Application Note

How to use phyCAM camera modules with phyCORE-i.MX6UL SBC

Revision History

Version	Changes	Author	Date
A0	Initial Release	H. Fendrich	10.04.2017
A1	New Release PD19.1.0 with added VM-011 and VM-009	H. Fendrich	04.07.2019
A2	PD21.2.0 changes added	H. Fendrich	02.09.2021

Content

1 Overview

The i.MX6UL Microcontroller supported 1 camera interface (see figure 1).

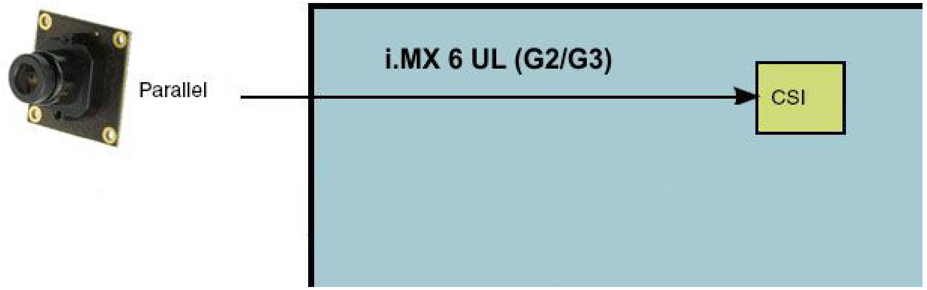

figure 1: Block Diagram Camera Interfaces of i.MX6UL Controller (G2 / G3)

On the phyCORE-i.MX6UL the CSI camera path go out as 10-Bit parallel signal.

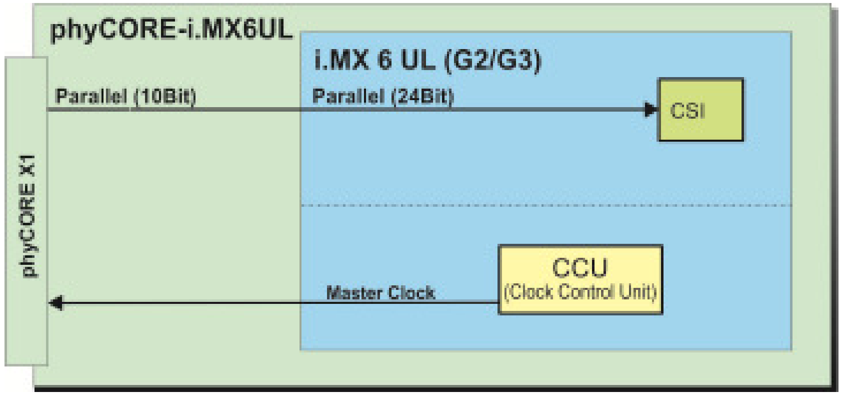

figure 2: Use of Parallel Camera (CSI)

On the PHYTEC or customer carrier boards can the interfaces are led out as phyCAM-P (or raw-parallel) see figure 3 and/or phyCAM-S+ see figure 4. For more information to phyCAM-P/S+ see manual L-748.

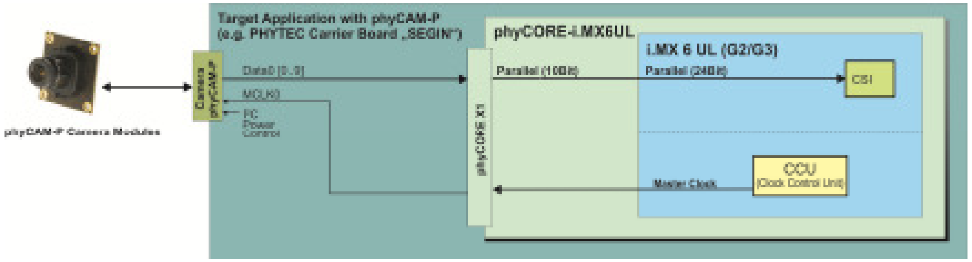

figure 3: Block Diagram of an phyCAM-P solution with phyCORE-i.MX6UL (G2 / G3) and the go out on the phyBOARD-SEGIN-i.MX6UL - SBC

On the phyBOARD-Mira baseboard is only the CSI0/IPU#1 camera path (thru a serializer) go out as LVDS signal (see figure 4).
Here you can connect one of the different phyCAM-S(+) camera modules. See the phyCAM-P/-S manual L-748 for more information.

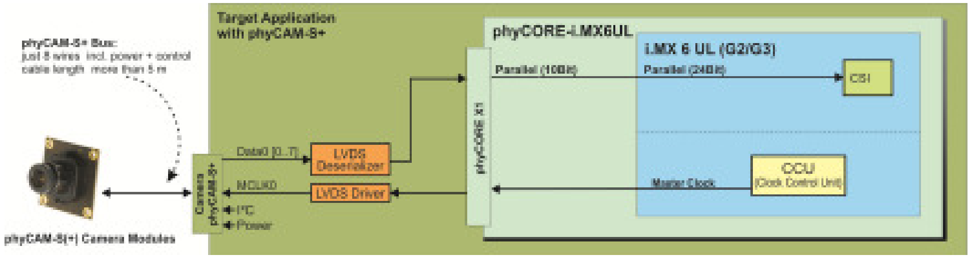

figure 4: Block Diagram of phyCAM-S(+) solution with phyCORE-i.MX6UL (G2 / G3) SBC

The BSP shipped with the Kit includes already the software drivers for the supported phyCAM camera modules. The drivers are compatible with v4l2. Also GStreamer scripts are included for the evaluation of the camera modules. If you need more then one camera interface or/and phyCAM-S+ support, please use the phyFLEX-i.MX6 or phyCORE-i.MX6 Modul.

Note:
The phyBOARD-SEGIN features one CSI interface for phyCAM-P camera modules.
Please find more information about the camera support in the path: …\Documentation\...

The kit version KPB-02013-Alpha-L.A0 is shipped with an i.MX6 controller board which has camera support installed.

The Linux BSP for these kits does support the phyCAM-P camera series from PHYTEC.
https://download.phytec.de/ImageProcessing/

The table below lists the options:

Hardware Configuration		Software Support
phyBOARD Kit	**phyCAM-P at CSI**	**phyCAM-P at CSI**
KPB-02013-Video-L01	Yes (VM-010-BW-M12 is included)	yes
KPB-02013-Video-L02	Yes (phyCAM-P camera is selectable)	yes

NM = no mount

2 Camera Connectors on the SEGIN - Carrier Boards

The development kits for the phyBOARD-SERGIN-i.MX6 contain:

- one carrier board (SERGIN)
- one phyCORE-i.MX6UL modul SOM

The phyCORE-i.MX6UL SOM is tightly soldered with the carrier board.

On the base board SERGIN (PB-02013) we led out the parallel camera interface on the phyCAM-P camera interface standard (see figure 3). The phyCAM-P interface is support on connector X12 (33pol. FFC). Information's to phyCAM-P are in the phyCAM-manual (L-748).

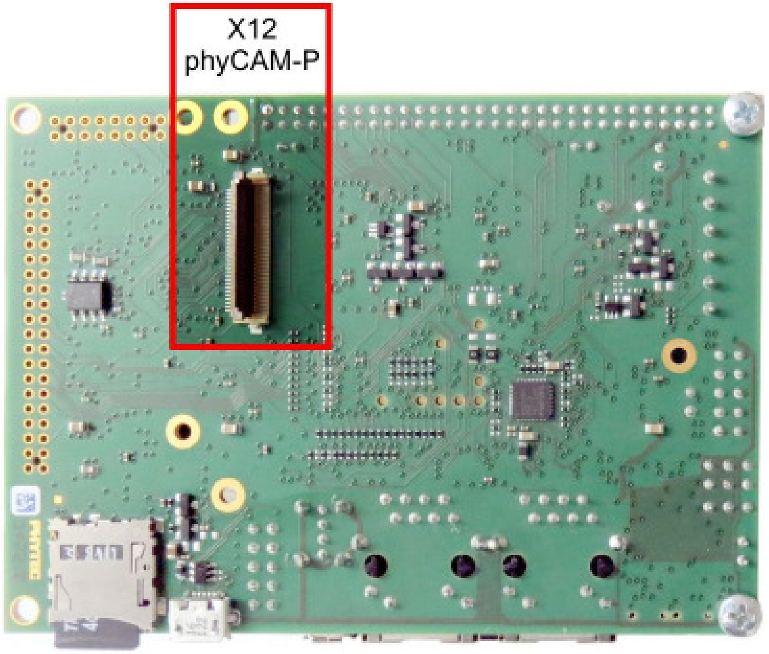

figure 5: Camera Interfaces on phyBOARD-SERGIN for the phyCORE-i.MX6UL SOM

Connecting the phyCAM-P Camera:

- Open the lock of the FFC connector on the camera by lifting the lock upwards.
- Plug the FFC cable into the FFC socket with the contact surfaces facing downwards until you feel the stop. The reinforcement of the FFC cable (usually highlighted in color) points to the bracket of the socket.
- Lock the FFC socket by carefully pressing down on the bracket.

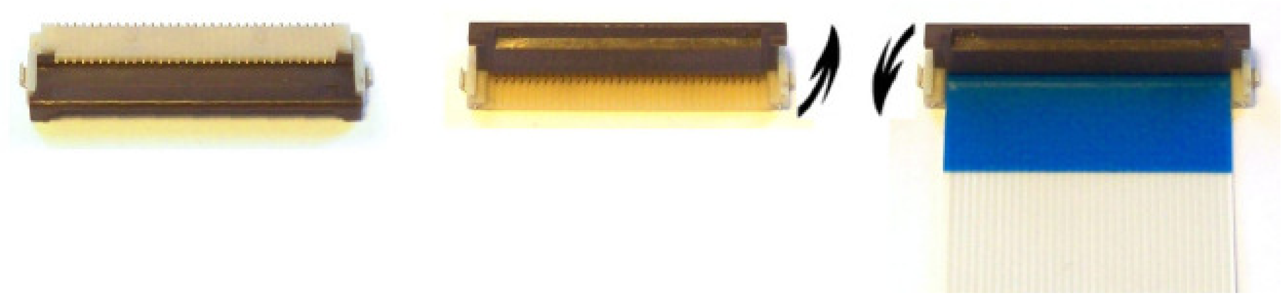

figure 6: phyCAM-P Flip Lock – FFC Socket Camera Connection

3 Change the cameras or camera parameters

3.1 Change the cameras or camera parameters (YOCTO Linux PD16.1.0 and higher)

At the moment is only the VM-010 camera series supported. The driver is compiled in the BSP and will be loaded at the boot sequence.
You can only change the color type of the camera between "color" and "monochrome".

If you need a higher resolution or more than one camera, we recommend using a phyCORE-i.MX6 or phyFLEX-i.MX6 SOM.

For more information see on FTP:
ftp://ftp.phytec.de/pub/ImageProcessing/phyBOARD-Segin-i.MX6UL_linux_PD17.2.0/

The configuration in the config-expansions file must match with the camera models that are connected to the camera interfaces.
Note that model, interface type and I²C-addresses must be set correctly. Otherwise the camera(s) are not working.
Please check, if the correct camera model is set in the config file. If not, please follow the steps below to set the appropriate configuration.

Changing the setting of the camera-bootarg parameters

Depending on the camera there has to be set a corresponding entry in the bootloader Barebox.
At the moment you can only change between the -COL (color) and –BW (monochrome) versions of VM-010.
To change the device-tree use the Barebox environment:
- type 'nv linux.bootargs.mt9v022="mt9v022.sensor_type=color"' <enter>, for color version
- type 'nv linux.bootargs.mt9v022="mt9v022.sensor_type=mono"' <enter>, for monochrome version
- type 'saveenv' <enter>
- type 'reset' <enter>

After login, change into the directory: \gstreamer_examples\..
cd gstreamer_examples <ENTER>.

Now you can start working with the GStreamer demo-scripts. Detailed information about the GStreamer examples can be found in the phyCAM-Manual L-748.

Hardware Configuration		Bootarg Parameters		Bootarg settings for default I²C-address (default jumper on camera and boards)
phyCAM camera model (part number)	**connected to**	**csi_cam_bus_type**	**csi_cam_i2c_address**	
VM-010-BW (-M12 / -H)	**X12 on SERGIN board**	phyCAM-P	0x48 default (0x4C, 0x58, 0x5C possible)	nv linux.bootargs.mt9v022="mt9v022.sensor_type=color"
VM-010-COL (-M12 / -H)	**X12 on SERGIN board**	phyCAM-P	0x48 default (0x4C, 0x58, 0x5C possible)	nv linux.bootargs.mt9v022="mt9v022.sensor_type=mono"

Notes:

- csi_cam address must be identical to the I²C-address of the corresponding camera. I²C addresses of the camera are set by hardware configuration (jumper setting on the camera and / or on the baseboard / mapper board. Please refer to the phyCAM-manual L-748 and the hardware manual of the kit. For change the I²C-Address in the BSP the device-tree have to change and the BSP you have to compile the BSP new.

3.2 Change the cameras or camera parameters (YOCTO Linux PD19.1.0 and higher)

At the moment are the VM-010, VM-011 and VM-009 camera series supported. The drivers is compiled in the BSP and will be loaded at the boot sequence.
You can change the cameras and the color type of the camera between "color" and "monochrome".

If you need more processor power or more than one camera, we recommend using a phyCORE-i.MX6 or phyFLEX-i.MX6 SOM.

For more information see on FTP:
ftp://ftp.phytec.de/pub/ImageProcessing/phyBOARD-Segin-i.MX6UL_linux_PD19.1.0/

The configuration in the config-expansions file must match with the camera models that are connected to the camera interfaces.
Note that model, interface type and I²C-addresses must be set correctly. Otherwise the camera(s) are not working.
Please check, if the correct camera model is set in the config file. If not, please follow the steps below to set the appropriate configuration.

Changing the setting of the config-expansions parameters

To change the config-file (config-expansions), use the Barebox environment: Use a terminal program e.g. "Putty".

- 'cd env' <enter>
- 'edit config-expansions' <enter>

The following settings in config-expansions - file are necessary:
<cam_type[color_format]>

- cam_type[color_format] = [VM-010-BW, VM-010-COL, VM-011-BW, VM-011-COL, VM-009]

Example: VM-010-COL:
[. /env/expansions/imx6ul-phytec-vm010-col]

The parameters for the phyCAM – modules are shown in the table below.

After changing the settings with the editor:

- close the editor (CTRL D)
- type 'saveenv' <enter> to save
- restart PHYTEC module

After login, change into the directory: \gstreamer_examples\..
cd gstreamer_examples <ENTER>.

Now you can start working with the GStreamer demo-scripts. Detailed information about the GStreamer examples can be found in the phyCAM-Manual L-748.

Hardware Configuration		**Bootarg Parameters**		**Bootarg settings for default I^{2}C-address (default jumper on camera and boards)**
phyCAM camera model (part number)	**connected to**	**csi_cam_bus_type**	**csi_cam_i2c_address**	
VM-010-BW (-M12 / -H)	**X12 on SEGIN board**	phyCAM-P	0x48 default (0x4C, 0x58, 0x5C possible)	. /env/expansions/imx6ul-phytec-vm010-bw
VM-010-COL (-M12 / -H)	**X12 on SEGIN board**	phyCAM-P	0x48 default (0x4C, 0x58, 0x5C possible)	. /env/expansions/imx6ul-phytec-vm010-col
VM-011-BW (-M12 / -H)	**X12 on SEGIN board**	phyCAM-P	0x48 default (0x5D possible)	. /env/expansions/imx6ul-phytec-vm011-bw
VM-011-COL (-M12 / -H)	**X12 on SEGIN board**	phyCAM-P	0x48 default (0x5D possible)	. /env/expansions/imx6ul-phytec-vm011-col
VM-009 (-M12 / -H)	**X12 on SEGIN board**	phyCAM-P	0x48 default (0x5D possible)	. /env/expansions/imx6ul-phytec-vm009

Notes:

- csi_cam address must be identical to the I^{2}C-address of the corresponding camera. I^{2}C addresses of the camera are set by hardware configuration (jumper setting on the camera and / or on the baseboard / mapper board. Please refer to the phyCAM-manual L-748 and the hardware manual of the kit. For change the I^{2}C-Address in the BSP the device-tree have to change and the BSP you have to compile the BSP new.

3.3 Change the cameras or camera parameters (YOCTO Linux PD21.2.0 and higher)

At the moment are the VM-009, VM-010, VM-011 and VM-016 camera series supported. The drivers is compiled in the BSP and will be loaded at the boot sequence.
You can change the cameras and the color type of the camera between "color" and "monochrome".

If you need more processor power or more than one camera, we recommend using a phyCORE-i.MX6 or phyCORE-i.MX8 - series

For more information see on FTP:
https://download.phytec.de/ImageProcessing/phyBOARD-Segin-i.MX6UL_linux_PD21.2.x/

The old camera configuration via "config-expansions" file has been changed. The configuration is now in the barebox.
The configuration in the barebox must match with the camera models that are connected to the camera interfaces.
Note that model, interface type and I^{2}C-addresses must be set correctly. Otherwise the camera(s) are not working.
Please check, if the correct camera model is set in the barebox. If not, please follow the steps below to set the appropriate configuration.

Changing the setting of the camera parameters

To change the camera, use the Barebox environment: Use a terminal program e.g. "Putty".

- Press space button during the barebox count. "Hit m for menu or any to stop autoboot: x"

Now you have to configure the right devicetree for your camera and camera interface.

Example: VM-010-BW-LVDS with i2C-address 0x48 on CSI0 (Camera_0) port

- type ‘nv overlays.select="imx6-vm010-bw-0.dtbo"’ <enter>

The parameters for the phyCAM – modules are shown in the table below.

After changing the settings with the editor:

- type 'saveenv' <enter> to save
- type 'reset' <enter> to restart

Check the set camera in the first rows of boot log:
e.g.: “Add /mnt/mmc0.1/overlays/imx6-vm010-bw-0.dtbo overlay”

Hardware Configuration		Bootarg Parameters			Device Tree settings for default I^{2}C-address (default jumper on camera and boards)	
phyCAM camera model (part number)	connected to	phyCAM Type	Cam Type	CSI Port i2c Address		
VM-009 (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-009	0x48, 0x5D	nv overlays.select="imx6ul-vm009.dtbo"	0x48
VM-010-BW (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-010-BW	0x48, 0x4C, 0x58, 0x5C	nv overlays.select="imx6ul-vm010-bw.dtbo"	0x48
VM-010-COL (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-010-COL	0x48, 0x4C, 0x58, 0x5C	nv overlays.select="imx6ul-vm010-col.dtbo"	0x48
VM-011-BW (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-011-BW	0x48,0x5D	nv overlays.select="imx6ul-vm011-bw.dtbo"	0x48
VM-011-COL (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-011-COL	0x48,0x5D	nv overlays.select="imx6ul-vm011-col.dtbo"	0x48
VM-016-BW-P (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-016	0x10, 0x18	nv overlays.select="imx6ul-vm016.dtbo"	0x10
VM-016-COL-P (-M12 / -H)	X12 on SEGIN board	phyCAM-P	VM-016	0x10, 0x18	nv overlays.select="imx6ul-vm016.dtbo"	0x10

4 Demo scripts

If you want to see the live images, we recommend using a HDMI display/monitor on the Nunki Board.

There are 3 sub directories with demo scripts for the cameras:

- gstreamer-examples
- v4l2_c-examples

Notes: Remove the qt-demo at the first start. With the \gstreamer-examples\remove_qt_demo.sh script, else the qt-demo is alway present on the display.

4.1 GStreamer scripte

After login, change into the directory: \gstreamer_examples\..
cd gstreamer_examples <ENTER>.

At the first start disable the QT-Demo. Start the script “remove_qt_demo.sh”.

Now you can start working with the GStreamer demo-scripts.
Start the scripts with den word phrase "col" or "bw" depending on the connected camera color type.

- _cam-fbdev_640x480.sh – scripts, show a livestream on display
- _cam-save_jpg_full_res.sh – scripts, save a JPG File in this directory
- _cam-save_raw_full_res.sh – scripts, save a RAW File in this directory
- func.sh – script, detect the camera typ and define the parameter for the scripts
- remove_qt_demo.sh – script, remove the qt-demo from autostart

Subdirectories:

- more_ar0144_scripts: contain more scripts for the VM-016 camera series (v4l-ctrl_ar0144.txt, list the v4l2-controls for this camera)
- more_mt9m131_scripts: contain more scripts for the VM-009 camera series (v4l-ctrl_mt9m131.txt, list the v4l2-controls for this camera)
- more_mt9p031_scripts: contain more scripts for the VM-011 camera series (v4l-ctrl_mt9p031_mt9p006.txt, list the v4l2-controls for this camera)
- more_mt9v024_scripts: contain more scripts for the VM-010 camera series (v4l-ctrl_mt4v024.txt, list the v4l2-controls for this camera)
- phytec_usb_cam: scripts for use the Phytec USB-cameras
- tools: contain scripts to get and set the camera register direct via i2c access

All camera/video components get a separate "/dev/video[x]" or "dev/v4l-subdev[x]" device.
The v4l2 - capabilities are showed if you type: "v4l2-ctl -d [device] -L" e.g. "v4l2-ctl -d /dev/v4l-subdev1 -L".

For the first use, the camera and controller-camera interface must be configured with the tool media-ctl.

4.2 Scripts to call C/C++ files based on v4l2 interface

After login, change into the directory: \v4l2_c-examples\..
cd v4l2_c-examples <ENTER>.

Now you can start working with the demo-scripts. Start the scripts with den word phrase "col" or "bw" depending on the connected camera color type.

- ar0144_col_full_save-raw – scripts, save a raw image from VM-016-COL in full resolution
- ar0144_bw_full_save-raw – scripts, save a raw image from VM-016-BW in full resolution
- mt9m131_col_full_save-raw – scripts, save a raw image from VM-009 in full resolution
- mt9p006_col_full_save-raw – scripts, save a raw image from VM-011-COL in full resolution
- mt9p031_bw_full_save-raw – scripts, save a raw image from VM-011-BW in full resolution
- mt9v02x_col/bw_full_save-raw – scripts, save a raw image from VM-010 in full resolution
- vita1300_col/bw_full_save-raw – scripts, save a raw image from VM-012 in full resolution
- vm05x_full_save-raw – scripts, save a raw image from VM-050/51 in full resolution

For saving the image we use the program Yavta. Yavta stands for "Yet Another V4L2 Test Application". This is a test application based on V4L2 Linux interface.

A other way to use in your C-program is the direct access call to the v4l2 interface. For example:

- *v4l2-ctl –d1 --stream-mmap --stream-count 1 --stream-to=raw_image1.raw*

For the first use, the camera and controller-camera interface must be configured with the tool media-ctl.

5 Configuration of camera features

5.1 Configuration with v4l2-ctl

To set the various camera functions (e.g. exposure, gain, ...) use please the v4l2-ctl functions. You can get an overview of the existing functions by entering the commands:

- for example: v4l2-ctl -d /dev/v4l-subdev1 --all (list all)
- for example: v4l2-ctl -d /dev/ v4l-subdev1 -L (list all detail)

With this control are many features usable. For example exposure. Set one exposure value (if the automatic disable):

- v4l2-ctl -d /dev/ v4l-subdev1 --set-ctrl=auto_exposure=1 (set AEC off)
- v4l2-ctl -d /dev/ v4l-subdev1 --set-ctrl=exposure=480 (set the exposure time for a time, that the sensor need to generate 480 rows)

5.2 Configuration camera register direct

To set or get a camera register direct use the i2c functions in the path …/gstreamer-examples/tools/...

Note: Use this function only, if you know the register reaction. Read the register reference manual of the camera sensor manufacturer.

6 GStreamer function for improved Tearing reduction

Tearing is a visual artifact in video display where the display image shows information from two or more frames in a single screen draw.
It appears as a horizontal border or line. The image part beyond the line seems to be shifted horizontally when the image content or the camera is moved.
The tearing line(s) usually move(s) vertically across the image.
The artifact occurs when the camera framerate differs from the display frame or the camera readout cycle is not in sync with the display's refresh.
Definition and background information about the tearing effect can be found in http://en.wikipedia.org/wiki/Screen_tearing.

The Freescale i.MX-6 processor contains an anti-tearing mechanism in the IPU unit, which can reduce the tearing effect.
However, since - depending on camera settings and camera model - the frame rate might be very different from the display’s refresh rate, tearing effects might still be visible even if the anti-tearing mechanism is active. For applications that are intended to display live camera images on the display, additional measures should be considered to obtain a perfect image quality. This measures can include frame rate control (trimming the camera frame rate to the display’s refresh rate), multi-buffering of the camera image etc.

We recommend to activate the anti-tearing mechanism of the i.MX-6 when live camera images are shown on the display. For evaluation purposes with the development kits, Phytec added GStreamer examples, that use a different fbsink – function, that activates the anti-tearing mechanism.

6.1 GStreamer function for improved Tearing reduction (Linux PD16.1.0)

From PD16.1.0 up to PD17.1.1 the GStreamer contains the " imxpxpvideosink use-vsync=tru" plugin.
This plugin reduce the tearing. Most of the Phytec GStreamer example scripts use this plugin. Please show the scripts in path ../gstreamer_examples/.. .
For more information to "imxpxpvideosink" parameters type:
- gst-inspect-1.0 imxpxpvideosink

7 De-Bayering (demosaicking) with NEON CoProcessor

Most of CMOS color chips provide the image in the bayer mosaicing (bayer raw) format. For get a color image in RGB format is it necessary to convert the bayer raw image.
- https://en.wikipedia.org/wiki/Bayer_filter
- https://de.wikipedia.org/wiki/Bayer-Sensor

There are exist different algorithm for converting.
- https://en.wikipedia.org/wiki/Demosaicing

If the microprocessor does not include debayering hardware, have to do the converting via software. For this you need additional processing power and the framerate goes down. It is better to use the NEON coprocessor of the i.MX6. For this support PHYTEC a special function.
It is present as GStreamer plugin "bayer2rgbneon" and in sources for use in an own C-program. We support a simple bilinear algorithm.

For use in GStreamer take "bayer2rgbneon" plugin. For more information to " bayer2rgbneon" parameters type:
- gst-inspect-1.0 bayer2rgbneon