Get the CSI BT656 without HSYNC/VSYNC working.

The problem, is that all implemented driver in the BSP are using the external HSYNC/VSYNC synchronization signal.

The SW designer is actually struggling to find the right way to generate the end of field active interrupt (end of frame).

> mxc_v4l2_still.out I get an:

> ERROR: v4l2 capture: mxc_v4l_read timeout counter 0

> When using mxc_v4l2_capture.out or mxc_v4l2_tvin.out I get an:

> ERROR: v4l2 capture: mxc_v4l_dqueue timeout enc_counter 0

To help him implementing the driver, I need to get some insight on the IPUx_CSIn_CCIR_CODE_1/2/3 as it seems that the bit description 21 to 19 (CSI_STRT_FLD0_ACTV) is not described (same for all the multiple bit field in this register.

Maybe it should match the ITU656, but here as well the driver examples does not match the bit description of the ITU standard.

So my questions:

IS it really possible to implement the BT656 with SAV / EAV and without external HSYNC/VSYNC?

If yes, is it possible to review the IPUx_CSIn_CCIR_CODE_1/2/3 fields and communicate which parameter should be provided here?

Reply-------------------------------

For no VSYNC and HSYNC case, in the sensor driver such as "linux-3.0.35\drivers\media\video\mxc\capture\adv7180.c", function ioctl_g_ifparm(), you should set p->u.bt656.bt_sync_correct to 0;
p->u.bt656.bt_sync_correct = 1; // It means external VSYNC and HSYNC will be used for SYNC.
p->u.bt656.bt_sync_correct = 0; // No external VSYNC and HSYNC, embedded EAV and SAV will be used for SYNC.

In iMX6 BSP, the CCIR related code is ready in "linux-3.0.35\drivers\mxc\ipu3\ipu_capture.c", function ipu_csi_init_interface(), no code modification was needed, that code was verified work.

For BT656 mode, your sensor driver such as adv7180, should also report correct parameters in function function ioctl_g_ifparm().
p->u.bt656.clock_curr = 0; // This will tell linux-3.0.35\drivers\media\video\mxc\capture\mxc_v4l2_capture.c to use "IPU_CSI_CLK_MODE_CCIR656_INTERLACED" in function mxc_v4l2_s_param().

The current iMX6 BSP mxc_v4l2_capture.c driver doesn't support BT656 progressive mode, it only supports BT656 interlace mode. To support BT656 progressive mode, the customer should modify the code in mxc_v4l2_s_param(), let csi_param.clk_mode = IPU_CSI_CLK_MODE_CCIR656_PROGRESSIVE.

This is exactly what they are doing, but still it doesn’t work.

Actually, they see the code being executed following the right steps in the ipu_csi_init_interface() going through PAL 720x625 configuration.

But still, they get the timeout!

```
else if (cfg_param.clk_mode == IPU_CSI_CLK_MODE_CCIR656_INTERLACED) {

    if (width == 720 && height == 625) {

      /* PAL case */

      /*

Field0BlankEnd = 0x6, Field0BlankStart = 0x2,

Field0ActiveEnd = 0x4, Field0ActiveStart = 0

    */

    ipu_csi_write(ipu, csi, 0x40596, CSI_CCIR_CODE_1);

    /*

                    Field1BlankEnd = 0x7, Field1BlankStart = 0x3,

                    Field1ActiveEnd = 0x5, Field1ActiveStart = 0x1

       */

     ipu_csi_write(ipu, csi, 0xD07DF, CSI_CCIR_CODE_2);

      ipu_csi_write(ipu, csi, 0xFF0000, CSI_CCIR_CODE_3);
```

So, I believe the parameters are wrong.

What could be missing. Can you review the driver?

For me it looks like the adv example, except the bt_sync_correct=0, which is what we want.

Reply--------------------------------------

- to capture the CSI data bus to check if there is correct output from sensor, such as EAV/SAV. For the "timeout" error, it always means there is no correct data on CSI data bus.