMF do not need to use it.

On top of its first functionality, ie. ARM-NMF world gateway, CM offers services related to eSRAM allocation.

The C user side interface relies on linux kernel driver and calls its ioctls.

The CM mediator is not used by the OMX components. It aims at initializating the CM stack and dynamically load the NMF components from the file system.

The CM engine is the OS agnostic part of NMF and brings the core of the technology.

8.1.2 CAMIO

The CAMIO API is a user-side API that exposes a collection of HW based services that are needed to execute camera use cases. The API is defined and developed by ST-Ericsson.

It especially aims at controlling low-level resources that are involved in the camera sub-system use cases and that are not controlled by MMDSP firmware:

- GPIOs (programming of the alternate functions, GPIO pin characteristic programming: pin direction, level, pull-up/down enable/disable, interrupt edge, sleep mode, …)
- Physical power resources
- Logical power resources
- Access to SIA registers and memories (load of the XP70 ISP FW binary into the XP70 TCMs, XP70 MMIO remapping on DDR, initialization of SIA block memories (Gridiron, Gamma LUT)

This CAMIO API only is used by the camera OMX Components.
It initializes the board, switches on external clock, powers on the sensor when requested, and sets the GPIOs for configuring the camera HW interfaces.

The CAMIO has two parts

- CAMIO Proxy
- CAMIO DD

The CAMIO Proxy lies in the user space and acts as a proxy to the kernel CAMIO driver. It will open and close the CAMIO device file which will lie in the /dev directory (/dev/camio_camera). And would send ioctl commands to the device driver. Every ioctl goes with a mgic number and an ordinal number. The magic number is device specific hence fixed for CAMIO device driver. Ordinal number would change with every command type. This will be accompanied with a r/w/r-w parameter.

The CAMIO device driver which resides on the kernel side provides implementation for all mandatory driver APIs like ‘init’, ‘release’, ‘open’, ‘close’ and all the ‘ioctl’ commands.

CAMIO Proxy and CAMIO DD might also exchange parameters or status information but with proper mapping to the user/kernel side.

8.1.3 Flash

There is a Flash User API supported (C++) at platform level which abstracts functionalities of the Flash HW device.
This API is OS agnostic C++ interface and will be used by the Omx Camera for controlling and configuring the flash HW device. It might be wrapped into an ARM-NMF component.

The Flash User API implementation is a Flash proxy which lies user side but relies on a kernel side flash device driver to actually control the flash HW device. The implementation of this proxy would allows to open and close this flash device and maps all flash user interface functions to ioctls of the flash device driver. Internally the Flash device driver programs the flash HW device using the platform GPIO and I2C support.

Since the Flash User API is separated from the Flash device driver, it allows easy porting to new flash devices without changing anything in the upper layers for eg. the Omx Camera.

The flash driver abstracts flash modes: still LED (with / without external strobe), video LED (with / without external strobe), still xenon (with / without external strobe), AF assistant, privacy indicators.

As can be seen above the driver provides the capabilities to either strobe the flash through the flash driver or through some external mechanism (eg a sensor directly controlling the flash strobe – this is usually done on our platform in still use case).

The Flash User API offers services related to flash state management, configuration and strobe in case the strobing is not delegated to another component.

The Flash User API provides services to return flash device information to its clients.

It also provides support to return status information or any erroneous states and support for self tests.

The driver hides the way the IC are physically programmed (through I2C or 2-bit logic).
It also hides the number of ICs required for the above use cases.
The flash driver has no dependency on the sensor module adaptation.

For the complete user API, please refer to /24/ .

8.1.4 Multimedia HW Buffer

There is a need to allocate contiguous buffers by most of the Omx components. For this a user side Multimedia HW Buffer API has been devised, which is used by all multimedia Omx components to allocate buffers. This Multimedia HW Buffer user API allows creation and manipulation of such buffers.

This low-level user side API is OS agnostic C++ interface and allows easy integration into OSI Omx components. The user side API proxy implementation relies on a kernel device driver which actually performs the memory allocations.

The API enables to allocate/free a pool of buffers and map them onto a chunk. Such a HW memory chunk is made for every port which is a buffer supplier. All buffers are physically contiguous and have the same set of properties in a given pool such as size, caching attributes, access permissions, alignment. The handle of this chunk is mapped in the client process and hence is thread/client specific (reference counting is maintained). Full process r/w permissions are allowed by default.
A handle of the omx component is also maintained inside the pool to keep client information. These chunks are accessible both kernel and user side.

A buffer represents only the payload part including any extradata.

This kernel device driver is currently used to allocate contiguous buffers accessed by the SIA/SVA DMAs but not seen through the DSP D$ cache. On the opposite, CM LDD offers services to allocate buffers seen through the DSP D$ cache.

8.1.5 B2R2

There is a post processing API available which is used inside the B2R2 IV Proc Omx component. This API abstracts the functionalities provided by the B2R2 HW.

B²R² HW will be used for several use cases like

- Bitmap manipulations (Blitting/Blending).
- Video/image post-processing
- Composition.

The B2R2 SW package consists of a user side library and a kernel device driver. The user side library acts as a proxy to the kernel driver which actually configures the B2R2 HW. This is shown below.

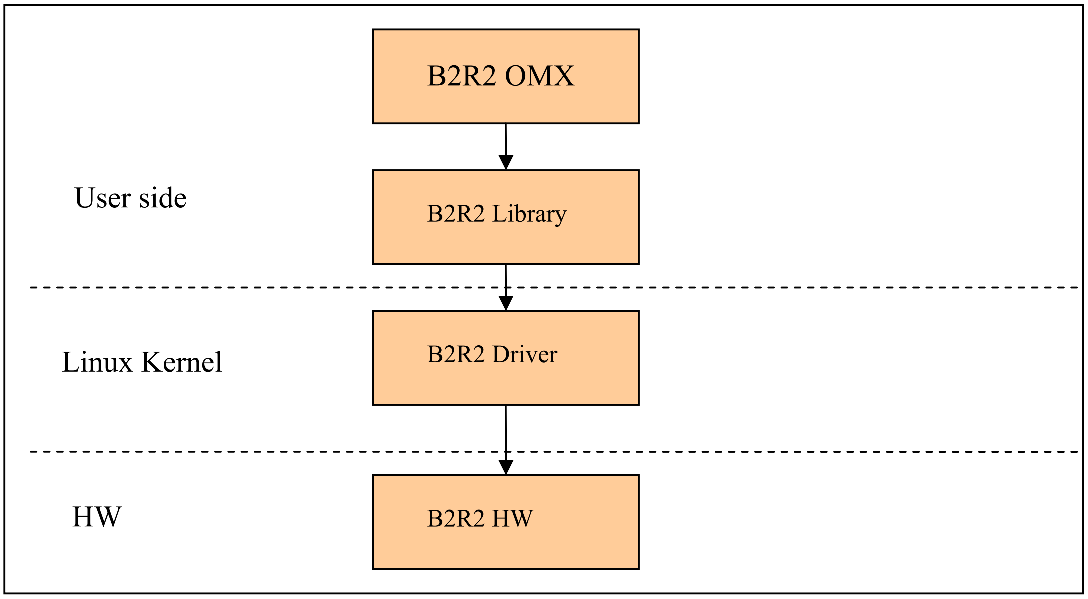


Figure 21.Linux B2R2 SW Stack

The B2R2 library provides mainly following set of operations

- Instance initialization and exit
- Buffer allocations and deallocations
- Post-Processor operations
- Graphic operations

The B²R² Driver is the only software component in charge of managing the B²R² hardware. The B²R² driver is in charge of receiving/sending commands, queueing/dequeuing commands, controlling the hardware (power, interrupts), managing queues (composition, application).

It is capable of handling multiple client requests at one time and maintains a queue of all the requests. No priorities are defined for the clients at SW level. However, depending on the type of HW operation, it is the HW which defines priorities for various operations incase several tasks are waiting to be scheduled at the same time. Hence priorities are defined for clients depending upon the operations being performed rather than user-programmable priorities. In all six different priorities are defined (0 – 5). For requests having same priorities at HW level, requests are completed in FIFO manner.

B2R2library or driver supports the following set of features:

- Fill colour
- Alpha Blending
- Colour Keying
- YUV planar to RGB(Format conversions)
- rotation
- resize
- Copy
- Scaling(up scaling and down scaling in 6.10 format)
- multiple operations like (rotation,resize,colourkeying and blending)
- rectangular clipping(both inside and out side)
- rectangular clipping with colour fill(inside and outside)
- Supports multiple Queues
- Supports B2R2 trigger on VSync.
- ROP operation (AND,OR..etc)
- Clut related operations
- Plane Mask
- VC1 “Range mapping / Range reduction” compensation algorithm.

9 Use Cases

The number of use cases possible and their various combinations is very extensive in imaging. However, all the use cases fall into two main categories – Video and Still Image.

The Video use case hierarchy can be seen as follows:

- The standalone video preview phase
- The concurrent preview + video capture
- The concurrent preview + video capture + still capture

Variants of the above video use cases are:

- digitally stabilized VF and video capture
- CAF during video record
- Rotation of VF and rotation before video encode

The Still Image use case hierarchy can be seen as follows:

- The still preview phase (with or without pre-capture enabled)
  - With or without CAF
  - With analysis (histograms, face tracking, object tracking)
- The half-press phase
  - One-shot AF
  - One-shot AF with pre-capture
  - Completing CAF
- The still capture phase
  - Single capture
  - Finite burst capture
  - Infinite burst capture (also referred to as serial capture)
  - With and without pre-capture

Variants of the above use cases are:

- Still image capture after pre-capture
- Digital image stabilization
- EV bracketing
- Single Still Capture with electronic shutter/mechanical shutter, no flash/Led flash, xenon flash
- Burst Still Capture with electronic shutter/mechanical shutter, no flash/Led flash, xenon flash

The list above is not comprehensive since the combinatorics is far more complex. It is just indicative.

9.1 Preview

This chapter brings a short introduction on viewfinder. More details on the very implementation can be found in 11.2.1.

Viewfinder is also referred to as preview. There are two types of preview depending on type of rendering of preview frames. These are bitmap viewfinder and DSA viewfinder. In bitmap viewfinder viewfinder buffers are sent to the application as Omx buffers. In DSA viewfinder the preview buffers are directly rendered onto the display either through the display sub-system or through the display sink omx component.

Overlays on the VF is not supported at Omx level and is supposed to be managed by the IL client or the application framework, including icons, face rectangles, ….

The supported VF resolutions are listed in Table 15.
nHD is the most probable panel resolution in current timeframe. Associated "+" resolutions also are supported, with adaptative overscan. They are given in 26.

Due to HW limitation (LR pipe), the maximum resolution supported for preview is XGA.

Still preview and video preview are considered separately. They are two distinct operative modes which have different sensor settings, zoom capabilities, AF capabilities, ISP pipe settings, framerate control ….

Constant framerate generally is used in video preview. In still preview, adaptative framerate generally is used except in night and sport exposure mode. However the OMX IL client is allowed to overwrite this policy.

The end-user can vary the zoom (OZ and DZ) during the preview.

9.1.1 Still Preview

Still preview consists in the phase that preceeds and follows (if post-exposure mode is not set) still image capture. In other words, generally, still preview is separated from the very still capture phase. This is due to the fact the sensor settings for still preview generally are not the same as those used for the still capture. Full resolution is used for still capture and pre-scaling configuration is used for still preview.

However, for some sensors allow to use the same settings for still capture and preview phase as they can sustain requisite frame rate at full resolution also.

If post-exposure mode is set at application level, the snap image is displayed on the panel/TV just after the shot to enable the end-user to see a small resolution image of what is going to be JPEG encoded. In that case the VF is not resumed automatically after the still image capture. The end-user must press a button to resume VF.

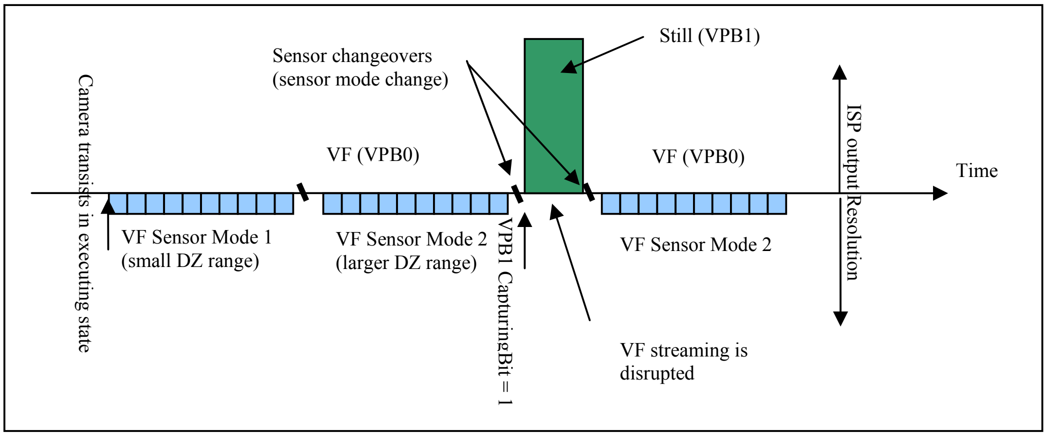


Figure 22. Preview vs Single Still Image Capture where different sensor modes are needed for viewfinder and still capture because of sensor constraints

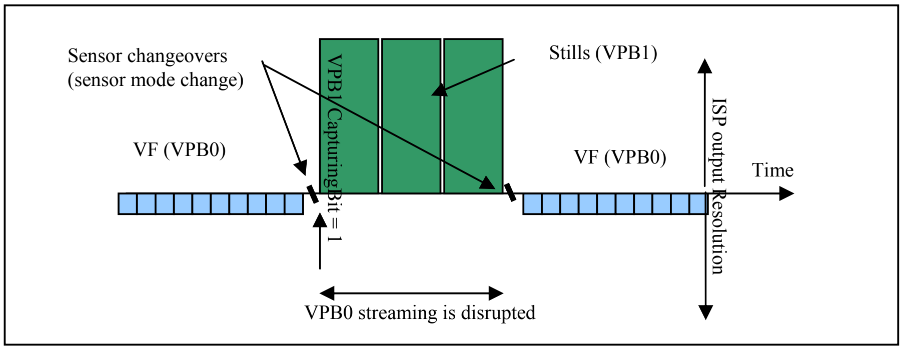


Figure 23. Preview vs Burst Still Image Capture where different sensor modes are needed for viewfinder and still capture because of sensor constraints

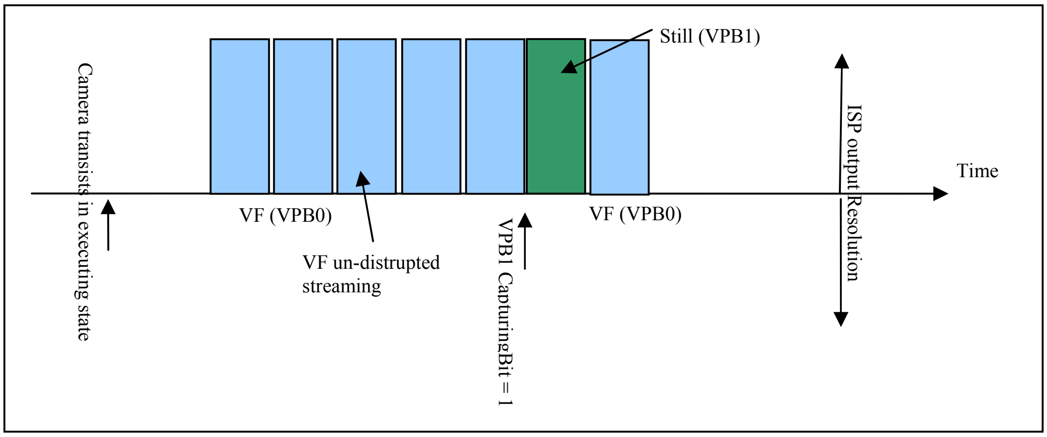


Figure 24. Preview vs Single Still Image Capture where different sensor modes are not needed for viewfinder and still capture because of no sensor constraints

During still preview, the user defines the DZ ROI whose latest configuration will be used for the very still capture. DZ is fully implemented by the camera, on-the-fly, along with the image reconstruction. This requires a minimum line blanking. The camera is likely to change the sensor mode during the still preview in order to choose a sensor mode that accommodates at best the current digital zoom and its minimal line length requirement. See 11.4 .

Along with VF frames, scene analysis is likely to take place on VPB2 stream:

- Face location detection and tracking
- Object tracking

They can help in assisting metering algorithms, meaning their result is likely to be fed back to the Omx camera.

Continuous AF can be active in still preview.

9.1.2 Video Preview

Video preview sensor settings generally are compatible with video capture settings although the output resolutions can be quite different (typically nHD for preview and 720p/1080p for video capture). The needs for video capture generally are anticipated when choosing the sensor mode for preview (of course if they are known). However there might be cases where a sensor changeover is required when moving to video record mode.

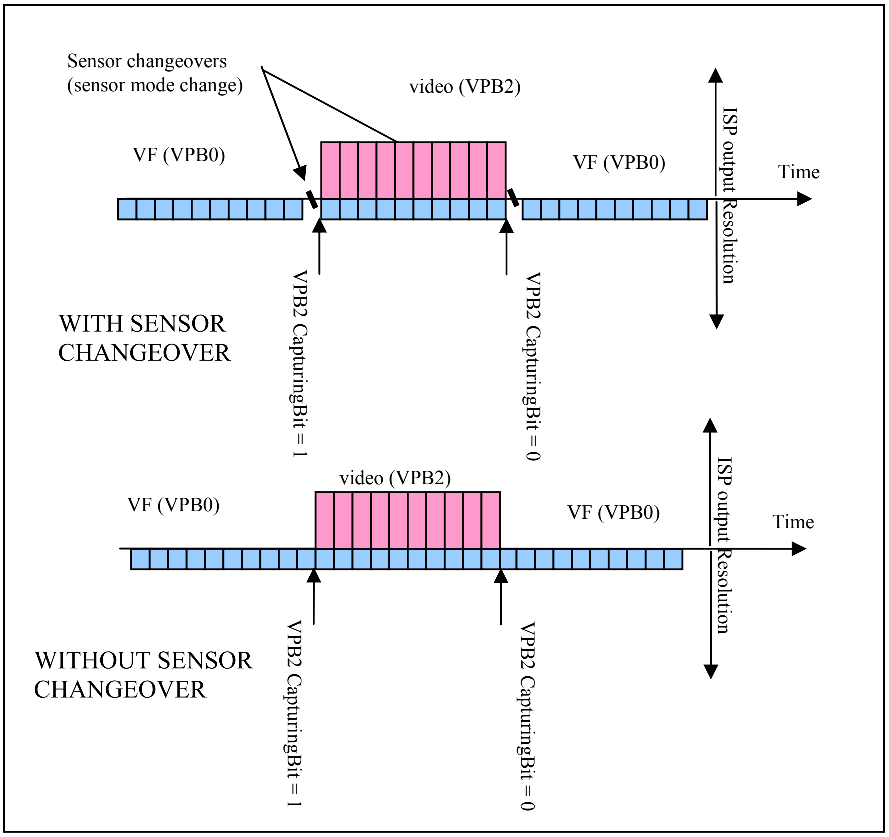


Figure 25. Video preview vs. video record

Identically to still preview, there can be sensor mode changes during the video preview and record to support a continuous digital zoom range. It is reminded the DZ is also allowed to be changed during the video record.

Video stab crop vector computation can be active during the video capture.

Continuous AF can be active in video preview.

No analysis is planned in video preview.

9.1.3 Sensor Settings

Sensor pre-scaling (analog binning, downsampling) or sensor input cropping generally are needed in order to sustain a high framerate (typically 30 fps). Indeed, both techniques enable to reduce the amount of active data that are readout by the sensor thus enabling to speed up the framerate.

In case the limitation stands at the ISP max throughput (200 or 300 Mhz), sensor digital scaling is used to decrease the output pixel clock.

9.2 Half-Press

The half-press period is linked to focus modality. It can be seen as a sub use case of preview. When CAF was off during the preview, it starts one-shot AF search.
The half-press period starts when the end-user presses the shot button half-way during preview and ends when:

- One-shot focus search completes. At that point, AF auto-locks.
- the end-user releases or fully presses the button .

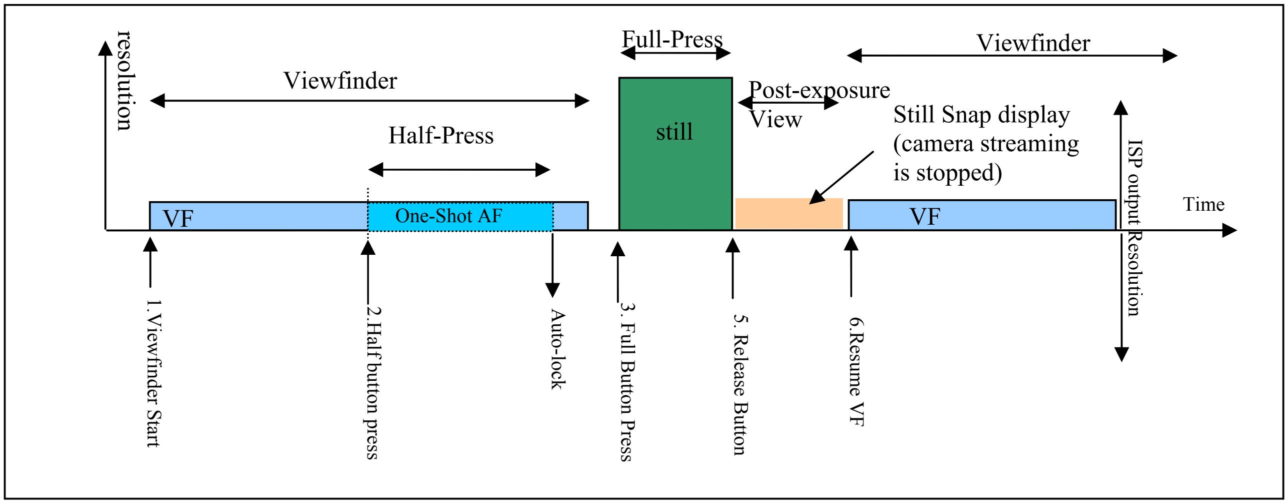


Figure 26. still preview, half-button period, full button, post-exposure view

If CAF was on, it completes this CAF phase by launching a one-shot AF search that starts from the last value computed by CAF. Half-press ending CAF goes faster than one-shot half-press since CAF generally already computated lens position close to the focus whereas one-shot AF started from arbitrary positions.

9.2.1 Sensor Settings

Same sensor mode as for preview generally is used for half-press. Framerate control is adapted (framerate can be decreased) to leave time for AF iteration and lens movement between 2 AF statistics collection.

9.3 Video Capture

The video capture starts when the end-user presses the button full-way.

During the very video recording, the VF generally is activated however this is not an obligation. Both the VF and video record datapaths need to be fully independent and no assumption can be made on the VF state.

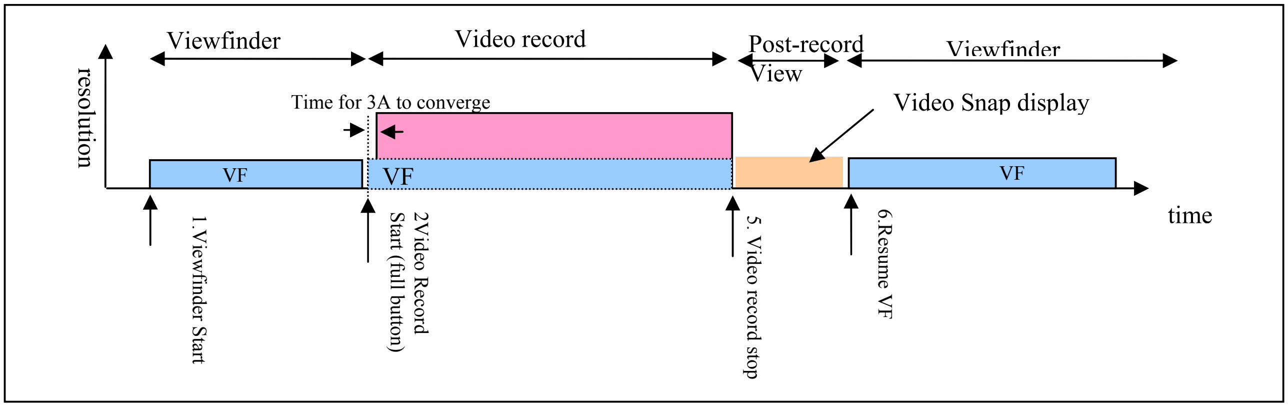


Figure 27. video preview, video capture, post-recording view

CAF is likely to run during VF and video record. Zoom can be changed during the very recording.

Video stab crop vector computation can be active during the video capture.

A video snap image (also referred to as video snapshot) generall needs to be generated. It corresponds to a low resolution uncompressed frame bound to be rendered on panel/TV during the so-called video post-recording view: It represents the first frame of the encoded sequence.

When the flash mode is forced on or auto, the video torch might be started when the full-button is pressed and stopped when the video capture ends. No video frames are streamed out from the pipe until they are correctly exposed, white-balanced and focussed. Flash decision is made automatically if the flash is in auto mode.

The targetted video resolutions are listed in Table 19.

Associated “+” resolutions also are supported, with adaptative overscan. Some examples are given in 26.

9.3.1 Sensor Settings

In video capture as well as in preview, sensor pre-scaling or sensor scaling is used. Constant framerate is sustained in accordance to downstream video encoder one. Also digital scaling can be used to decrease the output pixel clock.

For a target video resolution, a target framerate, there might different sensor modes used to cover the whole DZ range for this output resolution.

9.4 Slow Motion

It is also refered to as fast record use case. It consists in capturing video with very high framerate, typically 120 fps while the VF stays at 30 fps.

Supported video record resolution highly depends on the sensor capabilities. Dimensioning parameters are the ISP throughput when doing on-the-fly image reconstruction, 200 Mpel/s, the physical interface throughput and the sensor max throughput.

The VF can remain at 30 fps whereas the VPB2 framerate is 120 fps. The IL client is allowed to set framerates on VPB0 and VPB2 that are integer multiple of each other.

9.5 Still Capture during Video Record

This sub use-case consists in grabbing a still during the video capture. The main use case remains the video recording, meaning the recording generally must not be altered by the still capture. However this is arguable since some user experiences might priviledge the still compared to the outstanding video record, in which case we could afford doing a sensor changeover to satisfy at best the still requirements.

The still image reconstruction is done in SW since the HW ISP already is busy with on-the-fly reconstruction for VF and video. This means the still is grabbed in “raw” Raw Bayer; just after the receiver. The ISP recovery engine is not entered at all, meaning no 3A statistics are collected along with the still frame.

The still is shot with latest metering statistics, computed during latest video iterations. Static framerate control was used during video capture.

When shooting the still, the integration time is constrained by the video/VF framerate.

9.5.1 Sensor Settings

The working assumption is that there is no change in the sensor mode when the still image is captured. The rationale behind not changing the sensor mode is that no alteration of video/VF streaming is wished. As a consequence, the max still image resolution that can be captured is the sensor output at the time of the VPB1 capturing bit setting. This can be a pre-scaled resolution (typically after analog binning) of a cropped resolution.

If a crop mode is used in the sensor, this later contributes to the DZ. Anyway part of the DZ shall be implemented in the SW ISP or through a dedicated iv processor featuring and crop and scaling.

9.6 Still Image Capture

The OMX camera grabs stills in raw bayer (raw 8 or raw12) only. This raw frame then is reconstructed from memory to memory by the OMX ISP processor. For encoding into JPEG JFIF or EXIF the IL client needs to instantiate a stream featuring a camera, an ISP proc and a JPEG encoder.

The targetted still resolutions are listed in Table 17.

9.6.1 Single Still Image Capture

A single raw is captured. As soon as the still has been captured, either the snap is rendered on the display/TV or VF is resumed. This depends on the post-exposure mode.

9.6.1.1 Sensor Settings

The sensor settings are completely different from those used in still preview since grabbing a high Mpel frame generally cannot be done at the prevalent 30 fps VF framerate.

There is a sensor mode change inbetween the VF phase and the still capture and another one to resume the VF.

9.6.1.1.1 Sport Mode

In sport mode, where movement in the scene is assumed to be important, it makes sense to reduce the readout duration: this can be achieved by using sensor pre-scaling or crop sensor modes.

9.6.2 Burst Still Capture

Burst still image consists in shooting several still images (grabbed in raw bayer by the camera OMX component) at the maximum speed. In that use case, “true” VF at 30 fps is not resumed inbetween shots within the burst in order to minimize the shot to shot latency (this would oblige to change the sensor mode after each shot and probably stop the sensor, re-configure it and resume it). Only a snap image is rendered in between the shots. Those snap images act as fake VF frames (not streamed at 30 fps). Snaps also are stored in memory to be displayed after the shot sequence is over (typically in 2x3 grids).

Finite burst and infinite burst (also referred to as serial shooting) are distinguished.

In burst mode, 2 strategies can be enforced, leading to 2 different user experiences:

- achieve the best shot to shot performance (the sensor data is grabbed as fast as possible, even in the most raw format);
- achieve the best shot 2 display performance (snap image reconstruction and rendering to the panel/TV is more important than the capability to shoot again).

In both cases, it is assumed the image reconstruction or any post treatments of the stills can have less priority.

9.6.2.1 Fast Raw Capture

For best shot 2 shot performance, the “**fast-raw**” modality is the most adapted, as long as the sensor itselfs supports framerates faster than 200 Mpel/s. Fast raw consists in grabbing the raw on the raw pipe (pipe 2) without doing any image reconstruction (raw is output by BMS after the CSI-2/CCP receiver). This raw is not treated at all (it is a raw raw and not an ideal raw as usual). It corresponds to the sensor raw packed into memory. This allows capturing frames as fast as they are being output by the bayer sensor.

The actual image reconstruction can be delayed.

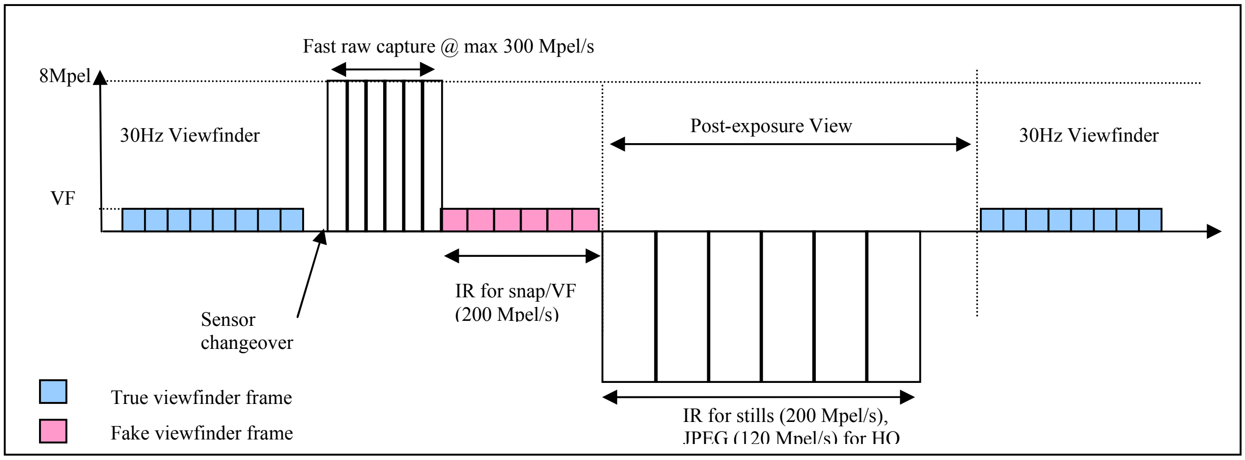


Figure 28. Fast Raw Capture

9.6.2.2 Ideal Raw Capture

This is the standard behavior of the OMX camera. The raw bayer that is output by the camera is partially treaded with:

- Channel offset correction
- Linearization
- Lens shading correction (LSC)
- Defect Pixel Correction.

See 7.1.1. Doing those treatments limits the ISP throughput to 200Mpel/s. Hence the capture speed is limited by the ISP throughput.

9.6.2.3 Sensor Settings

Burst still settings could be close to sport mode single still settings: each time a sensor mode that features pre-scaling or cropping can be used, it might be.

9.7 Pre-Capture Modality

Pre-Capture consists in grabbing raw bayer stills in a circular buffer while standard still preview is active. As soon as pre-capture mode is set, still frames are captured in a limited width circular buffer until the actual capture is notified. Although somehow similar to still capture during video record wrt sensor settings, the advantage of this use case is one is warned that stills must be captured. This makes possible to choose the best appropriate sensor mode when pre-capture mode is configured. Pre-Capture is a still image use case.

For OMX integration of this use case, please refer to 11.2.5

9.7.1 Sensor Settings

The camera OMX component programs the sensor so that 30 fps VF can be sustained. In general this corresponds to a ½ H&V binned configuration of a crop configuration for higher DZ.

10 Dataflows

10.1 Video Dataflow

The possible dataflows for video preview and record are given below. These dataflows are the ones which will be used at OSI level i.e. at OMX level.

There are three OMX components which are used for the main video record use case. Omx Camera (for retrieval of camera frames), Omx Video Encoder(for encoding the video stream coming out of Omx Camera) and Omx Display sink component (for displaying the preview frames coming from Omx Camera on the screen or writing them into memory before final display).

In between these Omx components we might have tunneling inplace.

But sometimes this becomes difficult due to constraints of the OS-dependent multimedia framework. For example, in Android the preview buffers are taken from the Omx Camera abstracted by the Camera HAL and directly sent to the Surface Flinger for displaying on the screen. Hence the Omx preview buffers are first provided to the Camera HAL then to the CameraService which then sends these to the Surface Flinger which runs in a separate process than the Camera HAL. The Surface flinger further relies on the copybit library which is built on top of B2R2 user library. Hence no display specific Omx component is used for the preview frames rendering.

For Video captured frames it is the PV multimedia framework or StageFright now that comes into the picture. Camera Service and hence inherently the Camera HAL is instantiated by the PV Author Camera Input Node which provides camera frames inside the PV dataflow.
The video frames coming out of the Omx Camera are provided to the Camera HAL, which sends then to the PV Author Camera Input Node and then to the PV encoder node for encoding (rather sending the data directly to the Video Encoder Omx component). There is no support for Omx tunneling in PV or StageFright. StageFright is the upcoming Multimedia framework for Android platform (introduced in Froyo release).
No Omx tunneling here implies that the video captured buffer from Omx Camera goes through the Camera HAL to the CameraService to the PV Author node which then sends it to the PV encoder node to the Video Encoder Omx component. Hence lots of SW overhead.

The details here for Android dataflows are merely for reference or example and do not intend to provide complete description of the Android dataflows and their associated constraints. For their complete description please refer to the /29/ .

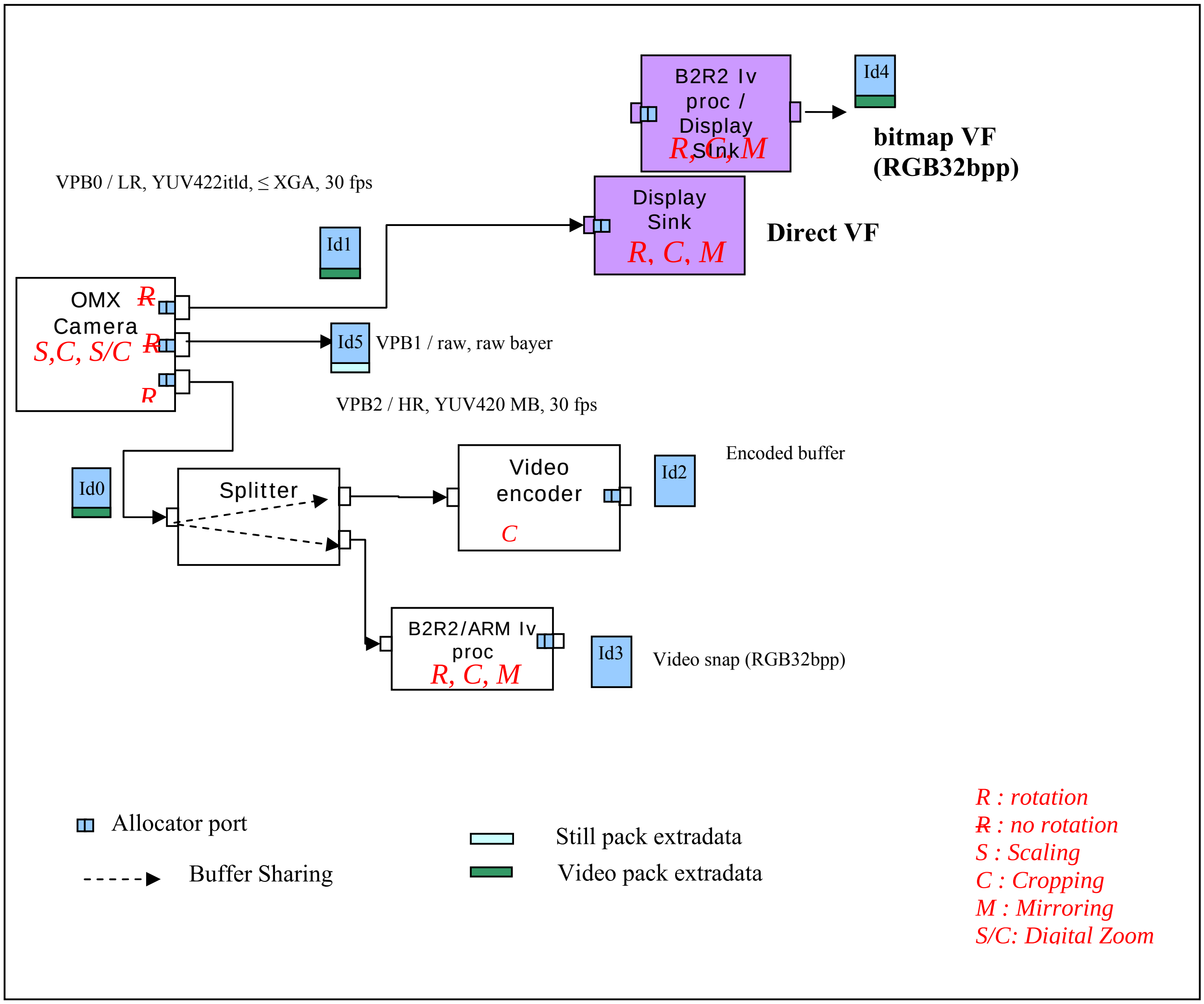


Figure 29. Video Capture OMX Dataflow

VF stream (camera VPB0 source) is limited to XGA resolution. On VF stream, rotation or stabilization cropping need to be implemented by a downstream iv processor/display sink, depending on whether the viewfinder works in bitmap mode or in direct rendering mode (only the direct mode is pictured above). On VPB0, YUV422 itld is assumed.

Id1 buffer is the preview buffer which goes to the Display Sink for direct rendering on the display side. Incase no direct rendering is required, the B2R2 IV Proc can be used. In this case there is an additional buffer that is used (Id4). In Android, the Id1 buffer is directly sent to the Surface Flinger and there is no Omx Component that is used for display.

There could also be associated constraints of default formats of buffers to be provided by the camera framework. For example in Android by default (if there is no specific user setting) the Camera HAL should output buffers in NV21 (YUV 420 SP) format. For such requirements certain adaptations are required in the dataflows which are addressed in the /29/ .

On video record stream Id0 buffers, YUV420 MB-tiled is recommended since ST video encoders are optimized for this format.

Since camera VPB2 outputs 420MB format and it is mapped onto the ISP HR pipe, HW supports rotation. In case of a SW encoder that would only support a YUV raster format, rotation would not be handled in the camera anymore.

In Android multimedia frameoworks and other multimedia frameworks we usually need to adapt it for this format to be exchanged between the camera and the encoder.

Then in video record we could also have a requirement for video snapshot which is the display of first frame of a video.

A snapshot could be generated in two ways.
By connecting a splitter at VPB0 followed by two IV Proc – one for viewfinder and other for snapshot. During the first frame the splitter passes the input buffer to the IV Proc for snapshot also alongwith passing it to the IV Proc for viewfinder display. After the first buffer the splitter passes on frames only to the IV Proc for viewfinder display. This logic needs to be implemented by the IL client by enabling and disabling the splitter port when required.

The second option is to use the VPB2 output buffer for video snapshot with the same logic as above.

The splitter used above is the non-sequential one.

There is no support today in the Android framework for such a snapshot support.

In case the IL client configures the camera to perform digital video stabilization, this latter outputs overscanned resolutions (also dubbed "+" resolutions) on VPB0 and VPB2. The "+" buffers are accompanied with the stabilization extradata so that the cropping can be performed in a downstream component (video encoder, iv processor/display sink). The camera component does not crop itself according to the crop vector it computed.

The passing of overscanned resolutions and extradata needs several adaptations in the Android framework due to constraints. This is addressed in the /29/ .

Digital zoom is fully implemented in the camera OMX component, on-the-fly, relying on the ISP crop and scaling stages located in the colour engines.

It is possible to capture a still during the video capture use case. In that case, camera VPB1 is used to capture a Raw Bayer frame. Image reconstruction would be done by a SW ISP during or after the video capture. Today there is no such SW ISP available on ST-Ericsson side.

Hence another option to support Still during Video use case by re-using the Video snapshot dataflow (without the logic for sending only the first frame). The splitter port would have to be enabled by the IL Client whenever such a capture is required.
The associated constraints would be of the resolutions that can be supported for such still captures. Only resolutions less than the video ones might be supported then. No extra processing would be possible on such resolutions in the RAW domain.

In case the TV is plugged, a single display sink is assumed to be used for both the rendering to the panel and to the TV. The working hypothesis is it is plugged onto VPB0. Max VPB0 output resolution is XGA, meaning there will be an upscale to the TV. If such upscale is not desired, dataflows would have to be re-worked.

10.1.1 Footprint

Buffer Id	Nature	Format	Size (B)	Number	Comments
Id0	Grabbed Video frame	YUV420 MB	Video or Video+ resolution x3/2	Configurable by IL Client	
Id1	Grabbed VF frame	YUV422 Itld	VF or VF+ resolution x2	Configurable by IL Client	VF makernote can only be exploited in case of bitmap VF
Id2	Encoded buffer	compressed	Depends on encoded stream	Configurable by IL Client	
Id3	Snap image	RGB32bpp	VF resolution x 4	1	To be confirmed
Id4	VF buffer	RGB32bpp	VF resolution x 4	Configurable by IL Client	Only exists in case of bitmap VF (not direct rendering with Display sink)
Id5	Still raw bayer	Raw bayer	Still resolution x1.5 (RAW10) Still resolution x1 (RAW8)	Configurable by IL Client	

Table 10 Buffers

Please refer to the MBL Footprints xls sheet (/26/ for eventual figures depending on number of buffers of each type.

Please refer to 25 for more details on extradata and content of still/video extradata “pack”.

10.2 Still Image Dataflow

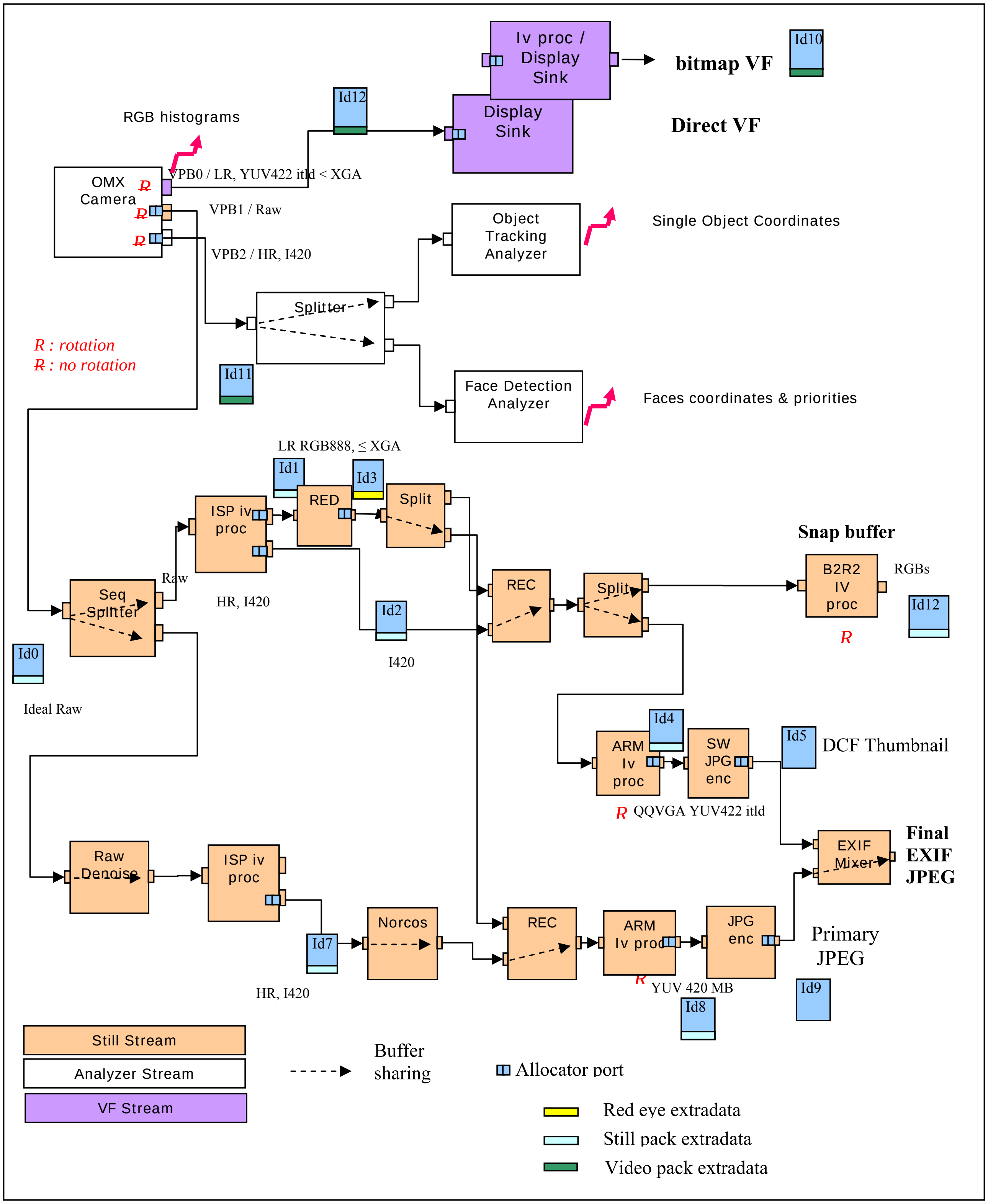


Figure 30. Single Still Dataflow

The Still dataflow provides an encoded still buffer alongwith embedded thumbnail. It also provides a snapshot to be displayed on the screen immediately after the shot. It also provides support for several advanced processings and algorithms like detection of faces, objects, red-eyes, etc. And pre and post processing algorithms such as denoising in RAW and YUV domains respectively.

VF and analyzer streams can be active at the same time (VPP0 and VPB2). However still stream (VPB1) and VF/analyzer streams are not active concurrently since they require different sensor settings.

The dataflow for burst still is the same than the one used for single still. Some blocks simply are bypassed.

It does not feature

- Digital image stabilization
- RED/REC
- Enhanced SW post-processings (DIS, dark frame substraction, Norcos noise reduction)

The number of buffers at various ports for burst dataflows is also different to ensure parallel processings on different domains.

10.2.1 Footprint

Buffer Id	Nature	Format	Size (B)	Comments
Id0	Grabbed “ideal” raw bayer still	Raw bayer (Raw10 or RAW 8)	Still x 1.5 (RAW 10) Still x 1 (RAW8)	Number depends on whether pre-capture or DIS is on
Id1	RED source	RGB888	Snap x 3	
Id2	Snap reconstructed source	I420	Snap x 3/2	
Id3	Red Eye metadata	Red eye struct		
Id4	Input of the DCF encoder	YUV422 interleaved	QQVGA x 2	
Id5	DCF thumbnail	JPEG	QQVGA x 1/8	
Id6	Viewfinder Buffer	RGB32 bpp	VF x 4	Only in bitmap mode
Id7	Still reconstructed source	I420	Still x 3/2	
Id8	Input of the primary JPEG encoder	YUV420 MB tiled	Still x 3/2	
Id9	EXIF JPEG	EXIF JPEG	Still x 1/8	1/8 is the compression factor
Id10	VF buffer	YUV422itld	VF x 2	Only exploitable in bitmap mode
Id11	Buffer for analyzers	I420	Analyis x 3/2	
Id12	Snap buffer	RGB32bpp	VF x 4	

Table 11 Buffers

Please refer to the MBL Footprints xls sheet (/26/ for eventual figures depending on number of buffers of each type.

Please refer to 25 for more details on extradata and content of still/video extradata “pack”.

10.2.2 VF Stream

Camera VPB0 is used for VF. It maps to ISP LR pipe. As such it is limited to XGA resolution. YUV422 itld format is assumed. No rotation is available on camera side. If VF needs to be rotated, the downstream iv processor/display sink needs to be used.

As in Video dataflow, we have similar constraints in still preview dataflow for Android. The preview buffers coming from the Omx camera is passed to the Camera HAL (IL client of Omx camera) which then sends to the Surface flinger directly that runs in a separate process from the Camera HAL.
The Surface flinger further relies on the copybit library to perform processings on the B2R2 HW through the B2R2 user library. No display specific Omx component is used in preview.

10.2.3 Analyzer Stream

Camera VPB2 is used for analysis purpose (it is started with a capturing bit as video capture). It maps to ISP HR pipe. Resolution is assumed to be VGA or nHD for 4:3, 16:9 aspect ratio respectively and format is supposed to be YUV420 raster (I420). However resolution will be tuned according to the bench results of analyzers. In practice the largest requested resolution is produced by the camera with possible downscale in some analyzers.

The face detection is usually performed at a lower framerate than the VF one (typically 15 fps for a 30 fps VF but never lower than 15 fps). In still preview, the framerate control generally is set to automatic. Framerates on both VPB0 and VPB2 only can be integer multiple of each other.

Anyway, the analysis framerate simply will be limited by the speed of the slowest analyzer (since the buffer will be pushed back to the camera component when all the analyzers have done their processing). The splitter used is the non-sequential one.

It must be noticed there is no rotation on camera VPB2 along with raster format. This means that the downstream analyzers must adapt to the device orientation (information contained in the scene characterization extradata). Also face information are likely to be used by analyzers, conveyed in the same extradata.

A tunnel can be established between these Omx components even incase of Android as the components in the Analyzer stream are totally contained in the Camera HAL (as per Froyo release).

The Camera HAL in Android would receive events from the analyzer components and provide them to the Omx camera by doing corresponding SetConfigs on the Omx camera.
This might not be true in some multimedia frameworks which would then require specific adaptations.

10.2.4 Still Stream

The still dataflow is split into 2 main sub-streams: the low resolution (LR) stream (used for snap/VF/DCF thumbnail generation) and the high resolution stream. Both streams use the same source, namely the camera VPB1 output port (Raw Bayer). The raw bayer image output by the camera component is already partially treated (it is abusively called “ideal raw” since it does not correspond to SMIA++ definition of ideal raw): offset removal, linearization, lens shading correction and defect pixel correction is already applied by the camera component.

When digital image stabilization is activated, the camera outputs the sharpest image out of n, grabbed in raw bayer.

These dataflows might be adapted in different multimedia frameworks in different ways. The dataflows shown here are ones to be used in the final Android reference implementation from our side with possible minor modifications for optimizations in certain use cases.

In Android, Camera HAL would contain the Omx camera alongwith the complete analyzer stream and still stream. Certain callbacks need to be implemented at certain timelines for the still dataflows such as Shutter Callback, YUV callback and JPEG callback. The JPEG encoder is also encapsulated inside the Camera HAL implementation. The final output from the Camera HAL for still use case is the JPEG compressed image.

10.2.4.1 Sequential Splitter

The split into the 2 sub-streams is managed by a sequential splitter. The component first supplies the buffer to the LR dataflow and then provides this latter to the still dataflow; this aims at optimizing the shot to snap latency: objective is to render the snap as fast as possible.

10.2.4.2 ISP Processors and Image Reconstruction

For both streams, the bulk of the image reconstruction is performed by the ISP IV processor instances.

The “snap” ISP IV processor uses the ISP parallel colour engines to generate 2 YUV’s:

- One on which red eye detection is performed (RGB888[1]) and
- Actual YUV buffer which would be used for snapshot and DCF thumbnail generation

The “still” ISP IV processors uses the ISP HR pipe to generate:

- the primary YUV (YUV420 raster) which will be eventually red eye corrected.

Note the primary YUV might also be treated with RAW processings (before the ISP Proc) and further YUV processings (after the ISP Proc) typically enhanced low light noise reduction (Norcos algo). This is optional and customer specific.

10.2.4.3 Viewfinder/Snap

In single still capture, the snap image is processed into a viewfinder bitmap by the B_2R_2 IV proc.

The IL client is in charge of displaying the snaps.

There could be different requirements for snapshot in different multimedia frameworks.

In Android for example the snapshot is provided in a full resolution YUV buffer inside a YUV callback from the Camera HAL. This buffer is then sent to the Surface Flinger for displaying it on the screen. Hence any processings required to be done on the YUV snapshot buffer would be done inside the Surface Flinger using the B2R2. In order to comply with this, the snapshot needs to be taken from the output of the “still” ISP Proc (at VPB2 port which is HR pipe) since the expected YUV should be of full resolution. This means that there might not be any need for having a parallel instantiation of the “snap” ISP Proc. This kind of dataflow would be totally new and has been addressed in the /29/ .

[1] LR pipe only generates RGB and YUV422 itld

There could be several associated advantages or dis-advantages associated with these dataflows and would also be customer specific depending on their requirements and priorities.

10.2.4.4 Red Eye Correction/Red Eye Reduction

Red eye detection (RED) is performed on RGB888 format. Red Eye Correction (REC) is performed on YUV 420 planar (I420).

RED and REC are performed on different resolutions (RED resolution being smaller). RED is performed on XGA resolution (advised when correction is done on a resolution inferior or equal than 8Mpel).

In our dataflows we use only one instance of RED on lower resolution which provides control metadata to the two instances of the REC connected in snapshot and still streams. However, this is optional and customer specific.

RED/REC is used along with flash use cases:

- Fast RED being activated when flash mode is set to auto/forced-on,
- Full RED activated when flash mode is set to flash red eye.

RED+REC is performed on the DZ WOI.

In case the flash is not lit, red eye metadata should be void and REC should simply perform buffer sharing. This means that even in bypass mode, the extradata must be exploited.

10.2.4.5 JPEG Encoders

ST-Ericsson HW accelerated JPEG encoder only supports YUV420 MBtiled as input. Thus the output of SW YUV correction algorithms need to be converted prior to the JPEG encoding. Current working assumption is that an ARM iv processor will be used to perform this YUV420 raster to YUV420MB conversion and potential rotation before encoding, if needed.

The DCF thumbnail is generated by utilizing the buffer from the snapshot dataflow and encoding through a SW JPEG Encoder which encodes input QQVGA or QVGA YUV 422 itld format buffers.

10.2.4.6 Colour effects and RED/REC

The current working assumption is in case colour effects hamper the behaviour of RED/REC, RED/REC shall be turned off automatically. Colour effect settings are part of the capture context extradata conveyed downstream to the camera. RED/REC shall automatically disable themselves according to those extradata.

10.2.4.7 TVout

In burst still capture, VF frames are likely to be displayed on-the-fly on the TV. Snaps might also be displayed on the TV but in a delayed manned (after the sequence, in the post-exposure view).

The same display sink is used for rendering on the panel and the TV.

Upscaling might be requested from XGA to TV resolution.

In Android the display is performed by the Surface Flinger and no display sink Omx component is used.

11 OMX Camera

The camera component enables to configure, control a sensor and retrieve grabbed buffers from its three data output ports.

It emits camera frames and statistics according to settings at its three output ports.

By camera it is meant the system composed of:

- the sensor
- the ISP (embedded ISP8500)
- the actuators (optical zoom, focus, mechanical shutter)
- the flash ICs

There is one implementation of this OMX component for the primary camera and another one for the secondary camera.

Primary and secondary camera components cannot be simultaneously in active state (idle/pause/executing) since they share the same ISP resource and execute different XP70 firmwares. The resource manager will prevent this from happening.

The camera OMX component is in charge of booting the ISP.

The OMX camera component itself does NOT implement JPEG encoding. A client wishing to encode the grabbed data needs to connect a downstream OMX JPEG encoder.

The camera performs on-the-fly image reconstruction only in preview and video modes. In still mode, it does not perform raw bayer reconstruction, ie. the only format supported in still is Raw Bayer output. As such, in still the raw bayer reconstruction must be implemented from memory to memory in a downstream ISP proc OMX component.

The ST-Ericsson OMX camera supports the following main use cases.

Use Case Category	Operative Mode Sub-Usecase	Comments
Preview	Standalone Still Preview	Viewfinder preceding a still capture
	Standalone video Preview	Viewfinder preceding a video capture
Still Image Capture	Single Still Image Capture	A single still is captured
	Burst Still Image Capture	A burst of n stills is started; stills are captured as fast as possible
Video Capture	Standard Video Capture	Video capture is started
	Still Capture during Video Capture	A still is captured in the middle of a video capture

Table 12 Operative Modes & Use Cases

The following use cases are not directly supported by the camera OMX component:

- Time lapse (a sequence of n stills is started; stills are captured at a constant time interval),
- Self Timer or Timed mode (the single shot happens after a programmed timeout value) and
- bracket merge mode (N stills are captured but only a single one is streamed out in SDRAM: typically used for still image stabilization, could be used as well for bracket merge).

Besides the operative mode, the rotation, and the mirroring, the following settings are supported by the camera components.

Properties that are component wide, can not be controlled individually for each port. Component wide indexes are used to control properties which can not be controlled individually for each port. Examples of such are e.g. sensor and flash settings.

	Primary				Secondary			
Settings	Still Preview	Still Capture	Video Preview	Video Capture	Still Preview	Still Capture	Video Preview	Video Capture
DZ	Max x20, to be precised				Max x4, to be precised			
Pre-capture warning	X	/	/	/	/	/	/	/
Metering control/lock	X	X	X	X	/	/	/	/
Pan-tilt	X	X	X	X	X	X	X	X
Bracketting	/	X	/	/	/	/	/	/
Framerate control	Auto Except sport and long exposure	/	static	static	auto	/	static	static
Scene Mode	Auto, Portrait, Landscape, Night Scene, Night portrait /Party, Sports, Macro, Document, Beach, Snow				/	/	/	/
Exposure Mode	Off, Auto, Night, BackLight, Sports, Spotlight, Snow, Beach, LargeAperture, SmallApperture, FacePriorityMode, Center, VeryLong, HwFunctionalTesting				Auto, night			
EV compensation	[-2 ; +2], +/-1/3 step		/	/	/	/	/	/
Metering Mode	X	X	X	X	/	/	/	/
ISO Setting	Auto, 100, 200, 400, 800, 1600, 3200		/	/	/	/	/	/
Flicker cancellation	X	X	X	X	/	/	/	/
WB preset	Off,Auto,SunLight,Cloudy,Tungsten,Fluorescent, Incandescent,Shade,Custom,Horizon,FacePriorityMode,Flash				auto			
Digital Video Stabilization	/	/	X	X	/	/	/	/
Color Tones/Effects					/	/	/	/
aperture	X	/	X	/	/	/	/	/
Shutter speed	X	/	X	/	/	/	/	/
One-Shot Auto-Focus	X	/	/	/	/	/	/	/
Continous Auto-Focus	X	/	X	X	/	/	/	/
YUV Range	X	/	X	X	X	X	X	X
Contrast	X	X	X	X	X	X	X	X
Brightness	X	X	X	X	X	X	X	X
Sharpness	X	X	X	X	/	/	/	/
Saturation	X	X	X	X	/	/	/	/
Flash	AF assitant	X	/	X	/	/	/	/

Table 13 Settings

It is reminded the secondary camera has a fixed focus.

The following settings have an OMX_ALL semantic (they apply to the recovery engine):
- flicker cancellation,
- metering lock,
- pan-tilt (sensor crop),
- exposure mode
- EV compensation,
- metering mode,
- ISO setting,
- WB preset
- DZ (although partially implement in the CE)

Other settings and especially those which apply to colour engines apply to specific port indexes
- colour effect
- YUV range (API not yet defined)
- contrast,
- brightness,
- saturation
- sharpening.

3A (automatic exposure control, automatic white-balance control and auto-focus are abstracted by the camera) by integrating the STE 3A library.

Component name : OMX.STE.CAMERA.PRIMARY, OMX.STE.CAMERA.SECONDARY
Component Role : camera.3outp.vf_image_video

11.1 Ports Settings

The camera exposes 3 output ports as opposed to the standard camera component.

- VPB0 is not associated with any "capturing bit" (it starts streaming as soon as the component is moved to executing and the port is enabled). It is used for viewfinder.
- VPB1 is used to capture still image in Raw Bayer format only (also during video record).
- VPB2 is used for video capture or analysis purpose.

VPB1 and VPB2 are granted with a separate "capturing bit". By "capturing bit" it means that frames would be outputted from the port only after this capture bit is set. This would be true for VPB1 and VPB2 ports as explained above.

11.1.1 VPB0

Mapped onto ISP LR pipe. Width is limited to XGA (1024x768) due to HW limitation. Only YUV422 itld is supposed to be used (OMX_COLOR_FormatCbYCrY). No rotation is available. Buffers need to be contiguous. Slice height always equals to frame height.

Format	Digital Video Status	Width (pixels)	Height (pixels)	Stride (pixels)	Start address alignment
YUV422itld	off	4	1	8	8
YUV422itld	on	8	16	8	8

Table 14 Alignment constraints on VPB0

The targetted standard VF resolutions are

Resolution	Width	Height	A:R
XGA (~0.8 Mpel)	1024	768	4:3
qHD (~ 0.5 Mpel)	960	540	16:9
WVGA (0.39 Mpel)	854	480	16:9
VGA (~0.3 Mpel)	640	480	4:3
nHD (~0.21 Mpel)	640	360	16:9
CIF	352	288	11:9
QVGA	320	240	4:3
QCIF	176	144	11:9

Table 15 Target VF Resolutions

Supported framerates are up to 30 fps. Automatic framerate is put in place in still (except in sport mode), varying into a preset range. Static framerate is observed in video preview and still sport mode.

11.1.2 VPB1

Mapped onto ISP raw pipe. Rotation is not available on that port. Few resolutions are available since there is no raw bayer scaler meaning the resolutions available are those the sensor output (as such depends on sensor mode). Buffers need to be contiguous. Slice height always equals to frame height.

The formats supported by this port are:

- OMX_COLOR_FormatRawBayer8bit,
- OMX_COLOR_FormatRawBayer12bit,
- OMX_COLOR_FormatRawBayer8bitcompressed

Format	Width (pixels)	Height (pixels)	Stride (pixels)	Start address alignment
RawXXX	2	2	2	8

Table 16 Alignment constraints on VPB1

Resolution	Width x height	A:R
20 Mpel	5280x3960 (tbc)	4:3
18Mpel	5024x3768 (tbc)	4:3
12 Mpel	4000 x 3000	4:3
	4000 x 2248	16:9
8.1 Mpel	3264x2448	4:3
	3264x1832	16:9
5 Mpel	2592x1944	4:3
	2592x1456	16:9
3.1 Mpel	2048x1536	4:3
3 Mpel	2592x1458	16:9
HD	1920x1080	16:9
2 Mpel	1600x1200	4:3
WXGA	1280x720	16:9
VGA	640x480	4:3
CIF	352x288	11:9
QVGA	320x240	4:3
QCIF	176x144	11:9

Table 17 Supported Still Resolutions

Resolutions whose width is larger than 4096 are supported through stripes.

11.1.3 VPB2

Mapped onto ISP HR pipe. The two formats exposed by this port are:

- YUV420 raster (OMX_COLOR_FormatYUV420PackedPlanar), I420
- YUV 420MB (OMX_COLOR_FormatYUV420MBPackedSemiPlanar)

Buffers need to be contiguous. Slice height always equals to frame height. Rotation is available along with YUV420 MB only.

Format	Digital Video Status	Width (pixels)	Height (pixels)	Stride (pixels)	Start address alignment
YUV420 raster	off	8	2	8	8
YUV420 MB	off	8	16	16	16
YUV420 MB	on	8	16	16	16

Table 18 Alignment constraints on VPB2

The supported resolutions on that port are listed below. Also overscanned resolutions are supported for stabilization purpose. Some examples are given in 26.

Resolution	Width	Height	A:R
1080p (Full HD)	1920	1080 (not 16pix aligned)	16:9
720p (HD)	1280	720	16:9
VGA	640	480	4:3
CIF	352	288	11:9
QVGA	320	240	4:3
QCIF	176	144	11:9
subQCIF	128	96	4:3

Table 19 Supported Video Resolutions

11.1.4 Framerates

The preview and video frame rates requested are 30, 24, 15, and 10. Variable frame rate control is requested for still-preview mode.

11.2 Operative Modes

The operative mode can be selected through the OMX indexes summarized in the table below. Their semantic is detailed for our 3-port camera.

OMX_Index/Struct	Field	Default Value	Khronos	Extensions	Semantic
OMX_IndexParamCommonSensorMode/ OMX_PARAM_SENSORMODETYPE	bOneShot	1	1.1.2		bOneShot = 1 indicates still is the main use case bOneShot = 0 indicates video is the main use case bOneShot makes possible to recognize the use case and determine the policy for A:R, sensor mode settings choice
	nFramerate	Never interpreted by the camera OMX component	1.1.2	/	
	sFrameSize	RO: Omx Camera updates this field according to the actual sensor mode	1.1.2	/	
OMX_IndexConfigCaptureMode/ OMX_CONFIG_CAPTUREMODETYPE Restricted to VPB1	bContinuous	0	1.1.2	/	Used to know whether the xFramerate set on the port is obeyed or capture is performed as fast as possible
	bFrameLimited	1	1.1.2	/	Used to indicate whether the capture is limited to a burst count=nFrameLimit or is infinite
	nFrameLimit	1	1.1.2	/	Burst count (number of finite bursts) if bFrameLimited = 1
OMX_IndexAutoPauseAfterCapture/ AutoPauseAfterCapturing Restricted to VPB1		0	1.1.2	/	Applies when the capture bit of VPB1 is reset.
OMX_Symbian_IndexConfigCommonExtCapturing/ Port specific capture bit. Restricted to VPB1 and VPB2	nPortIndex		/	SHAI 1.2	Enables to start streaming on VPB1 or VPB2 in case of finite capture, the capturing bit is reset automatically by the camera after nFrameLimit frames streaming. If the IL client resets this capture bit while ongoing capture, capture needs to be cancelled. In case of infinite sequence the capturing bit must be reset by the IL client.
OMX_Symbian_IndexConfigExtCaptureMode Restricted to VPB1	nFrameBefore	0	/	SHAI 1.2	Size of the ring buffer when bPrepareCapture = 1. Frames are started to be filled as soon as nFrameBefore is made non zero but buffers are not streamed until capturebit is set
	bPrepareCapture	0	/	SHAI 1.2	bPrepareCapture = 1 indicates

					that capture of nFrameBefore should be done. It infact enables this mode.
	bEnableBracketting	0	/	SHAI 1.2	= 1 enables the EV bracketting

Table 20 OMX indexes and semantic

The 3 ports can be active at the same time.

The next chapters describe the settings for each operative mode of the 3-port camera.

11.2.1 Preview

VPB0 is supposed to be used for preview streaming. To start the preview streaming, VPB0 must be enabled and the camera component must be in executing state. The camera component stops emitting VF frames when either the VPB0 port is disabled or the component exits the executing state.

OMX_IndexParamCommonSensorMode.bOneShot index enables to discriminate between still preview and video preview. It indicates the still or video intent (bOneShot = 1 means still is the main use case). Default value is 1 (still intent).

Video and still previews are characterized by different low-level settings (typically the framerate control strategy is different in still preview and video preview).

The preview framerate/resolution of the viewfinder corresponds to the framerate/resolution of VPB0 port. Prevalent framerate is 30 fps.

When the VPB1 capturing bit is set, no frame is output on VPB0 until the capturing bit is reset (this is due to the fact that VF and still usually requests incompatible sensor modes). When VPB2 capturing bit is set, VPB0 and VPB2 can stream concurrently, with the restriction they have the same framerate.

It is possible VPB1 or VPB2 capturing bits are set while VPB0 has never streamed. In that case the first frame output by either VPB1 or VPB2 must be correctly exposed and white balanced. As soon as the component moves to executing and even if no port is active/streaming, AEC and AWB might be transparently started. AF would be started only if enabled at OMX IL client level.

VPB0 always is implemented upon ISP LR pipe. As such its resolution is limited to XGA. On top of that, LR pipe only supports RGB fomats and YUV422itld. YUV422 itld is assumed.

11.2.2 Single Still Image Capture

This is also referred to as single shot.

Still image capture is meant to be implemented through VPB1 streaming. The following OMX settings must be used to stream a single frame.

OMX_Index/Struct	Field	Value
ParamCommonSensorMode/ OMX_PARAM_SENSORMODETYPE	bOneShot	1
	xFramerate	Don’t Care

	sFrameSize	Don’t Care
ConfigExtCapturing	bEnabled Index	1 VPB1
ConfigCaptureMode/ OMX_CONFIG_CAPTUREMODETYPE	bContinuous	Don’t Care
	bFrameLimited	1
	nFrameLimit	1
AutoPauseAfterCapturing		1 if post-exposure view, 0 in the contrary

Table 21 Single Still Image OMX Configuration

The start of the still capture phase is indicated by setting the capture bit on VPB1 (OMX_Symbian_IndexConfigExtCapturing). With above settings, a single frame is captured as fast as possible on VPB1 (Raw Bayer format only). From the OMX camera component perspective, the still capture phase stops as soon as the camera outputs the still buffer on VPB1. The associate buffer header is flagged with OMX_BUFFERFLAG_EOS and accordingly the camera emits an End Of Stream (EOS) OMX_EventBufferFlag event at that time. At that time the VPB1 capturing bit is automatically reset by the camera. Also a functional event is triggered at that time to notify the client a new exposure has been performed (OMX_Symbian_IndexConfigExposureInitiated).

Incase the IL Client resets the capture-bit before the EOS event, the capture should be cancelled.

The VPB1 capturing can be set even if VPB0/VPB2 are streaming.

However, when the capture is ongoing on VPB1 and bOneShot = 1, no more buffers are streamed out from VPB0/VPB2 until the capture is over. (until VPB1 capturing bit is reset). This is due to the fact VPB0 and VPB2 are meant for high framerate streaming (~15/30 fps) which require a different sensor mode than still one.

In case bOneShot=0 and VPB1 capturing bit is set, this corresponds to Still capture during video record use case. In that case, VPB0/VPB2 streaming must not be affected by VPB1 capturing bit.

11.2.2.1 Autopause after capture

If the *AutoPauseAfterCapturing* bit is set, the camera component is automatically paused after the shot, emulating the post-exposure view modality (freeze of the VF). The pause state is entered when the capturing bit is reset by the component itself. So after issuing the EOS event, the Omx camera resets the capture-bit and moves into the pause state.

In case there is no auto-pause, the VPB1 port stops streaming and a sensor changeover (sensor mode change) happens in order to accommodate fast streaming on other active ports, if any. Typically VPB0 and VPB2 might resume streaming. So after issuing the EOS event, the Omx Camera resets the capture-bit and performs a sensor change-over to resume viewfinder.

11.2.3 Burst Still Image Capture

It is also implemented by VPB1 streaming.

The burst can be **finite** or **infinite**.

The burst stills are captured on VPB1 under the form of “ideal” raw. VPB0 and VPB2 are allowed to be active at the same time but they will not produce any data until the capture is over, due to incompatibility of sensor modes.

The start of the burst is indicated by setting the capturing bit on VPB1 (OMX_Symbian_IndexConfigCommonExtCapturing). From the OMX camera component perspective, the burst phase stops as soon as the camera outputs the last still buffer on VPB1 if bFrameLimited is set or when the capturing bit is cleared by the IL client, in infinite mode (bFrameLimited = 0).

If the *AutoPauseAfterCapturing* bit is set, the OMX camera is paused when the capture bit is reset (in both cases above).

In case the auto-pause modality is not set, after the reset of the capturing bit, the VPB1 port stops streaming and a sensor changeover happens so that VPB0/VPB2 if active can resume streaming normally.

For each output still buffer, the associate buffer header is flagged with OMX_BUFFERFLAG_EOS and accordingly the camera emits an End Of Stream (EOS) OMX_EventBufferFlag event at that time. This currently makes possible the IL client renames the output file name, even in tunneling mode. Also a functional event is triggered for each new exposed frame.

OMX_Index/Struct	Field	Value
ParamCommonSensorMode/ OMX_PARAM_SENSORMODETYPE	bOneShot	1
	xFramerate	Don’t Care
	sFrameSize	Don’t Care
ConfigCaptureMode/ OMX_CONFIG_CAPTUREMODETYPE	bContinuous	Don’t Care
	bFrameLimited	1 if finite, 0 if infinite
	nFrameLimit	Frame count if finite, else don’t care
AutoPauseAfterCapturing		1 if post-exposure view, 0 in the contrary

Burst Still Image OMX Configuration

It is possible to end an outstanding infinite sequence by changing the value of *OMX_CONFIG_CAPTUREMODETYPE.bFrameLimited.nFrameLimit*. In that case, nFrameLimit are streamed out from the time the change is made on.

Conversely, it is possible to turn a finite sequence into an infinite one.

Infinite burst must be closed by an explicit clearing of the capturing bit. The IL client is allowed to cancel an outstanding finite burst by resetting the capturing bit itself.

The sensor is not stopped in between shots.

11.2.4 Video Mode

The video is captured on VPB2. Other ports are allowed to be active

OMX_Index/Struct	Field	Value
ParamCommonSensorMode/ OMX_PARAM_SENSORMODETYPE	bOneShot	0
	xFramerate	Don’t Care
	sFrameSize	Don’t Care
ConfigCaptureMode/ OMX_CONFIG_CAPTUREMODETYPE	bContinuous	Don’t Care
	bFrameLimited	Don’t Care
	nFrameLimit	Don’t Care
AutoPauseAfterCapturing		Not supposed to be used

Video Mode Configuration

The start of the video capture phase is indicated by setting the capturing bit on VPB2 (OMX_Symbian_IndexConfigCommonExtCapturing). The VPB2 streaming ends when the VPB2 capturing bit is cleared by the IL client.

When the camera outputs its last VPB2 buffer (upon clearing of the capturing bit), the associated buffer header is flagged with OMX_BUFFERFLAG_EOS and accordingly the camera emits an End Of Stream (EOS) OMX_EventBufferFlag event at that time.

AutoPauseAfterCapture is not supported for VPB2.

11.2.5 Pre-Capture Mode

It is possible to activate **pre-capture** modality before shooting a single still.

If the pre-capture is enabled through OMX_Symbian_IndexConfigExtCaptureMode.bPrepareCapture and if *nFrameLimit* is non zero, the camera component streams stills to SDRAM (through VPB1) during VPB0/VPB2 activity, without notifying either the downstream tunneled component nor the client (it does not call the FillBufferDone callback or the EmptyThisBuffer on the tunneling component).

When the VPB1 capturing bit is set, the *nFramesBefore* frames collected before the VPB1 capturing bit setting are output through VBP1 as well as the *nFrameLimit* frames succeeding the capturing bit setting.

The pre-capture should not prevent from generating VPB0/VPB2 frames and typically should not alter the target framerate. This means the sensor settings must accommodate both a video/VF streaming, typically at 30fps and a still capture. There is no plan to do any sensor changeover.

Configuration of pre-capture mode is theoretically possible to be set when streaming. In case it happens, a port settings change event should be sent to the IL client to notify this latter the buffer requirement have changed for VPB1 port, in case this latter was active.

Indeed when pre-capture mode is set, the VPB1 buffer pool must be redimensionned so that *nFramesBefore* min buffers are allocated. The camera component uses this external pool to store the temporary frames. In case of tunneling, this means the client needs to recycle the tunnel. Upon which, the camera OMX component will re-allocate the right number of buffers.

If *AutoPauseAfterCapture* was set, the camera enters pause state after the *nFramesBefore+nFrameLimit* frames are output into SDRAM. The capturing bit is reset when all those frames are output into SDRAM. The IL client is allowed to cancel an outstanding pre-capture mode sequence after the capturing bit has been set and before the sequence is over. In that case the currently collected frames are output to SDRAM (nFramesBefore + current number of frames).

For each output frame, the associated buffer header is flagged with OMX_BUFFERFLAG_EOS and accordingly the camera emits an End Of Stream (EOS) OMX_EventBufferFlag event at that time. This is started after the capture bit is set.

11.2.6 Still Capture during video capture (VPB1)

It is possible to shoot a raw bayer still image when video is captured. This corresponds to bOneShot set to 0 (indicating the main use case is video) and VPB1 capturing bit set while VPB2 capturing bit is already set.

From the sensor settings point of view, this use case brings the same constraints as pre-capture modality except a single frame is grabbed.

In that case the raw capture uses ISP8500 raw pipe, VPB2 streaming relies on HR pipe whereas VPB0 can be active, relying on LR pipe.

The AutoPauseAfterCapture is not supposed to be used in that case.

11.3 Frame Rate Control

Different framerates are allowed on VPB0, VPB1 and VPB2 port as long as they are integer multiple of each other. For example it is possible the following configuration:
VPB0: 30 fps
VPB2: 15 fps
VPB1 : 1 fps

Framerate is meaningful on all ports, even on the VPB1 port. On VPB1 port framerate is used to limit the still capture rate in serial capture use case (bFrameLimited = false and bContinous = false).

	.bOneShot = true (still mode)	.bOneShot = false (video mode)
VPB0 and VPB2 xFrameRate = zero	Viewfinder frame rate decided by camera component. In still capture case, variable frame rate is used in other than night and sport modes. IL client is allowed to set xFrameRate != 0 on VPB1 (serial capture). Then the camera component must accommodate VPB1 framerate.	Viewfinder frame rate selected by the camera according to the exposure mode.
VPB0 or VPB2 xFrameRate = non-zero	Viewfinder frame rate selected by client by setting xFramerate on either of the 2 ports. An error occurs if the IL client attempts to select non integer multiple framerates on VPB0, VPB1 and VPB2	Frame rate selected by client. VPB2/VPB0 are allowed to have framerates that are integer multiple of each other. For instance VPB0 is 15 fps and VPB2 is 30 fps or conversely and VPB0 = 30 and VPB2 = 120 fps). Then the camera OMX component implemens frame skipping on the lowest speed port. Other cases return an error.

Table 22 OMX Framerate Control

11.4 Digital Zoom & Pan-tilt

The camera OMX component abstracts DZ in preview and video capture mode only.

On-the-fly DZ (no intermediate buffer) is implemented in video mode.

In still capture mode, the camera grabs a raw bayer image. There is no raw bayer scaling in the ISP. As such the DZ is not implemented by the camera OMX component in still. It is implemented by the ISP processor, off-line, memory to memory.

In all cases the key block that implements DZ is the ISP crop and the General Purpose Scaler located in the two color engines.

In preview and video operating modes, DZ is implemented in the camera OMX component:

- Sensor cropping might be used
- Rest of the cropping is implemented in the ISP crop stage
- Eventual scaling to the output resolution is implemented in the ISP, on-the-fly.

In Still mode, the DZ is implemented in the ISP processor:

- Sensor full resolution usually is used (except maybe in sport mode where readout shall be reduced)
- DMA vertical cropping might be used during Bayer Memory Load
- Rest of the cropping is implemented in the ISP crop stage
- Eventual scaling to the output resolution is implemented in the ISP General Purpose Scaler

Digital zoom is a dynamic configuration. It is set through the OMX_Symbian_IndexConfigExtDigitalZoom. The OMX_Symbian_CONFIG_ZOOMFACTORTYPE.xZoomFactor corresponds to the crop ratio. The zoom config applies to all output ports (OMX_ALL). This is part of the extensions SHAI 1.2.

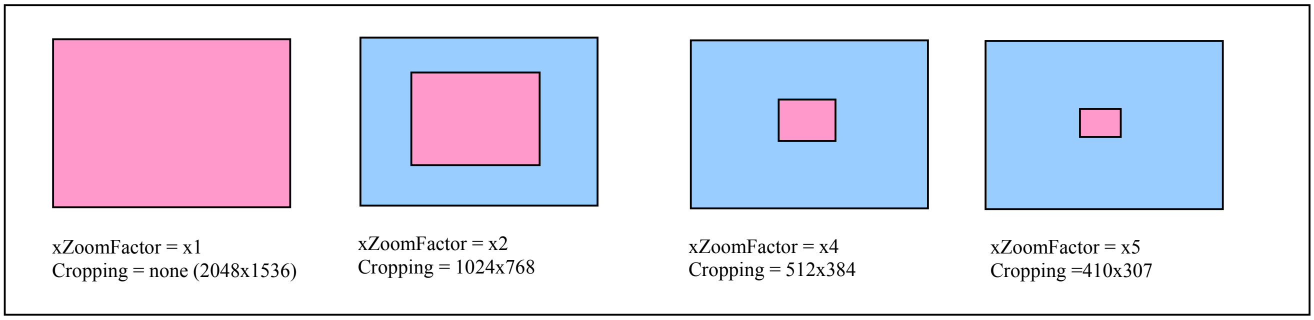


Figure 31. DZ factor Example (3Mpel Source Frame)

There is a similar configuration for optical zoom, OMX_Symbian_IndexConfigExtOpticalZoom, OMX_Symbian_CONFIG_ZOOMFACTORTYPE.xZoomFactor. However, no module with OZ currently is foreseen. This is part of the extensions SHAI 1.2.

OMX_IndexConfigCommonDigitalZoom and OMX_IndexConfigCommonOpticalZoom indexes are not used. OMX_IndexConfigCommonScale and OMX_IndexConfigCommonScaleQuality are also NOT supported by the Omx camera.

The ISP FW is responsible for choosing the sensor mode that is best fitted to the DZ target.

The figure below illustrates a x2 DZ where the sensor outputs a pre-scaled resolution of its FFOV. When the image is output from the sensor there is still no DZ applied (intrinsic DZ of the sensor mode is 1, since the region of interest still corresponds to the FFOV, even if it is downscaled). In that case DZ is fully implemented on ISP side, using the crop and general purpose scaler located in each colour engine pipe.

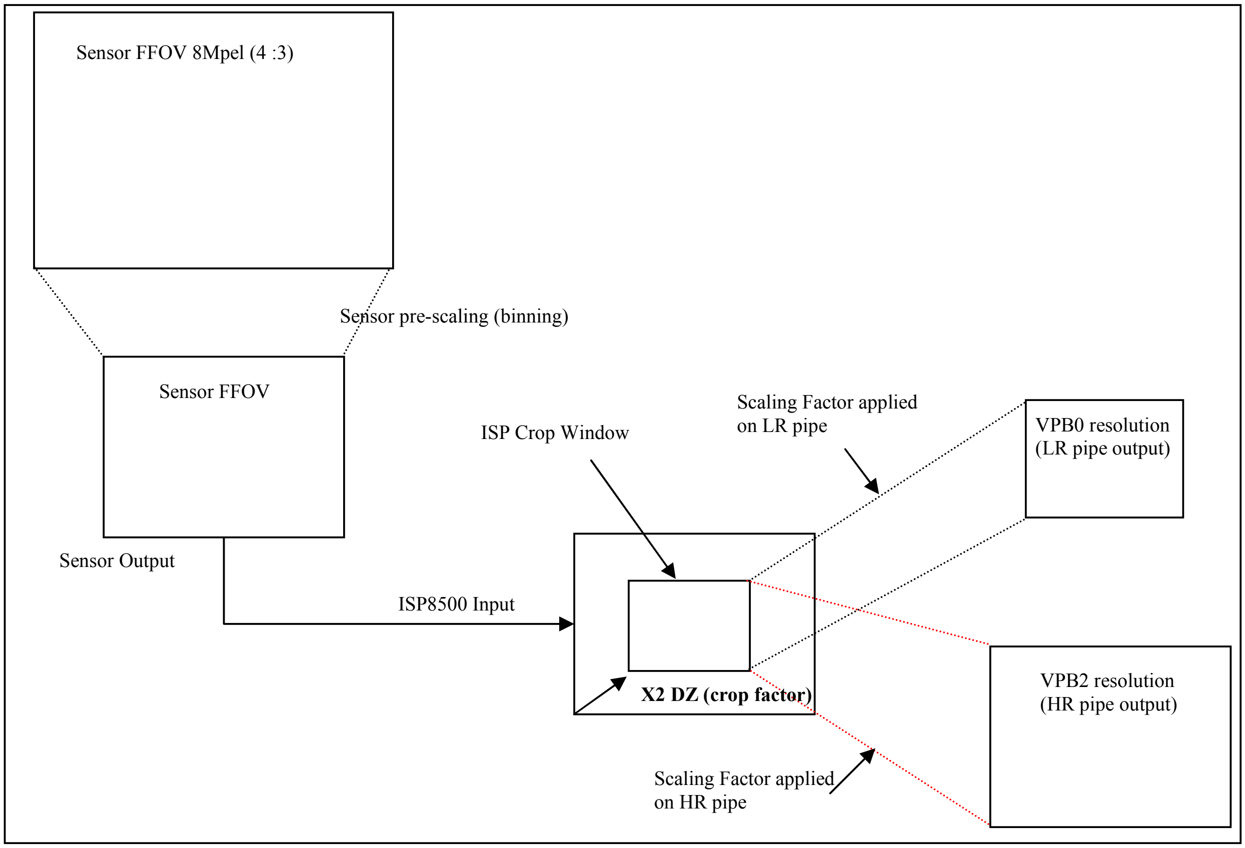

Sensor FFOV 8Mpel (4 :3)
Sensor pre-scaling (binning)
Sensor FFOV
Sensor Output
ISP8500 Input
ISP Crop Window
Scaling Factor applied on LR pipe
VPB0 resolution (LR pipe output)
X2 DZ (crop factor)
VPB2 resolution (HR pipe output)
Scaling Factor applied on HR pipe

Figure 32.   x2 DZ featuring sensor pre-scaling, ISP crop and ISP scaling

Digital zoom must not be mixed up with the scaling factor applied between the region of interest and the destination (VF or still image target resolution). The formulae linking digital zoom and scaling is reminded below.

$$DigitalZoom = Scaling \cdot \frac{source}{destination}$$

The figure below illustrates a x4 DZ. This time the sensor outputs a cropped area of its FFOV (x2 sensor cropping), ie. the image output by the sensor already features a DZ of x2. Then the cropping is completed by an ISP cropping of 2, leading to 4x DZ.

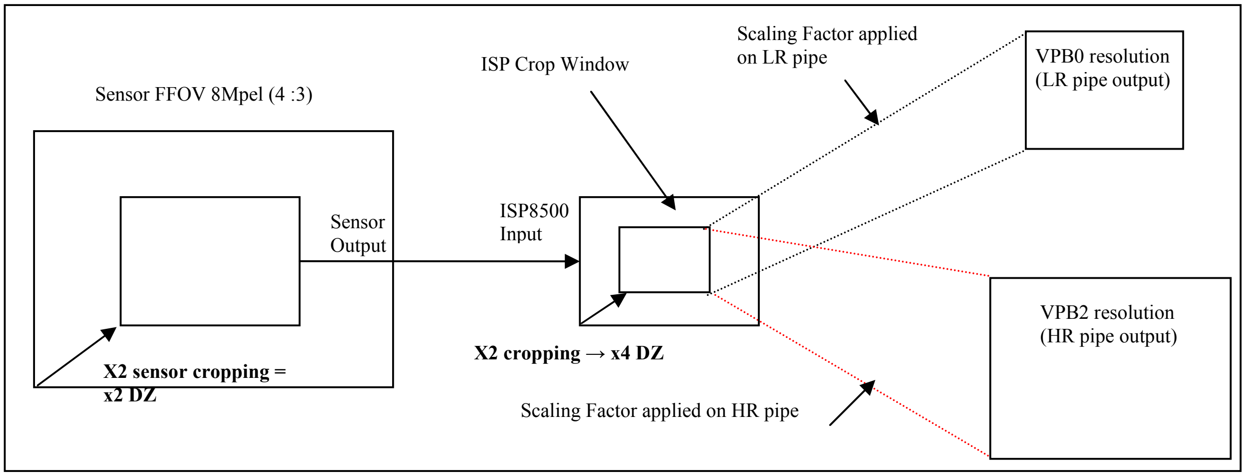

Sensor FFOV 8Mpel (4 :3)
ISP Crop Window
Scaling Factor applied on LR pipe
VPB0 resolution (LR pipe output)
Sensor Output
ISP8500 Input
VPB2 resolution (HR pipe output)
X2 sensor cropping = x2 DZ
X2 cropping → x4 DZ
Scaling Factor applied on HR pipe

Figure 33. x4 DZ featuring sensor cropping, ISP crop and ISP scaling

Again, the ISP FW is in charge of choosing the sensor mode and apply complementary settings in the ISP pipe.

The DZ position is reached as fast as possible. In case the application wishes to change the zoom at a given speed, it must emulate this speed by choosing the zoom factor appropriately.

The camera supports digital zoom in a restrained window of interest as pictured below. This typically happens when the secondary camera grabs a lower resolution that the sensor FFOV, in video mode. This is called SW titling.

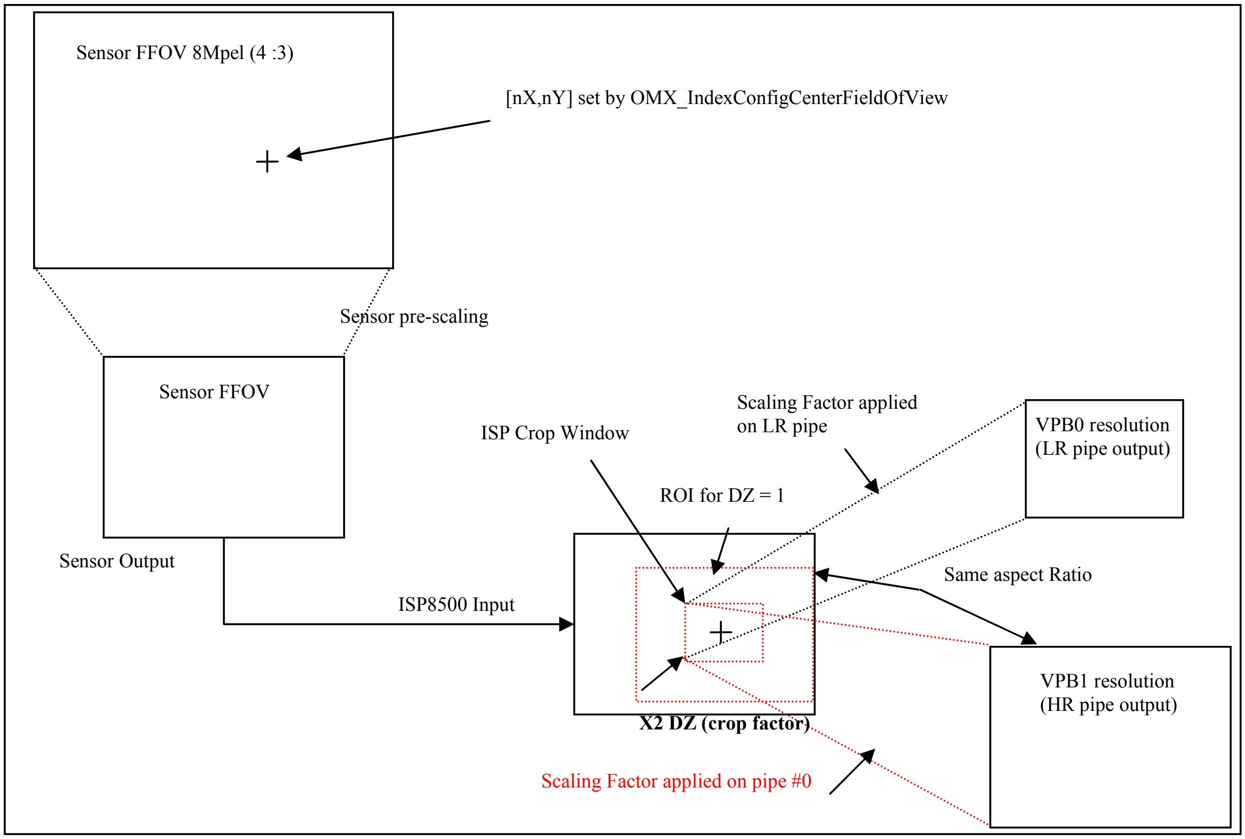

Sensor FFOV 8Mpel (4 :3)
[nX,nY] set by OMX_IndexConfigCenterFieldOfView
Sensor pre-scaling
Sensor FFOV
Scaling Factor applied on LR pipe
VPB0 resolution (LR pipe output)
ISP Crop Window
ROI for DZ = 1
Sensor Output
Same aspect Ratio
ISP8500 Input
VPB1 resolution (HR pipe output)
X2 DZ (crop factor)
Scaling Factor applied on pipe #0

Figure 34. x2 DZ featuring sensor pre-scaling, pan-tilt, ISP crop and ISP scaling

The center of the FOV is indicated through the OMX_Symbian_IndexConfigCenterFieldOfView/OMX_CONFIG_POINTTYPE index (relative to the sensor FFOV). This is part of the extensions SHAI 1.2.

The coordinates of the center of the field of view will use a Q16.15 representation, with –(1<<16,1<<16) being top left, (0,0) the center, and 1<<16,1<<16 being bottom right.

In that case the largest ROI centered on this point acts as the ROI for DZ = 1. Then the ISP FW is able to digizoom inside this ROI to serve camera active output ports. The A:R that characterizes this max ROI is the one of the port that serves the main use case (indicated by bOneShot) if active.

11.5 Rotation

The camera supports OMX_IndexConfigCommonRotate in Khronos 1.1.2 /11/ configurations, only on VPB2 port along with 420MB format. Supported angles are 0°, 90°, 180° and 270°.

The IL client is responsible for setting the port definition parameter accordingly: if after the rotation, the image is portrait, the port definition parameter settings must be portrait.

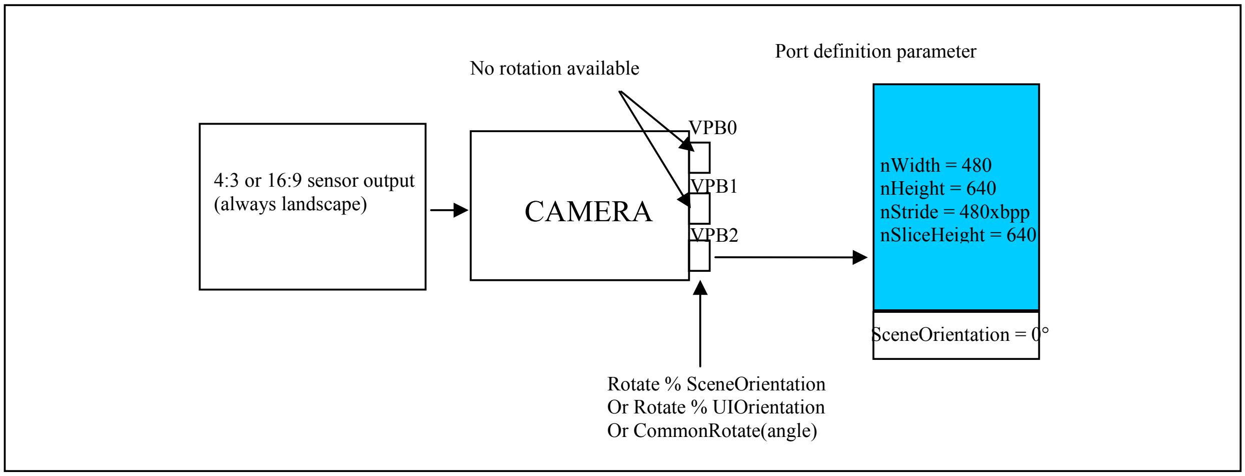


Figure 35. Rotation of the port definition parameter

Indeed, assuming the rotation leads to a portrait image and the port definition parameter settings are programmed with width/height/stride corresponding to a landscape, the camera component cannot guess whether this is wished or it is a configuration error (some ICs might be able to scale with distortion after the rotation block).

In case a rotation happens when port are active, a port settings changed event shall be emitted to the IL client so that it can reconfigure the whole stream.

The relative orientation of the sensor with respect to the scene (horizon) is given by the IL client to the camera OMX component through index OMX_Symbian_IndexConfigOrientationScene / OMX_Symbian_CONFIG_ORIENTATIONTYPE).

Usually this information comes from the application that uses information from an accelerometer. The scene is what the eyes see. In other words, if the sensor reads the sensor orthogonaly to the scene, the associated image or video to be encoded, stored and read again by the end-user will have a wrong orientation. For end-user convenience, it is good to encode and store the image as the end-user saw it.

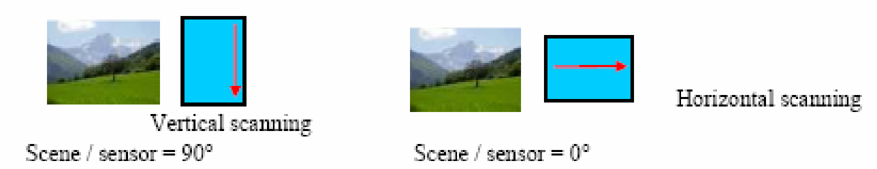


Figure 36. Relative orientation of the sensor wrt scene/horizon

Above information is embedded in the extradata inside the buffers. This allows all downstream components to have the orientation information and behave accordingly. If at any time the orientation is changed, the update inside the extradata of the buffer should be made.

11.6 Mirroring

IL Client can set mirroring through OMX_IndexConfigCommonMirror defined in Khronos 1.1.2 /11/ . None, Vertical, Horizontal and combination or vertical and horizontal is supported.

11.7 Digital Stabilization

Video and still stabilization are abstracted by the camera OMX component. They are turned on/off through the OMX_IndexConfigCommonFrameStabilization/OMX_CONFIG_FRAMESTABTYPE Boolean API defined Khronos 1.1.2 /11/ .

Still and video stabilization are discriminated through the bOneShot parameter. Still stabilization only applies to VPB1 output. Video stabilization applies to VPB0 and VPB2 outputs.

There is currently no way to further configure the stabilization algorithm (complexity, algo, …). Stabilization only can be activated if bOneShot is set to 0.

11.8 Effects

The primary camera supports the following **color effects** (also known as **color tones**): normal (off), sepia, negative, natural, vivid, B&W, solarize.

Color effects are set by the IL client through OMX_IndexConfigCommonImageFilter / OMX_IMAGEFILTERTYPE index.
The enum values originally available in /11/ were extended in 1.Y extension discussions.

There is no need anymore to support OMX_IndexConfigCommonColorEnhancement for grayscale or sepia.

OMX_IndexConfigCommonImageFilter OMX_CONFIG_IMAGEFILTERTYPE	Description	Khronos	Extensions
OMX_ImageFilterNone	OFF/None	1.1.2	/
OMX_ImageFilterNegative	Negative. Natively supported by the ISP CE special effect module	1.1.2	/
OMX_Symbian_ImageFilterSepia	Sepia. - Sepia produces an aged effect to photos - color matrix is set to produce a monochrome image; gamma	/	SHAI 1.2

	channels are tuned to tint the image a yellow-brown		
OMX_Symbian_ImageFilterGrayscale	Monotone (Gray) or Black and White. Saturation set to 0%	/	SHAI 1.2
OMX_Symbian_ImageFilterNatural	Saturation set to 100%	/	SHAI 1.2
OMX_Symbian_ImageFilterVivid	Saturation set at 120 %	/	SHAI 1.2
OMX_ImageFilterSolarize	Solarization	1.1.2	/
OMX_ImageFilterWatercolor	Water Color	1.1.2 Imaging Extension	/
OMX_ImageFilterPastel	Pastel	1.1.2 Imaging Extension	/
OMX_ImageFilterFilm	Film	1.1.2 Imaging Extension	/
OMX_STE_ImageFilterGrayscaleNegative	Monotone Negative (Gray)	/	ST-Ericsson Vendor Extensions

Table 23 Supported Color Tones

Finally, explicit control of saturation is used to implement Vivid(120%), natural (80%) and black & white (0%).

Those indexes can be changed during still/video preview and video record. These settings can be applied independently on output ports VPB0 and VPB2.

Color effect implementation relies on the following colour engine HW stages: RGB matrix, effect and gamma correction. In still, the color engine is not entered in the camera component. As such effect implementation is delegated to the downstream ISP proc.

11.9 Saturation

Manual saturation is exposed through OMX_IndexConfigCommonSaturation / OMX_CONFIG_SATURATIONTYPE.nSaturation [-100, +100].

This is defined in /11/

0x0 means no saturation change to pixel data. -100 produces all black pixels, 100 produces all white pixels.

Implementation uses the color engine HW. It can be applied independently on VPB0 and VPB2. The setting does not apply to VPB1 since in still the color engine is not entered.

In Still saturation is implemented in the ISP proc, independently programmable on both output ports.

11.10 Contrast

Both camera support the OMX_IndexConfigCommonContrast/OMX_CONFIG_CONTRASTTYPE [-100, +100].

This is defined in /11/

The value represents an offset value with respect to the value automatically computed by the camera component.

It can be applied independently on VPB0 and VPB2. The setting does not apply to VPB1 since in still the color engine is not entered.

Contrast setting can be changed while the pixel pipe is streaming (typically in preview view).

Implementation uses the color engine HW (within the [0, 200%] range).

11.11 Brightness

Both cameras support OMX_IndexConfigCommonBrightness / OMX_CONFIG_BRIGHTNESSTYPE.nBrightness [0%, +100%]

This is defined in /11/

0% all black pixels. 100% all white pixels.

It can be applied independently on VPB0 and VPB2. The setting does not apply to VPB1 since in still the color engine is not entered.

Implementation uses the color engine HW (within the [0, 200%] range).

11.12 Sharpening

Sharpness setting is controlled through OMX_Symbian_IndexConfigSharpness where the actual sharpness value is OMX_Symbian_CONFIG_S32TYPE.

Sharpness is implemented in the colour engine. As such this settings is not applied when outputting a raw bayer frame.

This setting is applicable only to output ports. Independent settings can be applied at output ports.

11.13 YUV Range

The YUV range is controlled through the OMX extension introduced in: OMX_Symbian_IndexParamColorPrimary
/ OMX_Symbian_PARAM_COLORPRIMARY. Full range, BT601 and BT709 are supported.

OMX_Symbian_PARAM_COLORPRIMARYTYPE in OMX_Symbian_PARAM_COLORPRIMARY	Description	Khronos	Extensions
OMX_ Symbian_ColorPrimaryFullRange	YUV range from 0 to 255	/	SHAI 1.2
OMX_ Symbian _ColorPrimaryBT601	YUV range	/	SHAI 1.2

	from 16 to 235 or 240 for chrominance with BT.601 defined YUV to RGB color conversion matrix		
OMX_ Symbian _ColorPrimaryBT709	YUV range from 16 to 235 or 240 for chrominance with BT.709 defined YUV to RGB color conversion matrix	/	SHAI 1.2

Table 24 YUV Range Values

The setting is not applied on camera side when outputting raw bayer. The setting is only available at VPB2 port.

In still/video preview modes and still capture operating mode, the full YUV range is supposed to be requested to the OMX camera. The colour engine OutputCoderControls.TransformType then is programmed with TransformType_YCbCr_JFIF. This leads to luma and chroma ranges within [0, 255].

JPEG encoder accommodates 0x0 and 0xFF YUV values.

In video mode however, the YUV range is likely to programed as full range or BT601, depending on the codec. BT601 range is defined as followed:

- Y within the range of [16; 235] and
- chroma within the range of [16; 240].

As opposed to H263, MPEG4 and H264 stream has syntax fields that encode the actual YUV range being used.

H263 encoder only manages BT601 range.

When BT601 is applied, the OMX camera programs the colour engine accordingly: ColourEngine0_OutputCoderControls.TransformType = TransformType_YCbCr_Rec601.

11.14 Exposure Presets

The camera supports the exposure presets through OMX_CONFIG_EXPOSURECONTROLTYPE. eExposureControl defined in Khronos /11/

OMX_EXPOSURECONTROLTYPE	Description	Khronos	Extensions
OMX_ExposureControlOff	AEC is disabled	1.1.2	/
OMX_ExposureControlAuto	AEC is enabled	1.1.2	/

OMX_ExposureControlNight	Long exposure time is used. Still preview Framerate should be within a lower range than normal (around 5 fps). Still the framerate control is auto.	1.1.2	/
OMX_ExposureControlBackLight	Backlight setting for bright backgrounds.	1.1.2	/
OMX_ExposureControlSports	Short exposure time	1.1.2	/
OMX_ExposureControlSpotlight	Used to ignore the surroundings of the objects. AEC is done on a relatively small area around the center (5% of the image resolution)	1.1.2	/
OMX_ExposureControlSnow	Snow setting for daylight exposure. (+1.33 EV compensation)	1.1.2	/
OMX_ExposureControlBeach	Beach setting for daylight exposure with reflective glare. (+1 EV compensation)	1.1.2	/
OMX_ExposureControlLargeAperture		1.1.2	/
OMX_ExposureControlSmallApperture		1.1.2	/
OMX_ExposureControlFacePriorityMode	Exposure control priority on face area. The exposure of the captured image or video stream is automatically adjusted using a face reference from within each frame.	1.1.2 Imaging Extensions	/
OMX_Symbian_EXPOSURECONTROLTYPE	Description	Khronos	Extensions
OMX_Symbian_ExposureControlCenter	Center	/	SHAI 1.2
OMX_Symbian_ExposureControlVeryLong	Very Long	/	SHAI 1.2
OMX_Symbian_ExposureControlHwFunctionalTesting	HW testing	/	SHAI 1.2

Table 25 OMX Exposure Presets

Other exposure settings are supported through OMX_CONFIG_EXPOSUREVALUETYPE as specified in Khronos 1.1.2. Shutter speed, sensitivity, Metering mode and aperature can be adjusted through this API to achieve optimal exposure of a scene.

Those exposure modes apply to still image capture and video use cases.

Exposure preset setting is an OMX configuration and can be changed while the pixel pipe is streaming (during still/video preview, during video record). It can also be changed in non active states (idle, paused, …).

However it is not supposed to be changed by the IL client once a still capture has started (within a burst).

This index is supported at all ports.

11.15 Exposure Compensation

EV compensation ([-2; +2] with 1/3 increments) is supported by the primary camera only. It can be changed during preview only – still and video. When EV bracketing is set, it might be changed after each still shot by the OMX Camera (see next chapter).

The OMX index used for this is OMX_IndexConfigCommonExposureValue/
OMX_CONFIG_EXPOSUREVALUETYPE.xEVCompensation (Q16 format). This API is defined in /11/ .

OMX_CONFIG_ EXPOSUREVALUETYPE	Description	Khronos	Extensions
.xEVCompensation	EV Value	1.1.2	/

Table 26 Configuring EV Values

This index is supported at all ports.

11.16 Bracketing

The OMX camera supports exposure (EV) bracketing enabling/disabling. Bracketting only makes sense in still mode (single or burst).

Bracket Merge, where a single image is eventually produced, is not supported by the camera component.

Bracketting is enabled through settings the following bool inside OMX_Symbian_CONFIG_EXTCAPTUREMODETYPE. bEnableBracketing with OMX_Symbian_IndexConfigExtCaptureMode index.

Bracketing parameters can be supported using OMX_Symbian_CONFIG_BRACKETINGTYPE. Today only EV bracketing is supported.

Bracket steps are +/- 0.5 EV.

Bracketing is implemented at OMX level which provides changed EV value to the STE3A library.

OMX_Symbian_ CONFIG_EXTCAPTUREMODETYPE	Description	Khronos	Extensions
. bEnableBracketing	Bool to enable or disable bracketing	/	SHAI 1.2

Table 27 To enable bracketing

OMX_Symbian_ CONFIG_BRACKETINGTYPE	Description	Khronos	Extensions
.eBracketMode (OMX_Symbian_BRACKETMODETYPE)	Type of bracketing	/	SHAI 1.2
nNbrBracketingValues	Number of bracketing values	/	SHAI 1.2
nBracketValues[5]	Bracketing values in an array. Q16 format.	/	SHAI 1.2

Table 28 Bracketing Configuration

OMX_Symbian_ BRACKETMODETYPE	Description	Khronos	Extensions
OMX_Symbian_BracketExposureRelativeInEV (only this is supported today)	Exposure value is changed relative to the value set by automatic exposure. Increment is additive.	/	SHAI 1.2

Table 29 Type of bracketing

These indexes are supported only at VPB1.

11.17 Metering Lock

The OMX camera supports individual lock of metering algorithms (AEC, AWB, AF) as well as lock of all 3A together.

Metering lock is supported by the OMX camera component through the indexes below. Separate indexes are used to separately lock the 3A algorithms.

OMX_Symbian_CONFIG_LOCKTYPE	Description	Khronos	Extensions
OMX_Symbian_IndexConfigCommonExposureLock	Only AEC is locked	/	SHAI 1.2
OMX_Symbian_IndexConfigCommonWhiteBalanceLock	Only White Balance is locked	/	SHAI 1.2
OMX_Symbian_IndexConfigCommonFocusLock	Only AF is locked	/	SHAI 1.2
OMX_Symbian_IndexConfigCommonAllLock	All 3A are locked	/	SHAI 1.2

Table 30 Indexes for locking 3A

OMX_Symbian_CONFIG_LOCKTYPE	Description	Khronos	Extensions
eImageLock	Lock Setting	/	SHAI 1.2

Table 31 Lock Parameters

Lock setting is controlled by OMX_IMAGE_CONFIG_LOCKTYPE.eImageLock to 1/0. When OMX_SYMBIAN_LockImmediate the lock occurs immediately (actually at the end of the current iteration), when OMX_SYMBIAN_LockAtCapture the lock is applied from the next capture on (intent is to freeze the algos after the first shot of a burst).

Lock setting is port independent and impacts all output ports.

The capability to lock 3A on the first shot of a sequence of stills also is an enabler for panorama stiching.

Lock is completely implemented inside Omx Camera and no interaction with the STE 3A is done.

If the lock occurs during the execution of an outstanding above call, this computation ends normally.

11.18 Metering Mode

Metering preset is supported by the primary camera only (OMX_IndexConfigCommonExposureValue/ OMX_CONFIG_EXPOSUREVALUETYPE .eMetering). This is defined in /11/

This metering information is passed to STE 3A.

It can be changed during still preview only.

Auto Exposure Metering Mode OMX_METERINGTYPE in OMX_CONFIG_EXPOSUREVALUETYPE OMX_IndexConfigCommonExposureValue	Description	Khronos	Extensions
OMX_MeteringModeAverage	Frame-Average	1.1.2	/
OMX_MeteringModeSpot	Spot Metering	1.1.2	/
OMX_MeteringModeMatrix	Evaluative	1.1.2	/
OMX_STE_MeteringModeCenterWeighted	Center Weighted	/	ST-Ericsson Vendor Extensions
OMX_STE_MeteringModeBeachAndSnow	Beach and Snow	/	ST-Ericsson Vendor Extensions
OMX_STE_MeteringModeScenery	Scenery	/	ST-Ericsson Vendor Extensions
OMX_STE_MeteringModePortrait	Portrait	/	ST-Ericsson

			Vendor Extensions
OMX_STE_FacePriority	Face Priority	/	ST-Ericsson Vendor Extensions

Table 32 Metering Mode

11.19 ISO Settings

The feature only is supported by the primary camera. It is applied during still preview. Automatic selection and 100, 200, 400, 800, 1600, 3200 are supported.

When automatic selection is activated, the actual used value must be returned in the EXIF data.

ISO settings are supported by the camera component through the OMX_IndexConfigCommonExposureValue/ OMX_CONFIG_EXPOSUREVALUETYPE index in /11/

The ISO can be manually set through the nSensitivity field whereas automatic choice can be made by setting bAutoSensitivity to true.

ISO Setting is passed to STE 3A.

Actual ISO value applied during the capture and eventually put in the associated EXIF tag can be derived from the analog gain as 100*analog gain. Otherwise it is determined as SensorSensitiviy x analog gain.

11.20 Flicker Cancellation

This feature only is supported by the primary camera, for still/video preview and video capture through the OMX_Symbian_CONFIG_FLICKERREMOVALTYPE.eFlickerRejection.

eFlickerRejection has enum values as described below. OMX_Symbian_IndexConfigFlickerRemoval index is used.

OMX_Symbian_FLICKERREMOVALTYPE in OMX_Symbian_CONFIG_FLICKERREMOVALTYPE	Description	Khronos	Extensions
OMX_Symbian_FlickerRemovalOff	Off	/	SHAI 1.2
OMX_Symbian_FlickerRemoval50	50 Hz	/	SHAI 1.2
OMX_Symbian_FlickerRemoval60	60 Hz	/	SHAI 1.2

Table 33 Flicker Removal types

The ISP does not contain any HW/FW for automatic flicker frequency detection. Anyway this HW would require the sensor to have a "super pixel" which is not foreseen.

50/60 Hz manual anti-flicker is implemented by STE 3A.

For Auto Flicker Cancellation in SW today country codes are used to determine the flicker frequency. This mapping needs to be implemented inside the IL Client.

OMX_IMAGE_CONFIG_FLICKERREJECTIONTYPE defined in /12/ is not used.

This index is applicable on all ports.

11.21 White Balance Presets

The OMX_IndexConfigCommonWhiteBalance index (OMX_CONFIG_WHITEBALCONTROLTYPE) enables to select the white-balance preset in /12/ . The OMX cameras bring the following support.

OMX_IndexConfigCommon WhiteBalance OMX_CONFIG_ WHITEBALCONTROLTYPE	Description	Khronos	Extensions
OMX_WhiteBalControlOff	N/A	1.1.2	/
OMX_WhiteBalControlAuto	Auto. The color temperature of the captured image or video stream is adjusted per frame using a white reference from within each frame.	1.1.2	/
OMX_WhiteBalControlSunLight	DayLight. Manual WB when the sun provides the light source.	1.1.2	/
OMX_WhiteBalControlCloudy	Cloudy. Manual WB when the sun provides the light source through clouds.	1.1.2	/
OMX_WhiteBalControlTungsten	Manual WB when the light source is tungsten.	1.1.2	/
OMX_WhiteBalControlFluorescent	Manual WB when the light source is Fluorescent	1.1.2	/
OMX_ WhiteBalControlIncandescent	Manual WB when the light source is Incandescent	1.1.2	/
OMX_WhiteBalControlShade	Manual WB when the sun provides the light source and scene is in shade	1.1.2	/
OMX_WhiteBalControlCustom	Associated with a functional event (OMX_FunctionalWhiteBalComputed)	1.1.2	/
OMX_WhiteBalControlHorizon	Manual WB when sun is in horizon	1.1.2	/
OMX_WhiteBalControl FacePriorityMode	White balance control priority on face area. The color temperature of the captured image or video stream is adjusted per frame using a face reference from within each frame.	1.1.2 Imaging Extensions	/
OMX_WhiteBalControlFlash	Manual WB when light source is flash	1.1.2	/

Table 34 WB Presets

Technically there is no issue behind supporting the same presets for the primary and secondary camera except calibration effort.

WB preset setting can be changed while the pixel pipe is streaming, ie. during preview, video record. However it is not supposed to be changed by the IL client once a still capture has started.

This setting is applicable for all ports.

11.22 Scene Modes

At OMX level scene modes can be configured through ConfigSceneMode index.

OMX SCENE MODE OMX_Symbian_IndexConfigSceneMode	OMX Config	Khronos	Extensions

OMX_Symbian_SCENEMODETYPE eSceneType in OMX_Symbian_CONFIG_SCENEMODETYPE			
Auto	OMX_Symbian_SceneAuto	/	SHAI 1.2
Portrait	OMX_Symbian_Scene Portrait	/	SHAI 1.2
Landscape	OMX_Symbian_Scene Landscape	/	SHAI 1.2
Night	OMX_Symbian_Scene Night	/	SHAI 1.2
Night portrait/Party	OMX_Symbian_Scene NightPortrait	/	SHAI 1.2
Sport	OMX_Symbian_Scene Sport	/	SHAI 1.2
Macro	OMX_Symbian_Scene Macro	/	SHAI 1.2
Document	OMX_STE_Scene Document	/	ST-Ericsson Vendor Extensions
Beach	OMX_STE_Scene Beach	/	ST-Ericsson Vendor Extensions
Snow	OMX_STE_Scene Snow	/	ST-Ericsson Vendor Extensions

Table 35 Scene Modes

The scene mode settings would be received by the OMX Camera and sent to the STE 3A. For conformance with other user mode configs, this will be done at the IL client level. So the IL client will ensure that the user configurations and scene mode setting are compliant with each other.

Hence the IL client would use customer specific mappings to set the user configs in compliance with the scene mode settings.

These settings are applicable to all ports.

11.23 Histograms

256-bin RGB Histograms can be generated by the OMX camera.

Generation is activated through the OMX_Symbian_IndexConfigHistogramControl.bMeasure flag.

OMX_Symbian_CONFIG_HISTOGRAMCONTROLTYPE is used for the configuration structure for histogram control.

OMX_Symbian_IndexConfigHistogramControl OMX_Symbian_CONFIG_HISTOGRAMCONTROLTYPE	Description	Khronos	Extensions
bMeasure	On/off histogram measurement	/	SHAI 1.2
nBins	Number of bins. 0 means maximum bins to be used (256)	/	SHAI 1.2

nBytesPerBin	Number of bytes per bin. OMX_ErrorUnsupportedSetting is returned if not supported.	/	SHAI 1.2

Table 36 Histogram configuration

Only 256 bin histograms are supported. Histogram availability is reported through OMX functional events.

The IL client shall register to the OMX_IndexConfigRGBHistogram index using OMX_IndexConfigRequestCallback. This callback would be fired whenever a new histogram is collected. The callback would have data in the following configuration structure.

OMX_Symbian_ CONFIG_RGBHISTOGRAM	Description	Khronos	Extensions
nRed[256]	Red histogram	/	SHAI 1.2
nGreen[256]	Green histogram	/	SHAI 1.2
nBlue[256]	Blue histogram	/	SHAI 1.2

Table 37 Returned histogram statistics

If histogram is enabled, the histogram RGB data are also available inside the extradata of the Omx camera buffers. These will be part of the video-pack extradata for VPB0 and VPB2 ports.

11.24 Pre-Capture Warning

The IL client is allowed to register a functional callback (OMX_IndexConfigRequestCallback) on the OMX_Symbian_IndexConfigPreCaptureExposureTime index. To receive pre-capture warning.

The following information is provided in the callback.

OMX_Symbian_CONFIG_ PRECAPTUREEXPOSURETIMETYPE. eExposureTime	Description	Khronos	Extensions
OMX_Symbian_ PreCaptureExposureNoneOrOngoing	Ongoing	/	SHAI 1.2
OMX_Symbian_PreCaptureExposureNormal	Normal	/	SHAI 1.2
OMX_Symbian_PreCaptureExposureShort	Short	/	SHAI 1.2
OMX_Symbian_PreCaptureExposureLong	Long	/	SHAI 1.2

Table 38 Pre-Capture Warnings

The notification is sent at the end of the One-Shot focus. It is not expected to be used in video mode. Hence this is only supported at VPB1 port of the Omx Camera.

11.25 Exposure Initiated

The IL client is allowed to register a functional callback (OMX_IndexConfigRequestCallback) on the OMX_Symbian_IndexConfigExposureInitiated index.

To receive an indication of when an image was exposed. The type of information exchanged in this callback is OMX_Symbian_CONFIG_BOOLEANTYPE defined in extensions SHAI 1.2.
When the image is exposed the Omx Camera sets bEnabled to true. And the IL Client sets bEnabled to false in the callback.

11.26 Providing Sensor Information

The IL Client can use the index OMX_Symbian_IndexConfigCameraSensorInfo/ OMX_Symbian_CONFIG_HWINFOTYPE to provide information of the Camera sensor.

OMX_Symbian_CONFIG_HWINFOTYPE	Description	Khronos	Extensions
nVersion1	Bits 0...7 Sensor Manufacturer (8 bits). Bits 8...15 Sensor Version/Revision (8 bit). Bits 16...31 Sensor Model (16 bits).	/	SHAI 1.2
nVersion2	empty	/	SHAI 1.2
cInfoString	Some string information	/	SHAI 1.2

Table 39 Camera Sensor Information

11.27 Hints

The camera OMX component supports the OMX_Symbian_IndexConfigHintPowerVsQuality hint control. This is only an indication to the Omx Camera and implementation can be defined based on internal strategy.

Today the working assumption is to use this index as a hint to influence the choice of the sensor mode. If power is priviledged, binning mode should be chosen in preview/video whereas if quality is priviledged, full resolution streaming would be influenced.

OMX_Symbian_ IndexConfigHintPowerVsQuality OMX_Symbian_ CONFIG_HINTPOWERVSQUALITYTYPE	Description	Khronos	Extensions
OMX_Symbian_QualityNotSpecified	No Indication	/	SHAI 1.2
OMX_Symbian_QualityLow	Low Quality	/	SHAI 1.2
OMX_Symbian_QualityBalanced	Medium Quality	/	SHAI 1.2
OMX_Symbian_QualityHigh	High Quality	/	SHAI 1.2

Table 40 Hint of Power vs Quality

11.28 Depth of Field

OMX_Symbian_IndexConfigHintDepthOfField is meaningful with EDOF sensors.

OMX_Symbian_CONFIG_HINTDOFTYPE. eDoFHint is used for the configuration which has the following potential values.

OMX_Symbian_IndexConfigHintPowerVsQuality OMX_Symbian_CONFIG_HINTPOWERVSQUALITYTYPE	Description	Khronos	Extensions
OMX_Symbian_DoFNotSpecified	Not specified, component decides by itself	/	SHAI 1.2
OMX_Symbian_DoFSmall	Small	/	SHAI 1.2
OMX_Symbian_DoFMedium	Medium	/	SHAI 1.2
OMX_Symbian_DoFLarge	Large	/	SHAI 1.2

Table 41 DOF Values

11.29 Flash Control

The index OMX_Symbian_IndexConfigFlashControl / OMX_Symbian_CONFIG_FLASHCONTROLTYPE. eFlashControl allows to select the type of flash control to be done at Omx level.

OMX_Symbian_IndexConfigCommonFlashControl OMX_IMAGE_FLASHCONTROLTYPE in OMX_Symbian_CONFIG_FLASHCONTROLTYPE	Description	Khronos	Extensions
OMX_IMAGE_FlashControlOn	On/Force	1.1.2	/
OMX_IMAGE_FlashControlOff	Off	1.1.2	/
OMX_IMAGE_FlashControlAuto	Auto	1.1.2	/
OMX_IMAGE_FlashControlRedEyeReduction	Red-Eye	1.1.2	/
OMX_IMAGE_FlashControlFillin	Fill In	1.1.2	/
OMX_IMAGE_FlashControlTorch	Torch	1.1.2	/
OMX_SYMBIAN_IMAGE_FlashControlSlowFrontSync	Slow Front Sync	/	SHAI 1.2
OMX_SYMBIAN_IMAGE_FlashControlSlowRearSync	Slow Rear Sync	/	SHAI 1.2
OMX_SYMBIAN_IMAGE_FlashControlOnTest	Flash On Test	/	SHAI 1.2
OMX_SYMBIAN_IMAGE_FlashControlTorchOnTest	Torch on Test	/	SHAI 1.2
OMX_SYMBIAN_IMAGE_FlashControlIndicatorLightOnTest	Indicator Light on Test	/	SHAI 1.2

Table 42 Flash Modes supported

11.30 AF Indicator

The index OMX_Symbian_IndexConfigAFAssistantLight

/ OMX_Symbian_CONFIG_HIGHLEVELCONTROLTYPE. eControl allows to select the Af control type at Omx level.

OMX_Symbian_IndexConfigAFAssistantLight	Description	Khronos	Extensions
OMX_Symbian_ControlOff	On/Force	/	SHAI 1.2
OMX_Symbian_ControlOn	Off	/	SHAI 1.2
OMX_Symbian_ControlAuto	Auto	/	SHAI 1.2

Table 43 AF Indicator control modes supported

11.31 Video Light Flash Mode

The index OMX_Symbian_IndexConfigVideoLight
/ OMX_Symbian_HIGHLEVELCONTROLTYPE. eControl allows to select the Video flash mode control type at Omx level.

OMX_Symbian_IndexConfigVideoLight	Description	Khronos	Extensions
OMX_Symbian_ControlOff	On/Force	/	SHAI 1.2
OMX_Symbian_ControlOn	Off	/	SHAI 1.2
OMX_Symbian_ControlAuto	Auto	/	SHAI 1.2

Table 44 Video Flash control modes supported

11.32 Xenon Life Counter

OMX_Symbian_IndexConfigXenonLifeCounter. nNumber can be used to get the xenon life counter from the Omx Camera. This is part of extensions SHAI 1.2. This is not supported today as Xenon Flash is not supported.

11.33 Xenon Flash status

OMX_Symbian_IndexConfigXenonFlashStatus/
OMX_Symbian_CONFIG_XENONFLASHSTATUSTYPE can be used for fetching the Xenon flash status. It is also possible to receive this information in a functional callback which can be registered for this index using OMX_IndexConfigCallbackRequest. This might also be used as a pre-capture warning.

OMX_Symbian_XENONFLASHSTATUSTYPE in OMX_Symbian_CONFIG_XENONFLASHSTATUSTYPE	Description	Khronos	Extensions
OMX_Symbian_XenonFlashNone	Status is not available	/	SHAI 1.2
OMX_Symbian_XenonFlashDischarged	Discharged	/	SHAI 1.2
OMX_Symbian_XenonFlashCharging	Charged	/	SHAI 1.2
OMX_Symbian_XenonFlashReady	Ready	/	SHAI 1.2
OMX_Symbian_XenonFlashNotAvailable	Flash is not available at this moment	/	SHAI 1.2

Table 45 Xenon Flash status information

11.34 Providing FlashGun Information

The IL Client can use the index OMX_Symbian_IndexConfigFlashGunInfo
/ OMX_Symbian_CONFIG_HWINFOTYPE
to provide information of the flash gun device.

OMX_Symbian_CONFIG_HWINFOTYPE	Description	Khronos	Extensions
nVersion1	Bits 0-7 Flash IC Info Bits 8-15 Flash IC revision Bits 16-17 Flash Module Info A as present in Module Info Reg.A of flash HW (Only valid for some Xenon flash, leave as 0 for LED and even for Xenon if not available) Bits 24-31 Flash Module Info B as present in Module Info Reg.B of flash HW (Only valid for some Xenon flash, leave as 0 for LED and even for Xenon if not available)	/	SHAI 1.2
nVersion2	filled in same way as nVersion1 but containing information for secondary flash IC connected to same camera. Leave as 0s if there is only single flash IC	/	SHAI 1.2
cInfoString	Some string information	/	SHAI 1.2

Table 46 Camera Sensor Information

11.35 Focus Modes

The index OMX_IndexConfigFocusControl/ OMX_IMAGE_CONFIG_FOCUSCONTROLTYPE allows to select the type of focus to be done at Omx level.

Focus Mode OMX_IndexConfigFocusControl OMX_IMAGE_CONFIG_FOCUSCONTROLTYPE	Description	Khronos	Extensions
OMX_IMAGE_FocusControlOn	Normal (Single) AF Mode. this is the single focusing triggered by the half press of capture button	1.1.2	/
OMX_IMAGE_FocusControlOff	N/A (Manual Focus)	1.1.2	/
OMX_IMAGE_FocusControlAuto	Continuous AF Mode	1.1.2	/
OMX_Symbian_FocusControlIdle	Infinity Position (Rest Position, Default). It is used at initialization when focus is not started	/	SHAI 1.2

Table 47 Focus Modes supported

11.36 Focus Range

The index OMX_Symbian_IndexConfigFocusRange/ OMX_Symbian_CONFIG_FOCUSRANGETYPE allows to configure the focus range of the targeted object at Omx level.

Focus Range OMX_Symbian_IndexConfigCommonFocusRange OMX_Symbian_CONFIG_FOCUSRANGETYPE	Description	Khronos	Extensions
OMX_Symbian_FocusRangeAuto	Auto / Normal	/	SHAI 1.2
OMX_Symbian_FocusRangeHyperfocal	Hyperfocal	/	SHAI 1.2
OMX_Symbian_FocusRangeSuperMacro	SuperMacro	/	SHAI 1.2
OMX_Symbian_FocusRangeMacro	Macro	/	SHAI 1.2
OMX_Symbian_FocusRangeInfinity	OFF / Infinity	/	SHAI 1.2

Table 48 Focus Ranges supported

11.37 Focus Region Control

The IL client can control the focus regions to be used by the Omx Camera component using the OMX_Symbian_IndexConfigFocusRegion / OMX_Symbian_CONFIG_FOCUSREGIONTYPE. There are two structures inside OMX_Symbian_CONFIG_FOCUSREGIONTYPE.

eFocusRegionControl (OMX_Symbian_FOCUSREGIONCONTROL) allows the focus region selection.

sFocusRegion (OMX_Symbian_RELATIVERECTTYPE) provides more information of the Focus region if focus region control is manual.

Focus Region should be updated by the Omx Camera incase of Auto, Face priority and Object Priority controls.

OMX_Symbian_FOCUSREGIONCONTROL in OMX_Symbian_CONFIG_FOCUSREGIONTYPE	Description	Khronos	Extensions
OMX_Symbian_FocusRegionAuto	Focus region decided by AF algorithm	/	SHAI 1.2
OMX_Symbian_FocusRegionManual	Manual focus region selected by user	/	SHAI 1.2
OMX_Symbian_FocusRegionFacePriority	ROI with priority face (if available) should be used as focus region, otherwise automatically selected by AF algorithm	/	SHAI 1.2
OMX_Symbian_FocusRegionObjectPriority	ROI with priority object (if available) should be used as focus region, otherwise automatically selected by AF algorithm	/	SHAI 1.2

Table 49 Focus Regions supported

11.38 Focus Status

The IL client is allowed to register a functional callback (OMX_IndexConfigRequestCallback) on the OMX_Symbian_IndexConfigExtFocusStatus index.

This to receive the status of focus once a focus operation is initiated.

OMX_SYMBIAN_AFROITYPE provides information about all the areas.

Inside this with other information, OMX_FOCUSSTATUSTYPE eFocusStatus provides the real focus status which is defined in Khronos and has the following values:

OMX_FOCUSSTATUSTYPE eFocusStatus in OMX_Symbian_ CONFIG_EXTFOCUSSTATUSTYPE	Description	Khronos	Extensions
OMX_FocusStatusOff	Focus request is disabled	1.1.2	/
OMX_FocusStatusRequest	Focus request is currently being processed	1.1.2	/
OMX_FocusStatusReached	Focus has been reached	1.1.2	/
OMX_FocusStatusUnableToReach	Focus is unreachable, the maximum is too close to the average noise	1.1.2	/
OMX_FocusStatusLost	Focus has been lost, the main subject has moved in the scene	1.1.2	/

Table 50 Focus Status supported

11.39 Region of Interest Information

OMX_Symbian_IndexConfigROI/ OMX_Symbian_CONFIG_ROITYPE is used to provide region of interest information.

nNumberOfROIs indicate the number of regions. There could be mixed regions. sROIs which is of type OMX_Symbian_ROIOBJECTINFOTYPE provides an array of ROIs and their associated information.

OMX_Symbian_ROIOBJECTINFOTYPE is composed of the following fields

OMX_Symbian_ ROIOBJECTINFOTYPE in OMX_Symbian_ ROIOBJECTINFOTYPE in OMX_Symbian_ CONFIG_ROITYPE	Description	Khronos	Extensions
nROIID	ID of the region, 0 by default	/	SHAI 1.2
OMX_Symbian_ RELATIVERECTTYPE	Relative rectangle information of the object.	/	SHAI 1.2
nPriority	Priority of the region, 0 being the highest	/	SHAI 1.2
OMX_Symbian_ ROIOBJECTTYPE	Type of object (OMX_Symbian_RoiObjectNone, OMX_Symbian_RoiObjectTypeFace, OMX_Symbian_RoiObjectTypeObject)	/	SHAI 1.2
OMX_Symbian_ 3DORIENTATIONYTYPE	3D orientation of object nYaw nPitch nRoll	/	SHAI 1.2

Table 51 ROI support

ROIs are sent by the Omx Camera to the STE 3A. The IL Client fetches the ROI information from IV Analyzers and provide this information to the Omx Camera using SetConfig. These can be used as focus regions or for AWB.

Component implementation must maintain separate list of ROIs for all different types of objects. When IL client does a GetConfig it specifies in the eObjectType which type of region's information is required.

ROI information is also added to the extradata inside the buffers.

12 OMX ISP Processor

This OMX component is a non standard component. It is abbreviated ISP proc. In concept it is derived from the standard image/video processor (IV proc). Hence it is under the Extended IV Processor Category.

However, whereas an OMX standard iv proc is supposed to perform basic video/image processing (like rotation, cropping, colour conversion), this ISP processor performs image reconstruction (raw to YUV/RGB domain conversion). It uses the ISP HW pipe in a memory to memory manner whereas the camera uses the pipe in a sensor to memory manner. The ISP processor also is able to perform some of the standards IV proc operations (crop, rotation, scaling) with restrictions.

The provision for such a component aims at being able to add/replace processings in raw bayer or YUV domain by other processings. Also having an intermediate raw bayer buffer for implementing DZ enables to relax the constraints on the sensor line length (since the DZ is not performed anymore on-the-fly).

The ISP proc only is used in still capture stream.

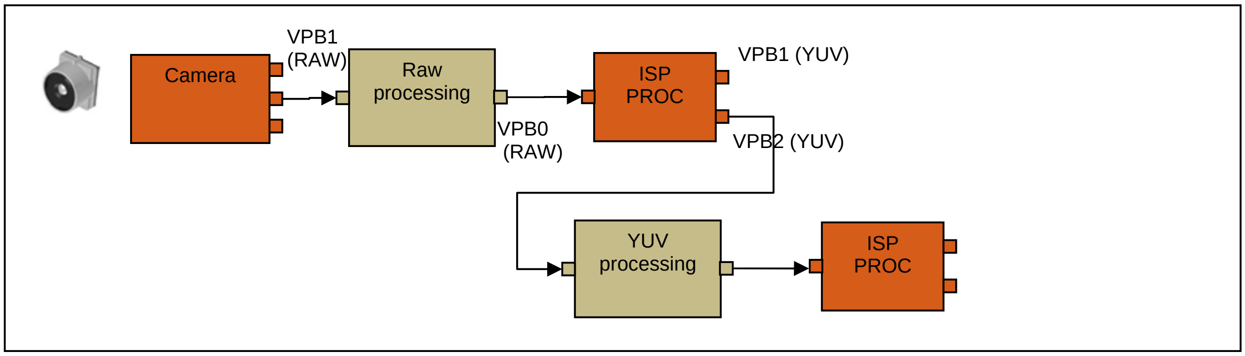


Figure 37. Open Pipe Concept

In concept, the ISP proc abstracts several bypass input modes of the ISP (Bayer Memory Load and RGB Memory Load) and bypass output modes (various Bayer Memory Store points).

But currently the supported bypass modes are limited to BML after Rx. Typically RML (capability to load RGB30) and BMS (capability to output Raw Bayer) are not supported at the ISP Proc ports. Only YUV formats are supported on VPB1 and VPB2. RGB formats are not supported as they considerably increase the camera IQ tuning work. As such the only mode that is currently supported is raw bayer input, YUV outputs.

For further explanations on bypass modes, please refer to 5.

Flexibility also comes from the fact that individual blocks of the ISP pipe can be disabled/enabled.

The ISP proc is involved in all still use cases. It is not part of video dataflows where the reconstruction always is performed on-the-fly.

This component does not control the sensor and does not abstract metering. Those are handled by the camera OMX component.

In still image the camera OMX component delegates the following processings to the ISP due to the fact part of the pipe is not entered:

Settings	IP	Comments
DZ and pan-tilt	Scaler and crop	Rely on Extradata from Omx Camera
DG appliance		Rely on Extradata from Omx Camera
Contrast	YUV Coder	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings).
Saturation	YUV Coder	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
Brightness	YUV Coder	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
Sharpening	Adsoc(Peaking)	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
YUV Range	YUV Coder	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
Special Effects - Sepia	Matrix	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
Special Effects - Grayscale	Matrix	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
Special Effects - Negative	SpecialFX	Rely on Extradata from Omx Camera or OMX Config (depends on CommonApplyUserSettings)
Gamma Correction	Gamma	Not user setting
Colour Enhancement	Matrix	Not user setting
Specific Noise Reduction – Scorpio	Scorpio	Not user setting
Noise Filtering	Duster	Not user setting
Rotation		Separate OMX Config

Table 52 Settings supported by the ISP Proc

Those settings are not provided by the IL client directly to the ISP proc. Instead they are applied on the camera OMX component and conveyed through extradata to the ISP proc (user settings extradata and capture parameters). This avoids duplicating the settings on both components and complexifying the data OMX API. On top of the user settings, the ISP proc needs more information related to captured raw frames (frame description).

Those 3 extradata types enable the cooperation between the camera and the ISP proc OMX components and a successful image reconstruction.

For comprehensive information on extradata, please refer to /extradata chapter/.

Component name : OMX.STE.ISP
Component Role : iv_processor.2outp.isp

12.1 Data Ports

The ISP Proc is an Extended IV Processor. It supports 1 input port and 2 output ports. The ISP proc does not implement buffer sharing.

The table below summarizes the purpose of each port in still mode and associated constraints.

Port Index	Dir	Formats	Description	Comments
VPB0 Non supplier	Input	RAW8 / RAW12	Input "ideal" raw bayer	Abstracts the BML0 i.e. the BML point before the DMCE. FFOV could be used except for cases like sport mode, pre-capture where we are obliged to do binning. DMA vertical crop could be used for digital zoom (possible only in V1). By using this, ISP processing time will be independent of the digital zoom factor.
VPB1 allocator	Output	YUV422 intld, RGB888	Reconstructed YUV frame of low resolution (< XGA)	Mapped onto ISP LR pipe. No support for rotation. Raster formats only
VPB2 allocator	Output	420 MB, I420	Reconstructed YUV frame of high resolution	Mapped onto ISP HR pipe. Rotation only if YUV420 MB. Max 16 MPel.

Table 53 ISP data ports

Output Ports can be disabled depending on which pipe/CE needs to be used. Below diagrams show the possible modes of the ISP Proc.

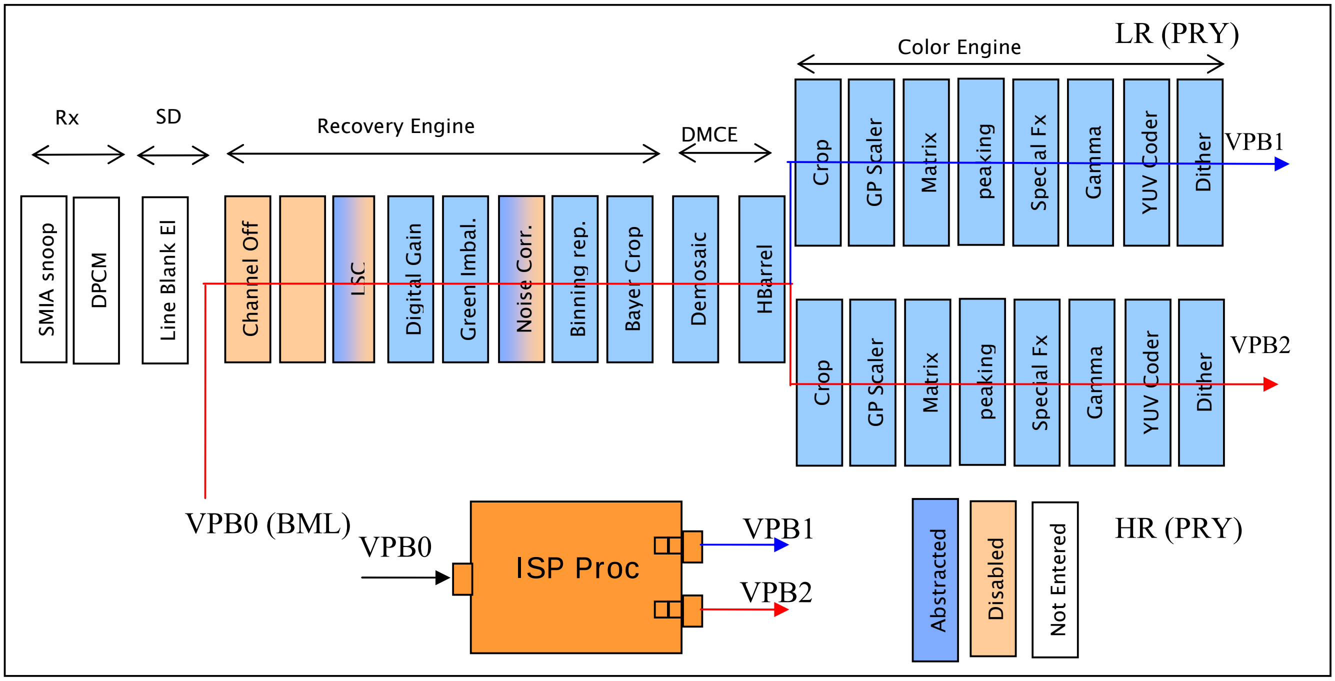


Figure 38. Pipes & ports

The formats and resolutions are statically defined through the port definition parameter. Their constraints are reminded below.

Index	Access	Description	Comments
OMX_IndexParamPortDefinition	r/w	nBufferCountMin = 1	

		nBufferSize = Pixel_data + metadata
		
		Pixel Data depends on format/mode as explained below.
		
		Pixel_data = OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth * nFrameHeight * 1 for Raw 8
		
		Pixel_data = OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth * nFrameHeight * 1.5 for Raw 12
		
		The width and height will change with stripe support. (Currently not planned)	

		bBufferContiguous = 1	
		nBufferAlignment = 8	
		OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth should be multiple of - 8 in case of RAW8. - 4 in case of RAW12. Width * height should be multiple of - 24 for RAW8. - 16 for RAW12. nFrameWidth = Sensor Width nFrameHeight = Sensor Height nStride = Sensor Width nSliceHeight = <same as default nFrameHeight>	

Table 54 VPB0 Port Def Params

Index	Access	Description	Comments
OMX_IndexParamPortDefinition	r/w	nBufferCountMin = 1	
		nBufferSize = OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth * nFrameHeight * 2 for YUV 422	
		bBufferContiguous = 1	
		nBufferAlignment = 8	
		OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth = multiple of 4 for YUV 422.	

Table 55 VPB1 Port Def Params

Index	Access	Description	Comments
OMX_IndexParamPortDefinition	r/w	nBufferCountMin = 1	

		nBufferSize = depends on format,
		nBufferSize = OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth * nFrameHeight * 1.5 for YUV 420 MB, planar
		nBufferSize = OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth * nFrameHeight * 2 for YUV 422	

		bBufferContiguous = 1	
		nBufferAlignment = 8 for YUV422 intld, and planar formats = 16 for MB formats	
		OMX_PARAM_PORTDEFINITIONTYPE.OMX_VIDEO_PORTDEFINITIONTYPE).nFrameWidth = multiple of - 16 for MB format - 8 for YUV 420, 422 planar - 4 for YUV 422 Intld	

Table 56 VPB2 Port Def Params

All values in the tables above are for 8500 V1 (A0).

The actual processing of buffers takes place only when all input/output buffers are available and the component is in executing.

The BMS / BML points can be programmed in the ISP FW .

12.2 Rotation

Rotation is a dynamic configuration supported by the camera OMX component through the OMX_IndexConfigCommonRotate/ OMX_CONFIG_ROTATIONTYPE index. OMX_CONFIG_ROTATIONTYPE.nRotation is used to configure the rotation angle.

The ISP Proc component supports 0°, 90°, 180° and 270° rotation in still mode, on VPB2 only in YUV420 MBtiled format.

12.3 Apply User settings

The ISP Proc acts in conjunction with the OMX Camera component to make a still use case work. As explained before the Camera provides lot of information to the ISP Proc inside the extradata of buffers provided to the ISP proc. Out of the set of extradata, there is User settings as explained in 25. These user settings are color effects, contrast, brightness, saturation, sharpness etc.

Now, the IL client can inform the ISP Proc whether or not it needs to apply these user settings onto the ISP pipe.

This can be done through index OMX_Symbian_IndexConfigApplyUserSettings. If true the ISP Proc relies to the user settings coming inside the extradata for ISP pipe configuration.

If this is false, the user settings should be applied to the ISP Proc by the IL Client.

This can be applied independently on the output ports of the ISP Proc (VPB1 and VPB2).

12.4 Contrast

For OMX Config see 11.10

12.5 Saturation

For OMX Config see 11.9

12.6 Brightness

For OMX Config see 11.11

12.7 Sharpening

For OMX Config see 11.12

12.8 YUV Range

For OMX Config see 11.13

13 ARM IV Proc

Component name : OMX.STE.VISUALPROCESSOR.2D-OPERATIONS.SW
Component Role : iv_processor.yuv

This is a standard IV Processor. It is used for processings of input image or video frames according to the port settings. Can be used as format convertion, resizing, rotation, mirroring etc.

OMX_IndexParamPortDefinition, OMX_IndexParamVideoPortFormat should be configured on both input and output ports for port configurations. The processings supported are given in 7.3.2

13.1 Mirroring

IL Client can set mirroring through OMX_IndexConfigCommonMirror defined in /11/ . None, Vertical, Horizontal and combination or vertical and horizontal is supported.

13.2 Rotation

The ARM IV proc is reactive to input scene orientation extradata.

When the component detects the scene orientation is misaligned with the input/output port definition param settings it should trigger a port settings changed on the output port .

Typically let's assume VPB0 is configured with 1600x1200 and VPB1 also is configured with 1600x1200, and input scene orientation is 90°, a port settings change event is sent on VPB1 by ARM IV proc.

Upon that event the IL client is assumed to call CommonRotate (defined in /11/) and sets VPB1 port definition parameters (nFrameWidth, nFrameHeight, nStride) to 1200x1600 and stride = 1200xbpp, and reenable the tunnel. IV proc propagates the scene orientation but takes into account that rotation was done, ie. It resets the scene orienatation to 0°.

Conversely if VPB0 is set to 1600x1200, VPB1 is set to 1200x1600, CommonRotate 90° is active and input scene orientation becomes 0°, a port settings changed event is called on VPB1. IL client is supposed to call CommonRotate with 0°, resets the VPB1 port def params to 1600x1200 and re-enable the tunnel. ARM IV proc still propagates the scene orientation extradata to the output port.

In case the scene orientation is 180°, still a port settings change event is called on VPB1. IL client sets CommonRotate 180° and only changes the nStride to be -nStride.

The IL client always is responsible to setting the same stride on both sides of a tunnel.

This mechanism makes possible to handle rotation before JPEG encoding and manage the fact the scene orientation can be changed between shots within a burst.

13.3 Cropping

Index OMX_IndexConfigCommonInputCrop / OMX_CONFIG_RECTTYPE defined in Khronos /11/ can be used to by the IL Client to define the input cropping rectangle if required.

13.4 Scaling

Index OMX_IndexConfigCommonScale / OMX_CONFIG_SCALEFACTORTYPE defined in Khronos /11/ can be used to define the amount of scaling. xWidth and xHeight need to be defined in % in Q16 format.

13.5 YUV Range

The YUV range is controlled through the OMX extension introduced in: OMX_Symbian_IndexParamCommonColorPrimary
/ OMX_Symbian_PARAM_COLORPRIMARY. Full range, BT601 and BT709 are supported.

OMX_Symbian_PARAM_COLORPRIMARYTYPE in OMX_Symbian_PARAM_COLORPRIMARY	Description	Khronos	Extensions
OMX_Symbian_ColorPrimaryFullRange	YUV range from 0 to 255	/	SHAI 1.2
OMX_Symbian_ColorPrimaryBT601	YUV range from 16 to 235 or 240 for chrominance with BT.601 defined YUV to RGB color conversion matrix	/	SHAI 1.2
OMX_Symbian_ColorPrimaryBT709	YUV range from 16 to 235 or 240 for chrominance with BT.709 defined YUV to RGB color	/	SHAI 1.2

	conversion matrix		

Table 57 YUV Range Values

14 Compositional IV Proc (B2R2)

Component name : OMX.STE.VISUALPROCESSOR.2D-OPERATIONS.HW
Component Role : iv_processor.yuv

This is a standard IV Processor. It is used for processings of input image or video frames according to the port settings. Can be used as format convertion, resizing, rotation, mirroring etc. B2R2 HW would be used for performing these processings.

OMX_IndexParamPortDefinition, OMX_IndexParamVideoPortFormat should be configured on both input and output ports for port configurations. The processings supported are given in 7.3.1.

14.1 Mirroring

IL Client can set mirroring through OMX_IndexConfigCommonMirror defined in /11/ . None, Vertical, Horizontal and combination or vertical and horizontal is supported.

14.2 Rotation

IL Client can set mirroring through OMX_IndexConfigCommonRotate /OMX_CONFIG_ROTATIONTYPE defined in /11/ .

14.3 Cropping

Index OMX_IndexConfigCommonInputCrop / OMX_CONFIG_RECTTYPE defined in /11/ can be used to by the IL Client to define the input cropping rectangle if required.

14.4 Scaling

Index OMX_IndexConfigCommonScale / OMX_CONFIG_SCALEFACTORTYPE defined in /11/ can be used to define the amount of scaling. xWidth and xHeight need to be defined in % in Q16 format.

14.5 YUV Range

The YUV range is controlled through the OMX extension introduced in: OMX_Symbian_IndexParamCommonColorPrimary / OMX_Symbian_PARAM_COLORPRIMARY. Full range, BT601 and BT709 are supported.

OMX_Symbian_PARAM_COLORPRIMARYTYPE in OMX_Symbian_PARAM_COLORPRIMARY	Description	Khronos	Extensions
OMX_ STE_ColorPrimaryFullRange	YUV range from 0 to 255	/	SHAI 1.2
OMX_ STE_ColorPrimaryBT601	YUV range from 16 to 235 or 240 for	/	SHAI 1.2

	chrominance with BT.601 defined YUV to RGB color conversion matrix		
OMX_ STE_ColorPrimaryBT709	YUV range from 16 to 235 or 240 for chrominance with BT.709 defined YUV to RGB color conversion matrix	/	SHAI 1.2

Table 58 YUV Range Values

15 Norcos IV Proc

Component name : OMX.STE.VISUALPROCESSOR.LLNR.YUV.NORCOS
Component Role : iv_processor.llnr.yuv

This is an IV Processor which offers chroma noise reduction in the YUV domain. Chroma noise reduction adaptively corrects chroma noise by detecting whether the region is associated to texture or uniform region. Filtering is made stronger when region does not contain details.

Noise reduction in YUV domain is operating on chroma only and on a sub-sampled version of chroma by 4x4 compared to YUV422. This means for an XGA image there are 256x192 U values and same for V.

The algorithm runs inside on the ARM side.

15.1 Algorithm Control

The IL Client can configure this standard IV processor using index OMX_Symbian_IndexConfigCommonLLNR/ OMX_Symbian_CONFIG_HIGHLEVELCONTROLTYPE .

OMX_Symbian_ IndexConfigCommonLLNR OMX_Symbian_ CONFIG_HIGHLEVELCONTROLTYPE	Description	Khronos	Extensions
OMX_Symbian_ControlOff	OFF	/	SHAI 1.2
OMX_Symbian_ControlOn	ON	/	SHAI 1.2
OMX_Symbian_ControlAuto	Auto	/	SHAI 1.2

Table 59 Norcos Configuration

When OFF, the Omx component always acts in a bypass mode. Which means that the Omx component is connected in the dataflow but is not doing any processings. When ON, the Omx component is always on which means that noise reduction always takes place on all the frames that pass through this component.

When in Auto mode, this component adapts itself to the stream according to extradata information contained in the inputs buffers.
The algorithm is automatically activated in low-light situation where chroma noise is very visible. The information is found in the ‘still-pack’ capture context extradata that accompanies the YUV data. In case the light conditions are good enough, the processing is bypassed in auto mode.

Other than above, OMX_IndexParamPortDefinition and OMX_IndexParamVideoPortFormat need to br programmed at the input and output ports.

16 Face Tracker IV Analyzer

Component Name: OMX.STE.VISUALANALYZER.FT
Component Role: iv_analyzer.nooutp.ft

This is a non-standard Omx component of IV Analyzer without output port type.

16.1 Region of Interest Selection

The IL Client can configure the FT Omx using the index OMX_Symbian_IndexConfigCommonROISelection / OMX_Symbian_CONFIG_ROISELECTIONTYPE. This config sets the selection parameters and user selections done for priority faces.

The fields are explained below:

OMX_Symbian_ CONFIG_ ROISELECTIONTYPE	Description	Khronos	Extensions
bReset	reset all selections done. Rest of the fields are not used	/	SHAI 1.2
bSelect	On/off the selection	/	SHAI 1.2
OMX_Symbian_ RELATIVERECTTYPE	Indicates the region where the object to be selected is located in the scene. This is mainly for sending user co-ordinates for eg. A rectangle which the user draws as the primary face to be detected. Mainly for face priority. The results returned by the FD algorithm for priority face are overridden by the user settings.	/	SHAI 1.2
nROIID	This field can contain the ID of object in order to assist object selection. 0 means undefined and is the default value. This makes more sense if exact co-ordinates are not available.	/	SHAI 1.2
OMX_Symbian_ ROIOBJECTTYPE	OMX_Symbian_RoiObjectTypeFace	/	SHAI 1.2

Table 60 Region Selection

16.2 Region of Interest Information

OMX_Symbian_IndexConfigCommonROI/ OMX_Symbian_CONFIG_ROITYPE is used to provide region of interest information by the IV Analyzer.

nNumberOfROIs indicate the number of regions. sROIs which is of type OMX_Symbian_ROIOBJECTINFOTYPE (in this case OMX_Symbian_RoiObjectTypeFace always) provides an array of ROIs and their associated information.

OMX_Symbian_ROIOBJECTINFOTYPE is composed of the following fields

OMX_Symbian_ ROIOBJECTINFOTYPE in OMX_Symbian_ CONFIG_ROITYPE	Description	Khronos	Extensions
nROIID	ID of the region, 0 by default	/	SHAI 1.2
OMX_Symbian_ RELATIVERECTTYPE	Relative rectangle information of the object.	/	SHAI 1.2
nPriority	Priority of the region, 0 being the highest	/	SHAI 1.2
OMX_Symbian_ ROIOBJECTTYPE	Type of object OMX_Symbian_RoiObjectTypeFace	/	SHAI 1.2
OMX_Symbian_ 3DORIENTATIONYTYPE	3D orientation of object nYaw nPitch nRoll	/	SHAI 1.2

Table 61 ROI support

The face computation completion is notified to the IL client through an OMX functional Event. Upon notification, the IL client retrieves the OMX_Symbian_CONFIG_ROITYPE structure from the analyzer and feeds it back to the camera OMX component as a standard OMX_SetConfig. This will help in assisting 3A if the Omx Camera is configured in that way.

Computational cost depends on

- max number of faces
- min size of the detected face (1/20 of the longer dimension)
- search area
- allowed face rotations.

Few searches have an important latency (corresponding to a new scene, latency less than 0.5s). Other searches on a scene that has been already searched consume much less time (TBD).

The face detection analyzer must interpret the input orientation extradata.

VGA input frame is assumed for our dataflows.

17 Object Tracker IV Analyzer

Component Name: OMX.STE.VISUALANALYZER.OT
Component Role: iv_analyzer.nooutp.ot

This is a non-standard Omx component of IV Analyzer without output port type.

Rest is same as for FT Analyzer except that OMX_Symbian_ROIOBJECTTYPE is OMX_Symbian_RoiObjectTypeObject.

17.1 Region of Interest Selection

Please refer to 16.1

17.2 Region of Interest Information

Please refer to 16.2

18 RER IV Analyzer

Component Name: OMX.STE.VISUALANALYZER.RER
Component Role: iv_analyzer.rer

This is an iv analyzer with an output port.
Red Eye Reduction analyzer is the one which has an output port and provides the anaysis information as metadata at its OPB0 buffer.
The extradata that is output through its OPB output port is OMX_Symbian_CONFIG_ROITYPE (same as in FT).

This component works on RGB888 data.

18.1 Algorithm Complexity

OMX_Symbian_ IndexConfigCommonREDComplexity OMX_Symbian_ CONFIG_ ALGORITHMCOMPLEXITYTYPE	Description	Khronos	Extensions
OMX_Symbian_ AlgComplexityLow	ID of the region, 0 by default	/	SHAI 1.2
OMX_Symbian_ AlgComplexityMedium	Relative rectangle information of the object.	/	SHAI 1.2
OMX_Symbian_ AlgComplexityHigh	Priority of the region, 0 being the highest	/	SHAI 1.2

Table 62 RED Complexity Levels

There might be 3 complexity levels for both RED/REC:

1. FULL RED (used when flash mode is RED EYE FLASH, max 1s latency) / High complexity
2. FAST RED (used when flash mode is set to auto or forced on, max 400 ms latency) / medium complexity
3. RED for snap (used along with all the above flash mode, on snap stream, target latency unknown) / low complexity

The complexity hint should mostly affects the detection and not the correction.

The analyzer exploits input capture context extradata where it typically uses

- The flash status during the capture
- The flash preset (forced-on, forced-off, auto, anti-red-eye flash)
- The estimated distance of the subject(s)

2/ and 3/ also use face metadata embedded in the capture context extradata. 1/ does not use those face metadata (more exhaustive search).

18.2 Algorithm Enabler

Index OMX_Symbian_IndexConfigCommonRedEyeRemoval can be used to set on or Off the detection mechanism inside the OMX component. This is required as the RER components are bypassed when flash is OFF for better performances.

19 RER IV Controlled Processor

Component Name: OMX.STE.CONTROLLED.VISUALPROCESSOR.RER
Component Role: iv_controlled_processor.rer

This Omx component is an Extended IV Controlled Processor. It relies on metadata coming from the RED IV Analyzer at its OPB0 port (OMX_Symbian_CONFIG_ROITYPE). Red eyes are then removed from the input images received at VPB0 and corrected images are sent at output port VPB1.

This component works on I420 format for input frames.

19.1 Algorithm Enabler

Index OMX_Symbian_IndexConfigCommonRedEyeRemoval can be used to set on or Off the correction mechanism inside the OMX component. This is required as the RER components are bypassed when flash is OFF for better performances.

20 JPEG Encoder

Component Name: OMX.STE.JPEGENCODER.HW (hw accelerated)
Component Name: OMX.STE.JPEGENCODER.SW (sw accelerated)
Component Role: image_encoder.jpeg

This omx component encodes the input image stream into a JPEG image. Both JFIF and EXIF image formats are supported. But the thumbnail insertion is done inside the EXIF Builder and not in this encoder component.

There is no support for rotation.

20.1 EXIF Headers

This is applicable only if output format is EXIF.

When the input buffer contains EXIF (including MakerNote), the JPEG encoder adds those tag values in the JPEG header. This is detected when OMX_BUFFERFLAG_EXTRADATA flag is set and EXIF extradata are decoded. This extradata are initialized by the Omx camera and propogate through all components in still OMX stream. The EXIF extradata contains the APP1 header, in a format compatible with the EXIF/TIFF standard. They are copied as is in the JPEG binary.

In case the EXIF extradata are not found at the input of the JPEG encoder and JPEG encoder is configured to produce EXIF, the JPEG encoder inserts empty EXIF segment with just necessary EXIF headers fields (EXIF segment length and IFD0 offset).

EXIF header has a maximum size of 64 kB (including the DCF thumbnail).

20.2 Maximum JPEG size

This index OMX_Symbian_IndexConfigImageMaxJpegSize puts a limit on the maximum allocated size in bytes for the encoded thumbnail as 1st IFD of an EXIF container. Only JPEG mandatory segments are included in this :SOI, DQT, DHT, SOF, SOS, compressed scan, E0I.

20.3 Quality Factor

The IL Client can configure the quality factor to be used for JPEG encoding using index OMX_IndexParamQFactor. nQFactor.

Default nQFactor could be 80 – 85. nQFactor can have any value between 1 and 100.

The scaling method for the given quality value (1 to 100) implemented by the omx component must be supported:

```
if (qualityvalue < 50)
   scalefactor = 5000 / qualityvalue;
else
   scalefactor = 200 - qualityvalue *2;
with:
quantiser[i] = (default_table[i] * scalefactor + 50L) / 100L;
```

The quantization tables (one for luminance and one for chrominance) given in the Jpeg standard must be used.

The QFactor is taken into account if OMX_IndexConfigMaxJpegSize is set to non-zero.

20.4 GPS location inside EXIF headers

The IL client can insert GPS information into the EXIF image by configuring the index OMX_Symbian_IndexConfigGPSLocation / OMX_Symbian_CONFIG_GPSLOCATIONTYPE.

The Jpeg encoder embeds these GPS information inside the EXIF headers and might also add to the extradata. But this should only be done if the location is known and output format is EXIF.

OMX_Symbian_ CONFIG_GPSLOCATIONTYPE	Description	Khronos	Extensions
nLatitudeDegrees	Latitude information in degrees	/	SHAI 1.2
nLatitudeMinutes	Latitude information in minutes	/	SHAI 1.2
nLatitudeSeconds	Latitude information in seconds	/	SHAI 1.2
nLongitudeDegrees	Longitude information in degrees	/	SHAI 1.2
nLongitudeMinutes	Longitude information in minutes	/	SHAI 1.2

nLongitudeSeconds	Longitude information in seconds	/	SHAI 1.2
nAltitudeMeters	Altitude information in meters	/	SHAI 1.2
bLatitudeRefNorth	Latitute reference is north or not	/	SHAI 1.2
bLongitudeRefEast	Longitude reference is east or not	/	SHAI 1.2
bAltitudeRefAboveSea	Altitude reference is above sea level or not	/	SHAI 1.2
bLocationKnown	Location is known or not	/	SHAI 1.2
bTimestamp	GPSInfo Exif.GPSInfo.GPSTimeStamp Indicates the time as UTC (Coordinated Universal Time). <TimeStamp> is expressed as three RATIONAL values giving the hour, minute, and second (atomic clock).		
bStatus	GPSInfo Exif.GPSInfo.GPSStatus. Indicates the status of the GPS receiver when the image is recorded. "A" means measurement is in progress, and "V" means the measurement is Interoperability.		

Table 63 GPS Information

21 Video Splitter

Component Name: OMX.STE.VIDEO.SPLITTER
Component Role: video_splitter.generic

This is a broadcaster component. Following diagram shows its operation.

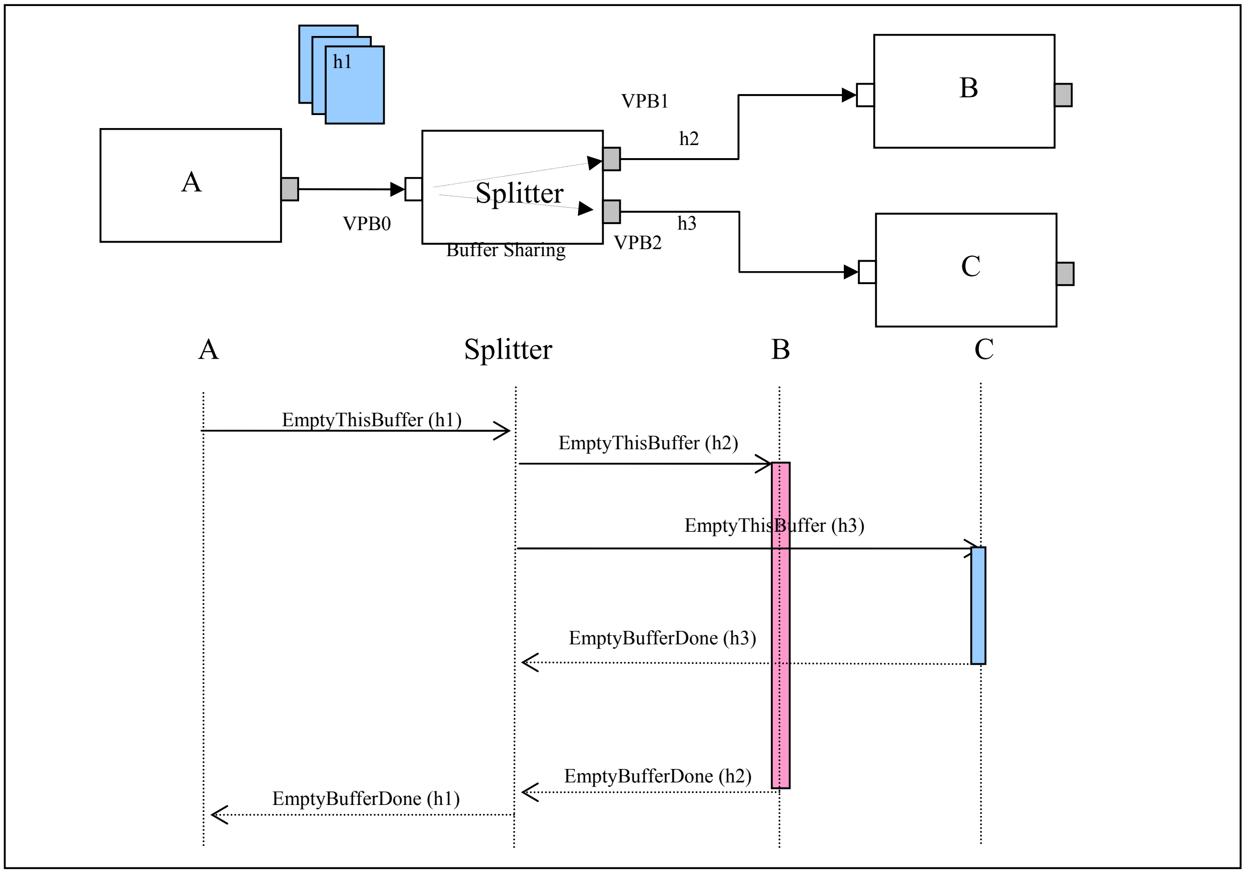


Figure 39. Normal splitter Sequence Diagram

For one physical input buffer, there are 3 buffer headers (h1, h2, h3 as illustrated above).

When delivering the input buffer to its sharing outputs ports, the splitter maintains a reference counter to detect when the output buffers have been consumed by the downstream components (cycle of EmptyThisBuffer/FillThisBuffer) as illustrated in the figure above. Once both EmptyBufferDone are received from B and C, the buffer can be given back to A.

The downstream components cannot perform in-place processings. As such the output tunnels must be read-only.

22 Video Sequential Splitter

Component Name: OMX.STE.VIDEO.SPLITTER.SEQUENCER
Component Role: video_splitter.sequencer.generic

This component is also a broadcaster with inherent priorities defined in its output ports. Priorities decrease with increasing port number.

Following figure shows its operations.

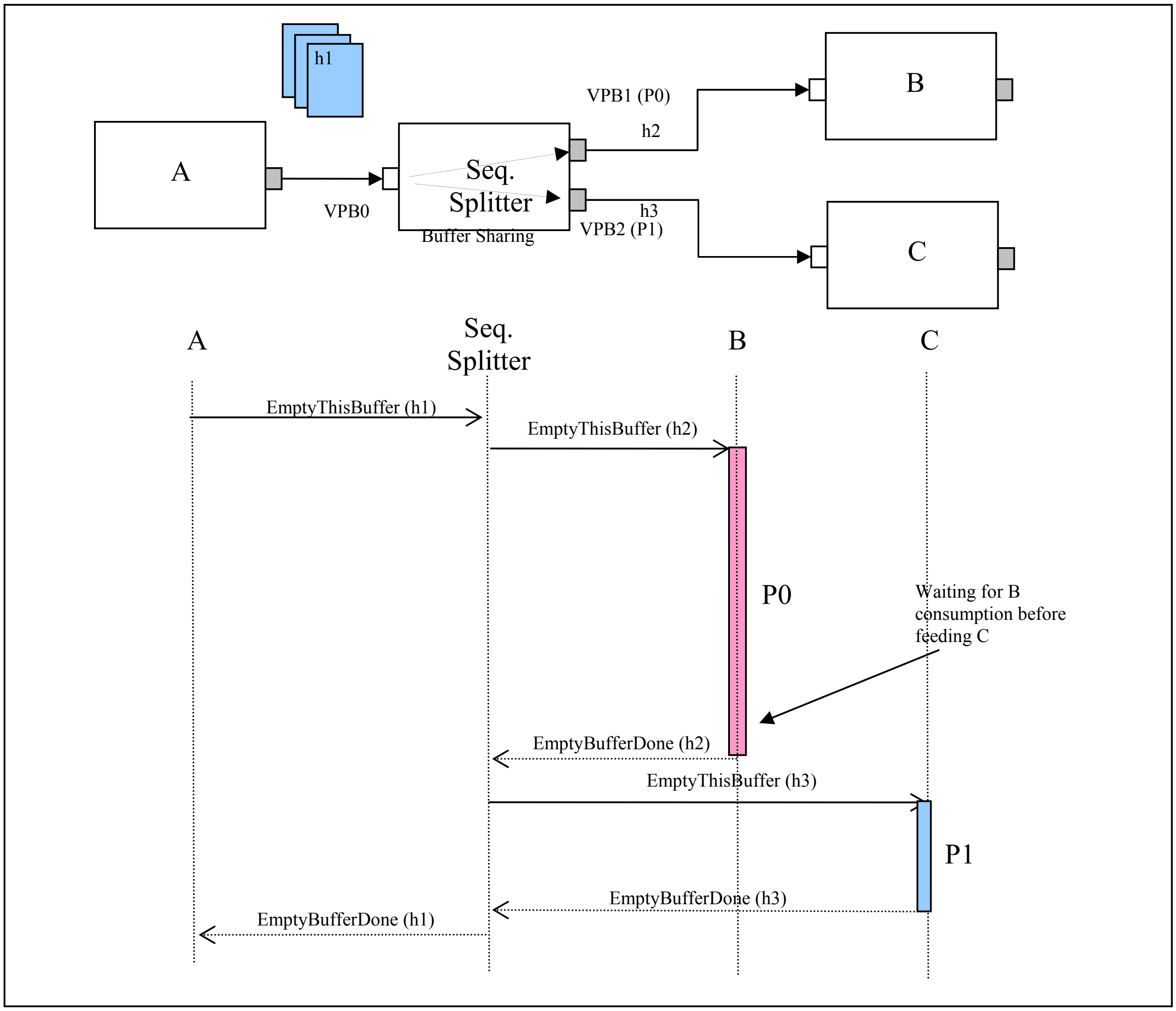


Figure 40. Sequential splitter Sequence Diagram

Output tunnels are not requested to be read-only since there is a guarantee that buffers have been used by downstream components before they are passed to other output ports.

23 EXIF Builder

Component Name: OMX OMX.STE.THUMBNAIL_ADDER.EXIF
Component Role: thumbnail_adder.2inp.exif

OMX_IndexParamPortDefinition and OMX_IndexParamImagePortFormat should be configured on all the three ports.

The thumbnail is inserted in IFD1 section. GPS information can be inserted if provided by the the IL client in the IFD0 tag (index?). Maybe OMX_Symbian_IndexConfigGPSLocation / OMX_Symbian_CONFIG_GPSLOCATIONTYPE defined in Extensions SHAI 1.2 can be used.

Most EXIF tag values are directly inserted by the primary JPEG encoder.

Its proxy relies on an ARM NMF component.

24 Display Sink

Component name : OMX.ST.VFM.DISPLAY.SINK
Component Role : display.sink

This is a standard IV Processor. It is used for processings of input image or video frames according to the port settings anad direct display to screen. Can be used as format convertion, resizing, rotation, mirroring etc. B2R2 HW would be used for performing these processings and MCDE for final display.

OMX_IndexParamPortDefinition, OMX_IndexParamVideoPortFormat should be configured on both input and output ports for port configurations.

24.1 Mirroring

IL Client can set mirroring through OMX_IndexConfigCommonMirror/ OMX_CONFIG_MIRRORTYPE defined in /11/ . None, Vertical, Horizontal and combination or vertical and horizontal is supported.

24.2 Rotation

IL Client can set mirroring through OMX_IndexConfigCommonRotate
/OMX_CONFIG_ROTATIONTYPE defined in /11/ .

24.3 Cropping

Index OMX_IndexConfigCommonInputCrop / OMX_CONFIG_RECTTYPE defined in /11/ can be used to by the IL Client to define the input cropping rectangle if required.

24.4 Scaling

Index OMX_IndexConfigCommonScale / OMX_CONFIG_SCALEFACTORTYPE defined in /11/ can be used to define the amount of scaling. xWidth and xHeight need to be defined in % in Q16 format.

24.5 Output Position setting

Index OMX_CONFIG_POINTTYPE can be used to set the output position of the window to be displayed.

25 Extradata

Extradata are in-band information added at the end of the OMX buffer payload, after the actual image data. They need to be considered when allocating the buffer since sufficient memory must be provisioned to add those extradata after the very pixel payload.

Extradata are laid out as OMX_OTHER_EXTRADATATYPE structures, as pictured below.

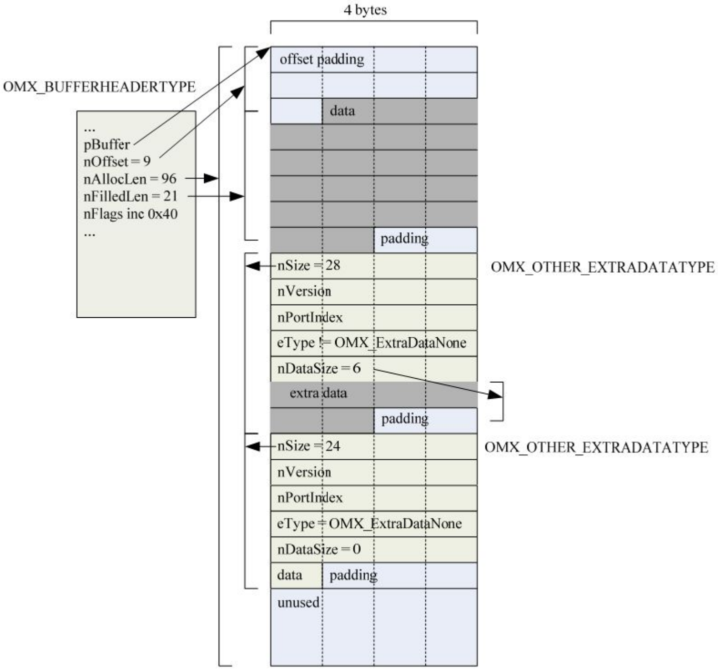


Figure 41. Generic structure of metadata (excerpt of the OMX IL specification)

Any buffer which contains extradata in its payload is flagged accordingly: the nFlags of the buffer header is set to OMX_BUFFERFLAG_EXTRADATA. The buffer header enables to easily locate the first extradata. Then different extradata structs are concatenated.

Any component receiving such a buffer shall propagate the extradata contained in this input buffer to the output buffer it contributes to, with possible modification. Hence, the extradata is propagated throughout a given stream from the producer to the consumer.

Extradatas are grouped into 2 big extradata packages:

- The still pack that contains capture parameters, user settings, Frame Description, EXIF, ROIs extradatas
- The video pack that contains capture parameters video stab, RGB histograms, ROIs

All the extradata that are mentioned here are produced by the camera OMX component. Still pack is conveyed on VPB1 stream. Video pack is conveyed on VPB0 and VPB2 streams.

OMX 1.1.2 does not allow to negotiate the size of the extradata before allocation of the buffers. The location of extradata per buffer thus is product dependent. The drawing hereafter explains the buffer requirements that must be given by the various OMX components so that at the end the right buffer size is negotiated so that the buffer can contain both the data and its extradata.

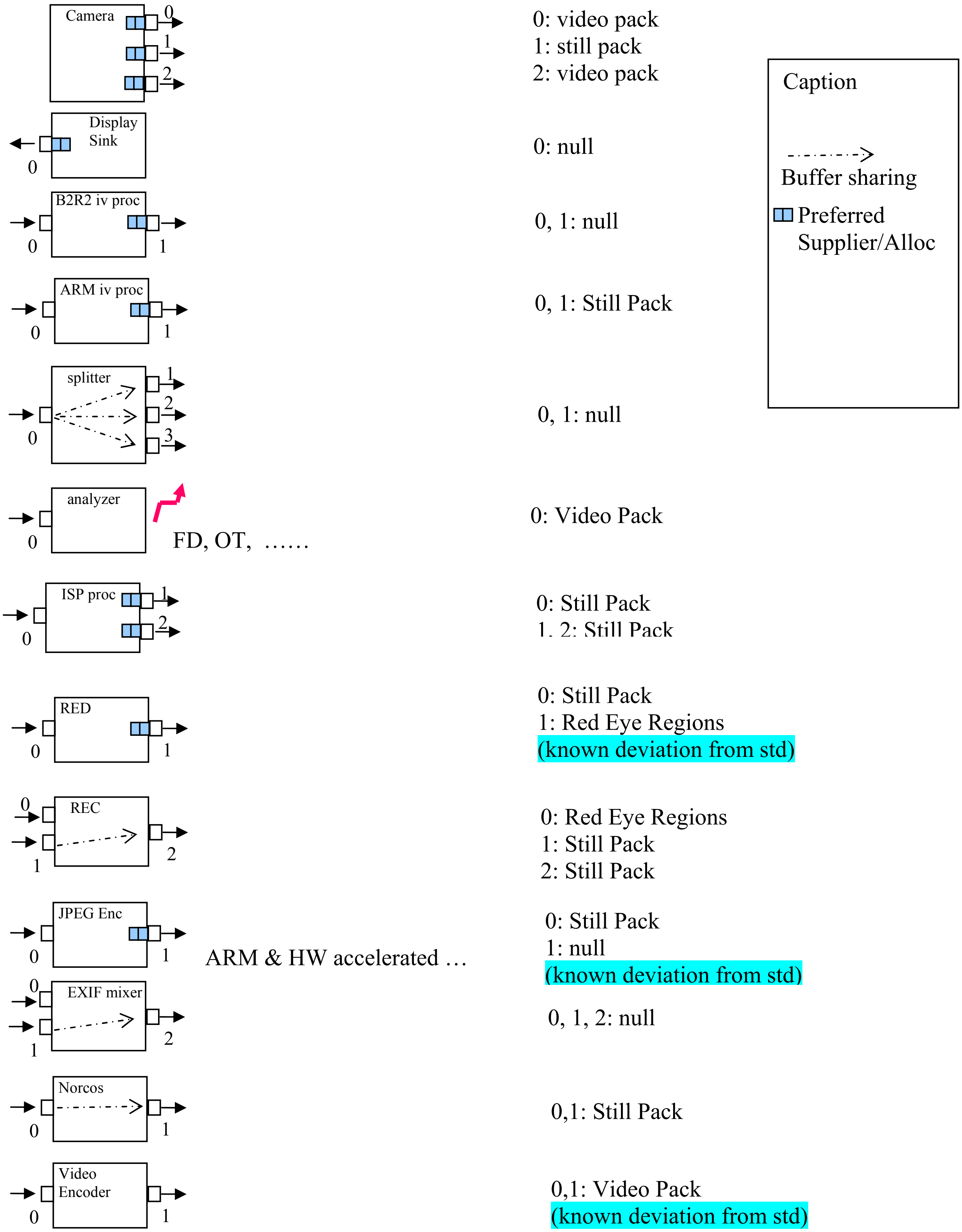


Figure 42. Extradata Provision Per Component (taken into account in nBufferSize)

Pack	OMX Index	OMX Type	Description
Still Pack	OMX_Index_ParamCommonExtraCameraUserData	OMX_ExtraDataCameraUserSettings	Read only and always delivered when in image mode. To have an index makes sense only to allow IL client to query its presence and realize in video mode is missing. Indicates that the data payload contains user camera settings needed by other components from camera related datapaths.
Still Pack	OMX_Index_ParamCommonExtraEXIFData	OMX_ExtraDataEXIFAppMarker	Indicates that the data payload contains EXIF meta data as described in EXIF 2.2.
Still/Video Pack	OMX_Index_ParamCommonExtraROIData	OMX_ExtraDataROI	Indicates that the data payload contains meta data related to image analysis.
Video Pack	OMX_Index_ParamCommonExtraVideoStabilizationData	OMX_ExtraDataVideoStabilization	Indicates that the data payload contains meta data needed for video stabilization.
Still Pack, Video Pack	OMX_Index_ParamCommonExtraCameraFrameDescriptionData	OMX_ExtraDataFrameDescription	Read only and always delivered in image mode. Indicates that the data payload contains SMIA ISL metadata.
Still Pack	OMX_Index_ParamCommonExtraCaptureParametersData	OMX_ExtraDataCaptureParameters	Read only and always delivered in image mode. Indicates that the data payload contains specific extra data for raw captures (extended SMIA ISL).
Video Pack	OMX_Index_ParamCommonExtraHistogramData	OMX_ExtraDataHistogram	Indicates that the data payload contains histogram.
Video Pack	OMX_Symbian_Index_ParamProductionTestsData	OMX_Symbian_ExtraDataProductionTests	

Table 64 OMX Extradata definitions

25.1 Capture Parameters

This extradata is generated on all camera output ports. On still stream it is part of the capture context that enables off-line image reconstruction in the ISP proc. On other streams it is likely to be used by analyzers, processors, … It is part of both the still and video pack. On still it reflects the characteristics of the still raw frame whereas on video streams, it reflects the latest known characteristics.

```
typedef struct OMX_CAPTUREPARAMETERSTYPE {
   OMX_VERSIONTYPE nExtraDataVersion;

          //Scene characterization
   OMX_Symbian_ORIENTATIONORIGINTYPE eSceneOrientation;
   OMX_U32 nMotionLevel;
   OMX_S32 nLuxAmbiant;

          //AEW
```

```
     OMX_S32 nTargetTotalExposure;
     OMX_S8  nExposureStatus;  // This is former nAECdistanceFromConvergence
     OMX_S16 nAnalogGain;
     OMX_S16 nDigitalGain;
     OMX_U32 nExposureTime;
     OMX_U32 nApertureFNumber;
     OMX_S8 nNDFilter;
     OMX_U32 nSensitivity;

     //focus
     // Current ext focus status extension includes only nAutoFocusStatus below
            // This will need updating in sync with camera extension
     OMX_U32 nFocusingDistance;
     OMX_IMAGE_FOCUSSTATUSTYPE nAutoFocusStatus;
     OMX_Symbian_RECTTYPE xFocusArea;
     OMX_Symbian_FOCUSREGIONTYPE eFocusRegion;
     OMX_U32 nAFsubWOIIndex;
     OMX_S32 nAFsteps;

            //flash
     OMX_BOOL bFlashFired;
     OMX_U16 nFlashPowerDuringCapture;
     OMX_U16 nFlashCtrlPulseInUsec;

     OMX_S32 nGamma;
     OMX_U8 nGammaLUT[256]; //Gamma LUT table
     OMX_S32 nRGB2RGBColorConversion[9]; //Q8
     OMX_S32 nGainG; //Q8
     OMX_S32 nGainR;
     OMX_S32 nGainB;
     OMX_S32 nOffsetG;
     OMX_S32 nOffsetR;
     OMX_S32 nOffsetB;
     OMX_S32 nColorTemperature;
     OMX_S32 nRG;
     OMX_S32 nBG;
     OMX_S32 nContrast;
     OMX_U32 nBrightness;
     OMX_S32 nSaturation;
     OMX_S32 nSharpness;

   } OMX_CAPTUREPARAMETERSTYPE
```

25.2 User Settings

This extradata is part of the still pack. It is conveyed on VPB1. It reflects the IL clients settings without any modification.

```
   typedef struct OMX_CAMERAUSERSETTINGSTYPE {

     OMX_S32 nX;
     OMX_S32 nY;
     OMX_U32 xDigitalZoomFactor;
     OMX_U32 xDigitalZoomFactor;

     OMX_BOOL bOneShot;
     OMX_BOOL bContinuous;
     OMX_BOOL bFrameLimited;
     OMX_U32 nFrameLimit;
     OMX_U32 nFramesBefore;
     OMX_BOOL bEnableBracketing;
```

```
  OMX_Symbian_SCENEMODETYPE nSceneModePreset;
  OMX_WHITEBALCONTROLTYPE eWhiteBalControl;
  OMX_EXPOSURECONTROLTYPE eExposureControl;
  OMX_IMAGE_FLASHCONTROLTYPE eFlashControl;
  OMX_IMAGE_FOCUSCONTROLTYPE eFocusControl;
  OMX_U32 nFocusSteps;
  OMX_U32 nFocusStepIndex;

  OMX_U32 nLockingStatus;

  OMX_METERINGTYPE eMetering;
  OMX_S32 xEVCompensation;
  OMX_U32 nApertureFNumber;
  OMX_BOOL bAutoAperture;
  OMX_BOOL bAutoShutterSpeed;
  OMX_U32 nSensitivity;
  OMX_BOOL bAutoSensitivity;
  OMX_IMAGEFILTERTYPE eImageFilter;

  OMX_S32 nContrast;
  OMX_U32 nBrightness;
  OMX_S32 nSaturation;
  OMX_S32 nSharpness;
  OMX_BOOL bStab;

  OMX_Symbian_DOFHINTTYPE eDoFhint;
  OMX_Symbian_ROIOBJECTTYPE eROI;
} OMX_CAMERAUSERSETTINGSTYPE
```

25.3 Frame Description

Produced on all ports hence part of video and still pack. It is part of the capture context information that enables off-line image reconstruction.

```
typedef struct OMX_FRAMEDESCRIPTIONTYPE {
  OMX_U32 nSize;
  OMX_VERSIONTYPE nExtraDataVersion;

  OMX_U8 nPixelFormat;
  OMX_U8 nDataEndianess;
  OMX_Symbian_RAWIMAGEPRESETTYPE eRawPreset;
  OMX_U8 nColorOrder;
  OMX_U8 nBitDepth;
  OMX_S32 nDataPedestal;
  OMX_U8 nFrameCounter;
  OMX_U8 nImageDataReadoutOrder;

  OMX_U8 nIslLinesTop;
  OMX_U8 nIslLinesBottom;

  OMX_U16 nBlackColsLeftStart;
  OMX_U16 nBlackColsLeft;
  OMX_U16 nBlackColsRightStart;
  OMX_U16 nBlackColsRight;
  OMX_U16 nBlackRowsTopStart;
  OMX_U16 nBlackRowsTop;
  OMX_U16 nBlackRowsBottomStart;
  OMX_U16 nBlackRowsBottom;
```

```
    OMX_U16 nDarkColsLeftStart;
    OMX_U16 nDarkColsLeft;
    OMX_U16 nDarkColsRightStart;
    OMX_U16 nDarkColsRight;
    OMX_U16 nDarkRowsTopStart;
    OMX_U16 nDarkRowsTop;
    OMX_U16 nDarkRowsBottomStart;
    OMX_U16 nDarkRowsBottom;

    OMX_U16 nDummyColsLeftStart;
    OMX_U16 nDummyColsLeft;
    OMX_U16 nDummyColsRightStart;
    OMX_U16 nDummyColsRight;
    OMX_U16 nDummyRowsTopStart;
    OMX_U16 nDummyRowsTop;
    OMX_U16 nDummyRowsBottomStart;
    OMX_U16 nDummyRowsBottom;

    OMX_U16 nVisColsStart;
    OMX_U16 nVisCols;
    OMX_U16 nVisRowsStart;
    OMX_U16 nVisRows;

    OMX_U16 nSpatialSampling;
    OMX_U16 nScalingMode;
    OMX_U16 nZoomFactor;

} OMX_FRAMEDESCRIPTIONTYPE
```

25.4 Stabilization

The camera OMX component uses this structure to store the stabilization vectors. This extradata is output on VPB0 and VPB2. It is part of the video pack.

```
typedef struct OMX_DIGITALVIDEOSTABTYPE {
    OMX_U32 nSize;
    OMX_VERSIONTYPE nVersion;
    OMX_U32 nPortIndex;
    OMX_BOOL bState;
    OMX_U32 nTopLeftCropVectorX;
    OMX_U32 nTopLeftCropVectorY;
    OMX_U32 nMaxOverscannedWidth; // W+30%
    OMX_U32 nMaxOverscannedHeight, // H +30%
} OMX_DIGITALVIDEOSTABTYPE
```

25.5 ROIs

This extradata is produced by the camera and part of both still and video pack. It reflects the latest know ROIs set by the IL client to the camera component. Those ROIs can be mixed (faces, objects, …). Usually those ROIs are computed by analyzers.

```
typedef struct OMX_Symbian_CONFIG_ROITYPE {
    OMX_U32 nSize;              /**< Size of the structure in bytes */
    OMX_VERSIONTYPE nVersion;   /**< OMX specification version information */
    OMX_U32 nPortIndex;         /**< Port that this structure applies to */
    OMX_U32 nNumberOfROIs;      /**< Number of ROIs included in this config */
    OMX_Symbian_ROIOBJECTINFOTYPE sROIs[OMX_Symbian_MAX_NUMBER_OF_ROIS]; /**< Array
of ROIs */
} OMX_Symbian_CONFIG_ROITYPE
```

25.6 EXIF

EXIF payload, part of the still pack. It is bound to be embedded in the JPEG 0th IFD header.

```
typedef struct OMX_EXIFAPPMARKERTYPE {
    OMX_U32 nSize;
    OMX_VERSIONTYPE nVersion;
    OMX_U32 nPortIndex;
    OMX_U8 nAppMarkerData[1];
} OMX_EXIFAPPMARKERTYPE
```

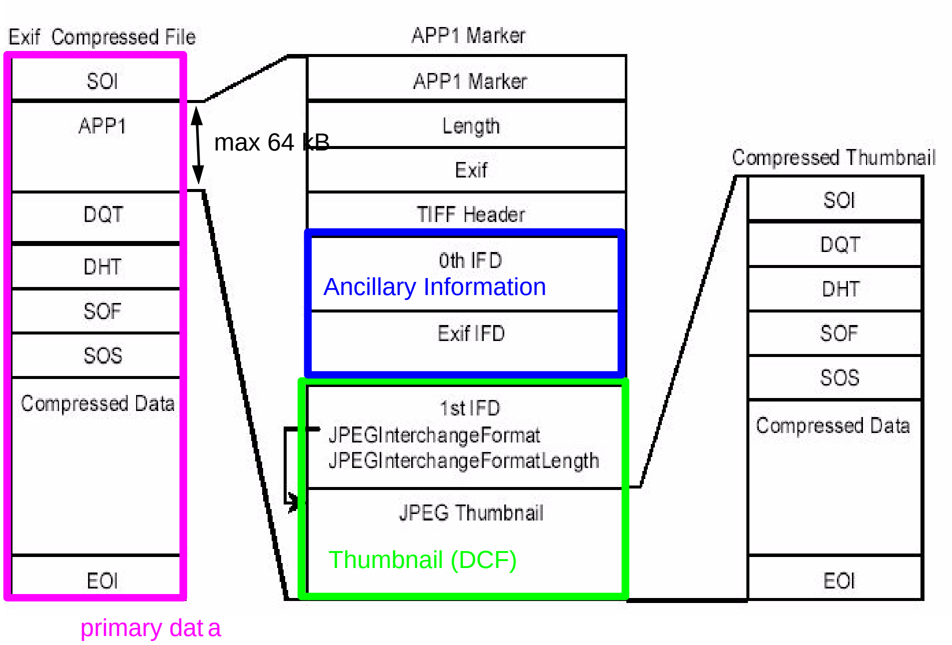


Figure 43. Compressed File Structure and APP1 marker segment

The IFD organization is reminded below (EXIF 2.2 excerpt).

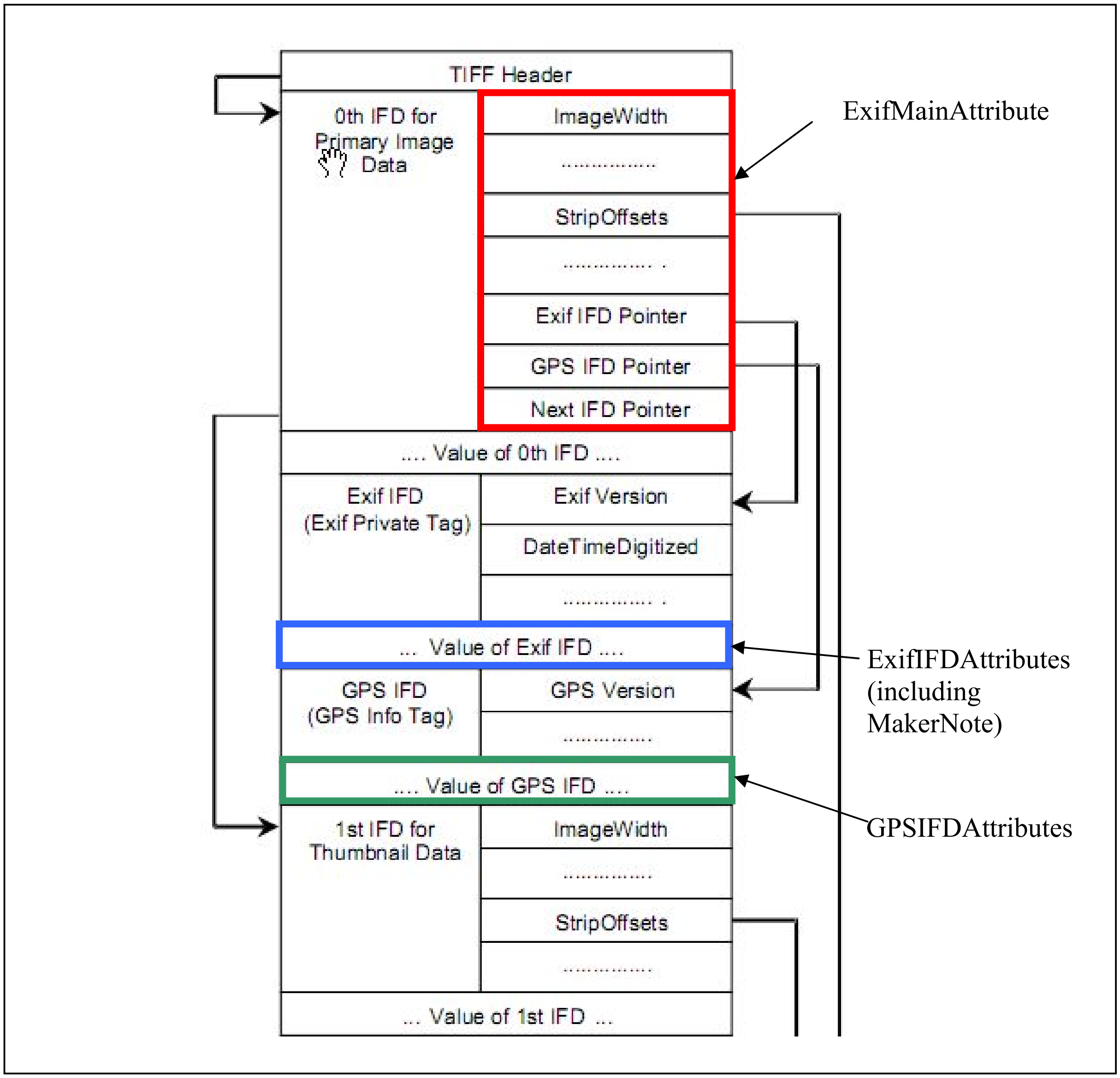


Figure 44. Oth IFD containing EXIF IFD and GPS IFD

The following tags are supported there. They are laid at as raw EXIF data as specified in the EXIF 2.2 standard. The tag values that are filled by the camera are described below. Other tags are provisioned but their value is not filled.

Type	Tag ID	Tags	Source	Size/Type	Value: static/ Dynamic	Value
TIFF 6.0	271	Make	Manufacturer	ascii	S	Customer specific
TIFF 6.0	272	Model	Model Name or number of the camera	ascii	S	Customer specific
TIFF 6.0	274	Orientation	The image orientation viewed in terms of rows and columns.	short	D	1 (always top left)
TIF	282	Xresolution	The number of pixels per	rational	D	Customer specific

F 6.0			*ResolutionUnit* in the *ImageWidth*			
TIFF 6.0	283	Yresolution	The number of pixels per *ResolutionUnit* in the *ImageLength* direction.	rational	D	Customer specific
TIFF 6.0	296	Resolution Unit	The unit for measuring *XResolution* and *YResolution* (inch or cm)	short	S	2 = inch
TIFF 6.0	531	YcbCrPositioning	The position of chrominance components in relation to the luminance component (centered, co-sited)	short	D	1= centered
EXIF	34665	Exif IFD Pointer	offset of the EXIF IFD	long	F	Computed by the camera proxy

Table 65 EXIF Main Attributes

Note the MakerNote information are embedded in this extradata. There is no plan today to support dynamic makernote. Makernote only are inserted in EXIF extradata.

EXIF IFD	34434	ExposureTime	Exposure time (in s)	Rational x 1	D	<exposure time>
EXIF IFD	33437	Fnumber	F-number of the camera	Rational x 1	D	
EXIF IFD	34855	ISOSpeedRatings	In case auto ISO is used, the value should reflect the value computed by the OMX camera	Short x any	D	<ISO number>
EXIF IFD	36864	Exif Version	EXIF version	Undefined x 4	F	“0220”
EXIF IFD	36867	DateTimeOriginal	Date & Time when the orginal data was created	Ascii	D	"YYYY:MM:DD HH:MM:SS"
EXIF IFD	36868	DateTimeDigitized	Date & Time when the original data was stored as digital data	Ascii	D	"YYYY:MM:DD HH:MM:SS"
EXIF IFD	37121	ComponentsConfiguration	order of Y, Cb, Cr in the components	undefined	S	1230 (Y, Cb, Cr)
EXIF IFD	37377	ShutterSpeedValue	shutter speed value in APEX units	Srational x1	D	V =-log2(<Exposure-Time>)
EXIF IFD	37378	ApertureValue	the lens aperture in APEX units	Rational x1	D	=3(2log2(F number))
EXIF IFD	37379	BrightnessValue (test/calibration purpose)	Brightness	Srationale x 1	D	<Brightness Value>
EXIF IFD	41493	ExposureIndex	EV compensation value	Rational x1	D	<EV compensation>
EXIF IFD	37382	SubjectDistance (test/calibration purpose)	The distance to the subject, given in meters.	Rationale x 1	D	<distance>
EXIF IFD	37383	MeteringMode	The metering mode	Short x 1	D	<spot mode>
EXIF IFD	37500	MakeNote	MakerNote Debug Information	Undefined, any	D	
EXIF	37384	LightSource	The kind of light source	Short x 1	D	According to WB

IFD			(daylight, fluo, tungsten, flash, ...)			preset
EXIF IFD	37385	Flash	status of flash when image was shot, one byte bitfield (fired? returned status, mode, function, red-eye)	Short x1	D	<Flash status>
EXIF IFD	37386	FocalLength	actual focal length of the lens in mm	Rationale x1	S	5.6
EXIF IFD	37396	SubjectArea	location and area of the main subject in the overall scene.	Short, 2 or 3 or 4	D	<area>
EXIF IFD	41483	FlashEnergy (test purpose)	Indicates the strobe energy at the time the image is captured	Rational x1	D	<current value> BCPS unit not complied with
EXIF IFD	40960	FlashPix Version	FlashPix version	undefined, 4	S	“0100”
EXIF IFD	40961	ColorSpace	Color space specifier	Short x 1	S	1 = “sRGB”
EXIF IFD	40962	PixelXDimension	Image Width the valid width of the meaningful image shall be recorded in this tag, whether or not there is padding data or a restart marker.	Short or long x 1	D	<ImageWidth>
EXIF IFD	40963	PixelYDimension	Image Height the valid width of the meaningful image shall be recorded in this tag, whether or not there is padding data or a restart marker.	Short or long x 1	D	<ImageHeight>
EXIF IFD	41985	CustomRendered	Indicates whether custom SW preprocessing was performed (2D lens shading or noise correction)	Short x 1	D	0 when no special preprocessing performed 1 when custom (color tones for instance)
EXIF IFD	41986	ExposureMode	Exposure mode when picture was shot (0=auto-exposuren 1 manual exposure, 2 auto-bracket)	Short x 1	D	0=auto
EXIF IFD	41987	WhiteBalance	WB mode when picture was shot (auto or manual)	Short x 1	D	0 auto 1 manual (preset)
EXIF IFD	41988	DigitalZoomRatio	Actual Digital zoom when picture was shot	Rational x 1	D	<DigitalZoomRatio>
EXIF IFD	41990	SceneCaptureType	Type of the scene that was shot(std, landscape, portrait, night scene)	Short x 1	D	1 standard 3 night mode
EXIF IFD	41991	GainControl	Degree of overall gain adjustment	Rational x 1	D	[0, 4]

EXIF IFD	41992	Contrast	direction of contrast processing applied by the camera when the image was shot.	Short x 1	D	Normal, soft, hard

Table 66 Supported 0th IFD tags

What size should be provisioned for this extradata?

1 = BYTE An 8-bit unsigned integer.,
2 = ASCII An 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL.,
3 = SHORT A 16-bit (2-byte) unsigned integer,
4 = LONG A 32-bit (4-byte) unsigned integer,
5 = RATIONAL Two LONGs. The first LONG is the numerator and the second LONG expresses the denominator.,
7 = UNDEFINED An 8-bit byte that can take any value depending on the field definition,
9 = SLONG A 32-bit (4-byte) signed integer (2's complement notation),
10 = SRATIONAL Two SLONGs. The first SLONG is the numerator and the second SLONG is the denominator.

25.7 RGB Histograms

RGB histograms are part of video pack. They are used by some analyzers (Automatic scene detection).

```
typedef struct OMX_Symbian_CONFIG_RGBHISTOGRAM {
   OMX_U32 nSize;             /**< Size of the structure in bytes */
   OMX_VERSIONTYPE nVersion;   /**< OMX specification version information */
   OMX_U32 nPortIndex;         /**< Port that this structure applies to */
   OMX_U32 nRed[256];          /**< Histogram for red color */
   OMX_U32 nGreen[256];        /**< Histogram for green color */
   OMX_U32 nBlue[256];         /**< Histogram for blue color */
} OMX_Symbian_CONFIG_RGBHISTOGRAM;
```

25.8 Red Eye Extradata

Those extradata are ST-Ericsson private. They only are used between the RED and REC OMX components.

25.9 Production Test Extradata

These data needs to be added for certain production test requirements. This should be provided at VPB0 while preview is running.

```
typedef struct OMX_Symbian_PRODUCTIONTESTTYPE {
   OMX_U32 nSize;
   OMX_VERSIONTYPE nVersion;
   OMX_U32 nPortIndex; // might not be needed
   OMX_U32 nIlluminanceValue;
   OMX_U32 nFlashChromaPointX; // 1st co-ordinate
   OMX_U32 nFlashChromaPointY; // 2nd co-ordinate

} OMX_Symbian_PRODUCTIONTESTTYPE
```

26 Annex

26.1 Examples of Overscanned resolutions

The table below provides overscanned resolutions for main VF/video resolutions with various overscan percentage.

The formulae applied to compute the overscanned resolution is

- W*(100+overscan_percentage)/100 –mod(W*(100+ overscan_percentage)/100, 16)
- H*(100+overscan_percentage)/100 –mod(H*(100+ overscan_percentage)/100, 16).

Overscan (%)	0	5	10	15	20	25	30
Width (pix)	1920	2016	2112	2208	2304	2400	2496
Height (pix)	1080	1120	1184	1232	1296	1344	1392
A:R	1.777778	1.8	1.783784	1.792208	1.777778	1.785714	1.793103
Width(MB)	120	126	132	138	144	150	156
Height(MB)	67.5	70	74	77	81	84	87

1080p+

Overscan (%)	0	5	10	15	20	25	30
Width (pix)	1280	1344	1408	1472	1536	1600	1664
Height (pix)	720	752	784	816	864	896	928
A:R	1.777778	1.787234	1.795918	1.803922	1.777778	1.785714	1.793103
Width(MB)	80	84	88	92	96	100	104
Height(MB)	45	47	49	51	54	56	58

720p+

Overscan (%)	0	5	10	15	20	25	30
Width (pix)	640	672	704	736	768	800	832
Height (pix)	480	496	528	544	576	592	624
A:R	1.333333	1.354839	1.333333	1.352941	1.333333	1.351351	1.333333
Width(MB)	40	42	44	46	48	50	52
Height(MB)	30	31	33	34	36	37	39

VGA+

Overscan (%)	0	5	10	15	20	25	30
Width (pix)	352	368	384	400	416	432	448
Height (pix)	288	288	304	320	336	352	368
A:R	1.222222	1.277778	1.263158	1.25	1.238095	1.227273	1.217391
Width(MB)	22	23	24	25	26	27	28
Height(MB)	18	18	19	20	21	22	23

CIF+

Overscan (%)	0	5	10	15	20	25	30
Width (pix)	176	176	192	192	208	208	224
Height (pix)	144	144	144	160	160	176	176
A:R	1.222222	1.222222	1.333333	1.2	1.3	1.181818	1.272727
Width(MB)	11	11	12	12	13	13	14
Height(MB)	9	9	9	10	10	11	11

QCIF+

Overscan (%)	0	5	10	15	20	25	30
Width (pix)	960	1008	1056	1104	1152	1200	1248
Height (pix)	540	560	592	608	640	672	688
A:R	1.777778	1.8	1.783784	1.815789	1.8	1.785714	1.813953
Width(MB)	60	63	66	69	72	75	78
Height(MB)	33.75	35	37	38	40	42	43

qHD+