V1.5

GSV2011 Application Guide

2020-4-25

GSCOOLINK

(This page is intentionally left blank)

CONTENTS

1. UDP Mode Description

1.1 Design Purpose

The purpose of GSV2011 UDP mode is to transmit lossless HDMI video/audio content between HDMI interface and a uniform minimized parallel interface. GSV2011 can support all the video/audio formats within HDMI 2.0b definition in UDP mode. GSV2011 supports TTL/LVDS as input/output in UDP mode for HDMI 2.0b timings. In the HDMI to Parallel bus UDP application mode, HDMI input's HDCP1.4/2.3 encryption will be decrypted by GSV2011. Then its raw Video/Audio/Control Packets are time division multiplexed to a group of pairs of parallel bus. A pair of dedicated clock pins are used for LVDS transaction between GSV2011 and external chip. A single pin clock is used for TTL transaction between GSV2011 and external chip.

1.2 HDMI to Parallel Bus Pin Mapping

For parallel bus, A 36-bit logical data bus and 1 separate DE pin is mapped from/to the HDMI interface. The allocation of the bits are listed below.

Name	Description	Bit Map[35:0]	Category
video_de	Video Data Enable	Separate Pin	
video_data[35:0]	Valid Video Pixel Data	BUS[35:0]	Valid When video_de=1
hdmi_ctl[2:0]	Includes: pkt_lgb/ pkt_tgb/ pkt_pre/ video/ lgb/ video_pre	BUS [23:21]	Valid When video_de=0
vs	Vertical Sync	BUS [19]	
hs	Horizontal Sync	BUS [18]	
pkt_de	Packet Data Enable	BUS [17]	
pkt_data[8]	Packet Data	BUS [16]	
pkt_data[7:0]	Packet Data	BUS [11:4]	

For parallel bus, when DE = 1, 36 bits are fully occupied by active video data. When DE = 0, BUS [35:24], BUS [20], BUS [15:12] and BUS [3:0] are 0s, the rest bits are the valid control signals and audio data.

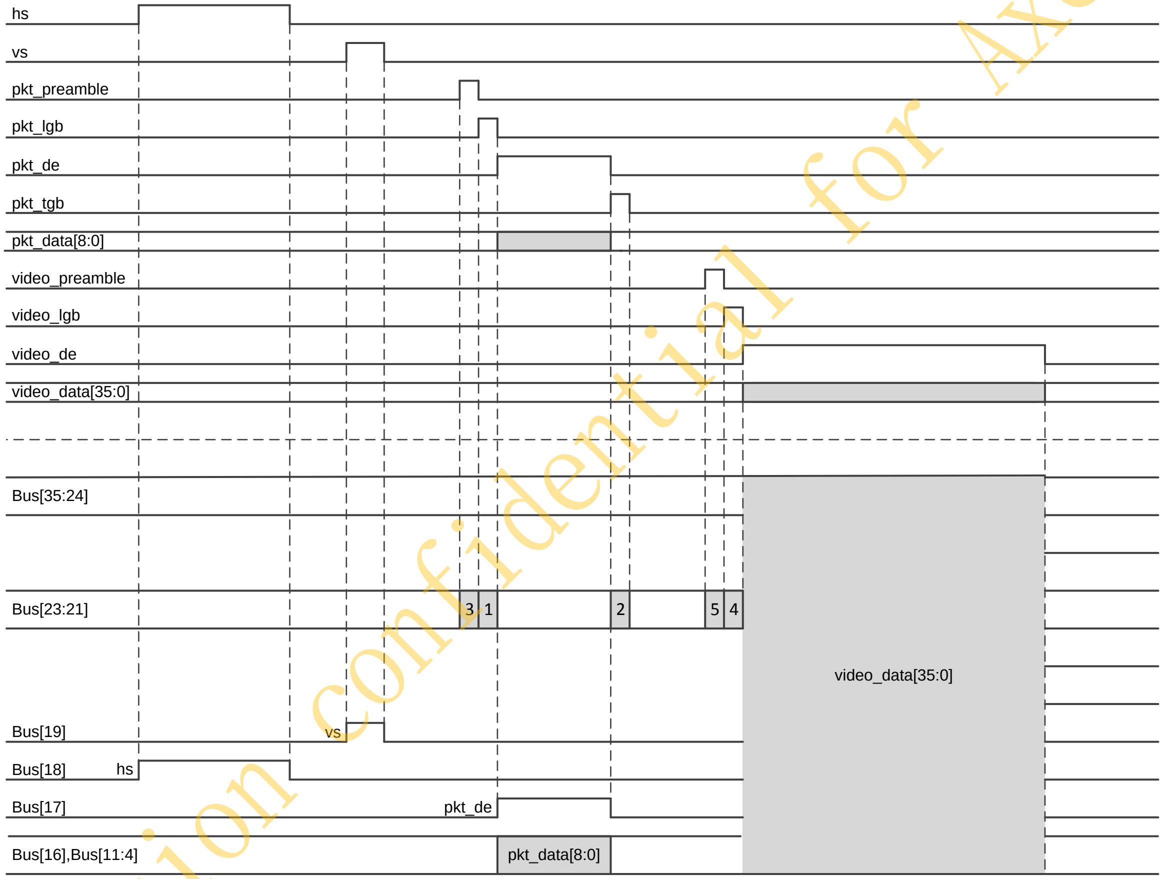
The mapping between HDMI_CTL and pkt_lgb/ pkt_tgb/ pkt_preamble/ video/ lgb/ video_preamble are listed below:

hdmi_ctl[2:0]	Controls
1	pkt_lgb
2	pkt_tgb
3	pkt_preamble
4	video_lgb
5	video_preamble

The parallel bus clock has the capability to be set in SDR/DDR mode according to the external receiver’s capability.

If external transmitter needs to send UDP mode parallel bus to GSV2011 for HDMI transmission. It should strictly follow HDMI specification on all the signals listed in the above table. GSV2011 TX digital core will build the HDMI output stream with the raw information from parallel bus, HDCP 1.4/2.3 encryption could to enabled by GSV2011 on the HDMI TX port as well.

The detailed example mapping is given in below picture. In the picture, each channel is shown as 12-bit, but in current UDP application, each channel is mapped using higher 8-bit (lower 4-bit is ignored).



1.3 Packet Information mapping

UDP Packet information is provided on BCH level.

Following HDMI 1.4b specification Figure 5-3, packet data is allocated on 9 positions on Channel 0/1/2. Data Island Period active bits are:

TMDS Channel 0 D[2] = Packet Header
TMDS Channel 1 D[3:0]
TMDS Channel 2 D[3:0]

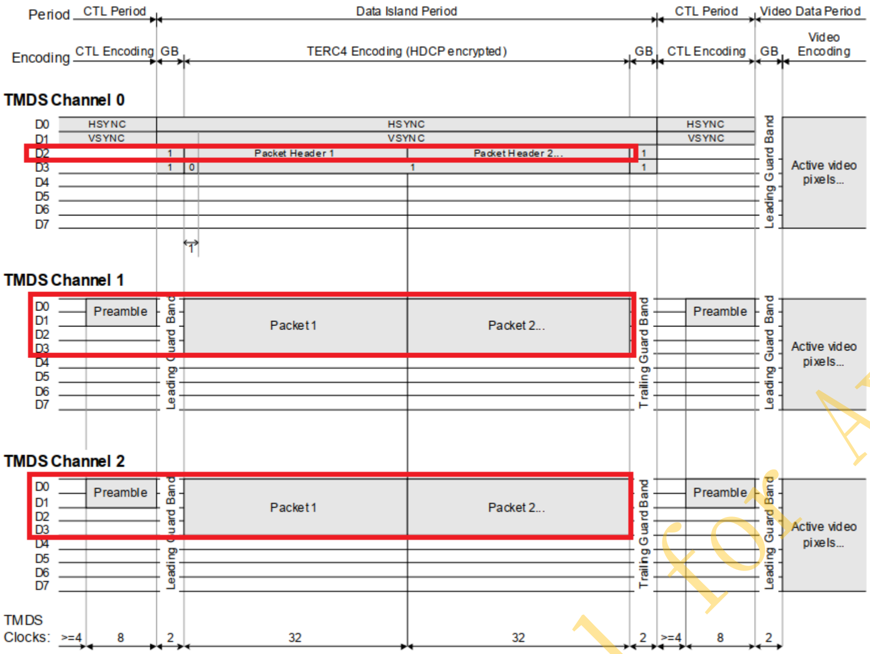


Figure 5-3 TMDS Periods and Encoding

The channel and BCH block mapping is shown in Figure 5-4 of HDMI 1.4b specification.

Channel 0 D2	BCH block 4 (Packet Header)
Channel 1 D0 and Channel 2 D0	BCH block 0
Channel 1 D1 and Channel 2 D1	BCH block 1
Channel 1 D2 and Channel 2 D2	BCH block 2
Channel 1 D3 and Channel 2 D3	BCH block 3

And the 9-bit UDP pkt_data is mapped with Figure 5-3/5-4 with following table.

Pkt_data	TMDS Channel	D	Figure 5-4 Designation	BCH block
PKT[8]	2	D[3]	C3	3
PKT[7]	2	D[2]	C2	2
PKT[6]	2	D[1]	C1	1
PKT[5]	2	D[0]	C0	0
PKT[4]	1	D[3]	B3	3
PKT[3]	1	D[2]	B2	2
PKT[2]	1	D[1]	B1	1
PKT[1]	1	D[0]	B0	0
PKT[0]	0	D[2]	A2	4

1.4 HDMI to LVDS in UDP mode

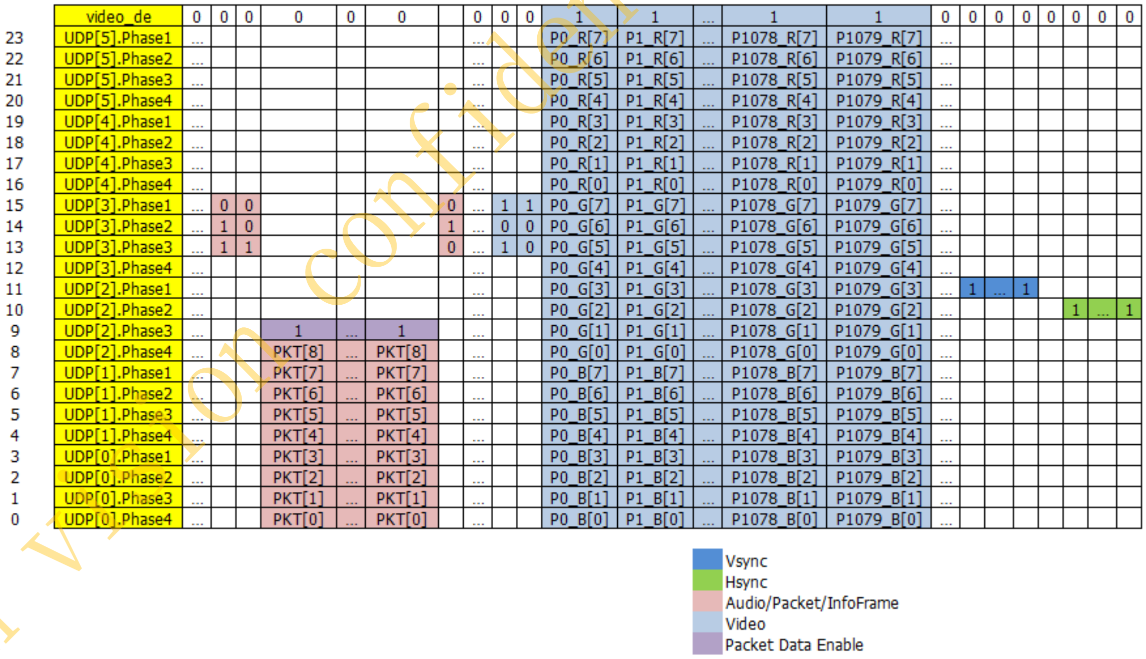
In this HDMI to LVDS UDP application mode, HDMI input’s raw Video/Audio/Control Packets are time division multiplexed to a group of 9 pairs of LVDS differential pins in

single pixel mode or 12 pairs of LVDS differential pins in dual pixel mode. An extra pair of LVDS differential pins is dedicated to transmit DE signal.
Single pixel LVDS x4 mode is used for <=300MHz pixel clock timing (4K30Hz 12-bit in maximum bandwidth), refer to Table 25 LVDS 4x Single Pixel YCbCr/RGB 4:4:4 Pin Mapping in <GSV2011 datasheet> for the 9 pairs of pins. The maximum LVDS data lane frequency is 1.2Gbps in this mode.
Dual pixel LVDS x4 mode should be used for >300MHz pixel clock timing (4K60Hz 8-bit in maximum bandwidth), please refer to Table 27 LVDS 4x Dual Pixel YCbCr/RGB 4:4:4 Pin Mapping in <GSV2011 datasheet> for the 12 pairs of pins. The maximum LVDS data lane frequency is 1.2Gbps in this mode.

By default, the external receiver should receive these pairs of LVDS differential pairs of data and a pair of input LVDS clock. The external receiver should use the given LVDS clock to generate its own sampling clock, then utilizes the pair of DE to decode the rest pairs of data into Video/Audio/Control Packets.
The decoding of DE is critical for separating active video pixel data and other valid packets. When DE = 1, video data is present on the differential pairs. When DE = 0, audio/control packets are present. The DE pin behavior follows the active pixel data enable definition of video timings defined by CEA/VESA.

Typical LVDS UDP mode pin mapping diagrams are shown below.

UDP mode Single Pixel 4x LVDS pin mapping demo (RGB 8-bit, 1080p@60Hz)



	video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
23	UDP[5].Phase1	...							...			P0_R[7]	P1_R[7]	...	P1078_R[7]	P1079_R[7]	...							
22	UDP[5].Phase2	...							...			P0_R[6]	P1_R[6]	...	P1078_R[6]	P1079_R[6]	...							
21	UDP[5].Phase3	...							...			P0_R[5]	P1_R[5]	...	P1078_R[5]	P1079_R[5]	...							
20	UDP[5].Phase4	...							...			P0_R[4]	P1_R[4]	...	P1078_R[4]	P1079_R[4]	...							
19	UDP[4].Phase1	...							...			P0_R[3]	P1_R[3]	...	P1078_R[3]	P1079_R[3]	...							
18	UDP[4].Phase2	...							...			P0_R[2]	P1_R[2]	...	P1078_R[2]	P1079_R[2]	...							
17	UDP[4].Phase3	...							...			P0_R[1]	P1_R[1]	...	P1078_R[1]	P1079_R[1]	...							
16	UDP[4].Phase4	...							...			P0_R[0]	P1_R[0]	...	P1078_R[0]	P1079_R[0]	...							
15	UDP[3].Phase1	...	0	0				0	...	1	1	P0_G[7]	P1_G[7]	...	P1078_G[7]	P1079_G[7]	...							
14	UDP[3].Phase2	...	1	0				1	...	0	0	P0_G[6]	P1_G[6]	...	P1078_G[6]	P1079_G[6]	...							
13	UDP[3].Phase3	...	1	1				0	...	1	0	P0_G[5]	P1_G[5]	...	P1078_G[5]	P1079_G[5]	...							
12	UDP[3].Phase4	...							...			P0_G[4]	P1_G[4]	...	P1078_G[4]	P1079_G[4]	...							
11	UDP[2].Phase1	...							...			P0_G[3]	P1_G[3]	...	P1078_G[3]	P1079_G[3]	...	1	...	1				
10	UDP[2].Phase2	...							...			P0_G[2]	P1_G[2]	...	P1078_G[2]	P1079_G[2]	...					1	...	1
9	UDP[2].Phase3	...			1	...	1		...			P0_G[1]	P1_G[1]	...	P1078_G[1]	P1079_G[1]	...							
8	UDP[2].Phase4	...			PKT[8]	...	PKT[8]		...			P0_G[0]	P1_G[0]	...	P1078_G[0]	P1079_G[0]	...							
7	UDP[1].Phase1	...			PKT[7]	...	PKT[7]		...			P0_B[7]	P1_B[7]	...	P1078_B[7]	P1079_B[7]	...							
6	UDP[1].Phase2	...			PKT[6]	...	PKT[6]		...			P0_B[6]	P1_B[6]	...	P1078_B[6]	P1079_B[6]	...							
5	UDP[1].Phase3	...			PKT[5]	...	PKT[5]		...			P0_B[5]	P1_B[5]	...	P1078_B[5]	P1079_B[5]	...							
4	UDP[1].Phase4	...			PKT[4]	...	PKT[4]		...			P0_B[4]	P1_B[4]	...	P1078_B[4]	P1079_B[4]	...							
3	UDP[0].Phase1	...			PKT[3]	...	PKT[3]		...			P0_B[3]	P1_B[3]	...	P1078_B[3]	P1079_B[3]	...							
2	UDP[0].Phase2	...			PKT[2]	...	PKT[2]		...			P0_B[2]	P1_B[2]	...	P1078_B[2]	P1079_B[2]	...							
1	UDP[0].Phase3	...			PKT[1]	...	PKT[1]		...			P0_B[1]	P1_B[1]	...	P1078_B[1]	P1079_B[1]	...							
0	UDP[0].Phase4	...			PKT[0]	...	PKT[0]		...			P0_B[0]	P1_B[0]	...	P1078_B[0]	P1079_B[0]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
▨ Video
🟪 Packet Data Enable

UDP mode Single Pixel 4x LVDS pin mapping demo (YCbCr444 8-bit, 1080p@60Hz)

	video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
23	UDP[5].Phase1	...							...			P0_Cr[7]	P1_Cr[7]	...	P1078_Cr[7]	P1079_Cr[7]	...							
22	UDP[5].Phase2	...							...			P0_Cr[6]	P1_Cr[6]	...	P1078_Cr[6]	P1079_Cr[6]	...							
21	UDP[5].Phase3	...							...			P0_Cr[5]	P1_Cr[5]	...	P1078_Cr[5]	P1079_Cr[5]	...							
20	UDP[5].Phase4	...							...			P0_Cr[4]	P1_Cr[4]	...	P1078_Cr[4]	P1079_Cr[4]	...							
19	UDP[4].Phase1	...							...			P0_Cr[3]	P1_Cr[3]	...	P1078_Cr[3]	P1079_Cr[3]	...							
18	UDP[4].Phase2	...							...			P0_Cr[2]	P1_Cr[2]	...	P1078_Cr[2]	P1079_Cr[2]	...							
17	UDP[4].Phase3	...							...			P0_Cr[1]	P1_Cr[1]	...	P1078_Cr[1]	P1079_Cr[1]	...							
16	UDP[4].Phase4	...							...			P0_Cr[0]	P1_Cr[0]	...	P1078_Cr[0]	P1079_Cr[0]	...							
15	UDP[3].Phase1	...	0	0				0	...	1	1	P0_Y[7]	P1_Y[7]	...	P1078_Y[7]	P1079_Y[7]	...							
14	UDP[3].Phase2	...	1	0				1	...	0	0	P0_Y[6]	P1_Y[6]	...	P1078_Y[6]	P1079_Y[6]	...							
13	UDP[3].Phase3	...	1	1				0	...	1	0	P0_Y[5]	P1_Y[5]	...	P1078_Y[5]	P1079_Y[5]	...							
12	UDP[3].Phase4	...							...			P0_Y[4]	P1_Y[4]	...	P1078_Y[4]	P1079_Y[4]	...							
11	UDP[2].Phase1	...							...			P0_Y[3]	P1_Y[3]	...	P1078_Y[3]	P1079_Y[3]	...	1	...	1				
10	UDP[2].Phase2	...							...			P0_Y[2]	P1_Y[2]	...	P1078_Y[2]	P1079_Y[2]	...					1	...	1
9	UDP[2].Phase3	...			1	...	1		...			P0_Y[1]	P1_Y[1]	...	P1078_Y[1]	P1079_Y[1]	...							
8	UDP[2].Phase4	...			PKT[8]	...	PKT[8]		...			P0_Y[0]	P1_Y[0]	...	P1078_Y[0]	P1079_Y[0]	...							
7	UDP[1].Phase1	...			PKT[7]	...	PKT[7]		...			P0_Cb[7]	P1_Cb[7]	...	P1078_Cb[7]	P1079_Cb[7]	...							
6	UDP[1].Phase2	...			PKT[6]	...	PKT[6]		...			P0_Cb[6]	P1_Cb[6]	...	P1078_Cb[6]	P1079_Cb[6]	...							
5	UDP[1].Phase3	...			PKT[5]	...	PKT[5]		...			P0_Cb[5]	P1_Cb[5]	...	P1078_Cb[5]	P1079_Cb[5]	...							
4	UDP[1].Phase4	...			PKT[4]	...	PKT[4]		...			P0_Cb[4]	P1_Cb[4]	...	P1078_Cb[4]	P1079_Cb[4]	...							
3	UDP[0].Phase1	...			PKT[3]	...	PKT[3]		...			P0_Cb[3]	P1_Cb[3]	...	P1078_Cb[3]	P1079_Cb[3]	...							
2	UDP[0].Phase2	...			PKT[2]	...	PKT[2]		...			P0_Cb[2]	P1_Cb[2]	...	P1078_Cb[2]	P1079_Cb[2]	...							
1	UDP[0].Phase3	...			PKT[1]	...	PKT[1]		...			P0_Cb[1]	P1_Cb[1]	...	P1078_Cb[1]	P1079_Cb[1]	...							
0	UDP[0].Phase4	...			PKT[0]	...	PKT[0]		...			P0_Cb[0]	P1_Cb[0]	...	P1078_Cb[0]	P1079_Cb[0]	...							

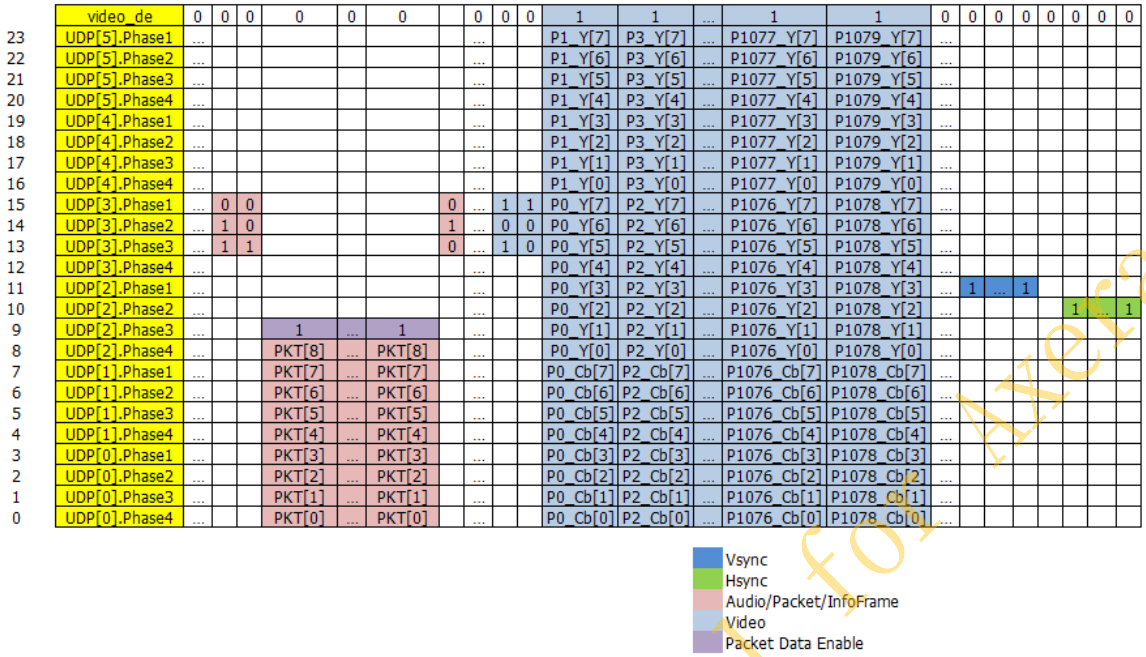
🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
🟦 Video
🟪 Packet Data Enable

UDP mode Single Pixel 4x LVDS pin mapping demo (YCbCr422 8-bit, 1080p@60Hz)

	video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
23	UDP[5].Phase1	...							...			P0_Cb[11]	P0_Cr[11]	...	P1078_Cb[11]	P1078_Cr[11]	...							
22	UDP[5].Phase2	...							...			P0_Cb[10]	P0_Cr[10]	...	P1078_Cb[10]	P1078_Cr[10]	...							
21	UDP[5].Phase3	...							...			P0_Cb[9]	P0_Cr[9]	...	P1078_Cb[9]	P1078_Cr[9]	...							
20	UDP[5].Phase4	...							...			P0_Cb[8]	P0_Cr[8]	...	P1078_Cb[8]	P1078_Cr[8]	...							
19	UDP[4].Phase1	...							...			P0_Cb[7]	P0_Cr[7]	...	P1078_Cb[7]	P1078_Cr[7]	...							
18	UDP[4].Phase2	...							...			P0_Cb[6]	P0_Cr[6]	...	P1078_Cb[6]	P1078_Cr[6]	...							
17	UDP[4].Phase3	...							...			P0_Cb[5]	P0_Cr[5]	...	P1078_Cb[5]	P1078_Cr[5]	...							
16	UDP[4].Phase4	...							...			P0_Cb[4]	P0_Cr[4]	...	P1078_Cb[4]	P1078_Cr[4]	...							
15	UDP[3].Phase1	...	0	0				0	...	1	1	P0_Y[11]	P1_Y[11]	...	P1078_Y[11]	P1079_Y[11]	...							
14	UDP[3].Phase2	...	1	0				1	...	0	0	P0_Y[10]	P1_Y[10]	...	P1078_Y[10]	P1079_Y[10]	...							
13	UDP[3].Phase3	...	1	1				0	...	1	0	P0_Y[9]	P1_Y[9]	...	P1078_Y[9]	P1079_Y[9]	...							
12	UDP[3].Phase4	...							...			P0_Y[8]	P1_Y[8]	...	P1078_Y[8]	P1079_Y[8]	...							
11	UDP[2].Phase1	...							...			P0_Y[7]	P1_Y[7]	...	P1078_Y[7]	P1079_Y[7]	...	1	...	1				
10	UDP[2].Phase2	...							...			P0_Y[6]	P1_Y[6]	...	P1078_Y[6]	P1079_Y[6]	...					1	...	1
9	UDP[2].Phase3	...			1	...	1		...			P0_Y[5]	P1_Y[5]	...	P1078_Y[5]	P1079_Y[5]	...							
8	UDP[2].Phase4	...			PKT[8]	...	PKT[8]		...			P0_Y[4]	P1_Y[4]	...	P1078_Y[4]	P1079_Y[4]	...							
7	UDP[1].Phase1	...			PKT[7]	...	PKT[7]		...			P0_Cb[3]	P0_Cr[3]	...	P1078_Cb[3]	P1078_Cr[3]	...							
6	UDP[1].Phase2	...			PKT[6]	...	PKT[6]		...			P0_Cb[2]	P0_Cr[2]	...	P1078_Cb[2]	P1078_Cr[2]	...							
5	UDP[1].Phase3	...			PKT[5]	...	PKT[5]		...			P0_Cb[1]	P0_Cr[1]	...	P1078_Cb[1]	P1078_Cr[1]	...							
4	UDP[1].Phase4	...			PKT[4]	...	PKT[4]		...			P0_Cb[0]	P0_Cr[0]	...	P1078_Cb[0]	P1078_Cr[0]	...							
3	UDP[0].Phase1	...			PKT[3]	...	PKT[3]		...			P0_Y[3]	P1_Y[3]	...	P1078_Y[3]	P1079_Y[3]	...							
2	UDP[0].Phase2	...			PKT[2]	...	PKT[2]		...			P0_Y[2]	P1_Y[2]	...	P1078_Y[2]	P1079_Y[2]	...							
1	UDP[0].Phase3	...			PKT[1]	...	PKT[1]		...			P0_Y[1]	P1_Y[1]	...	P1078_Y[1]	P1079_Y[1]	...							
0	UDP[0].Phase4	...			PKT[0]	...	PKT[0]		...			P0_Y[0]	P1_Y[0]	...	P1078_Y[0]	P1079_Y[0]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
🟦 Video
🟪 Packet Data Enable

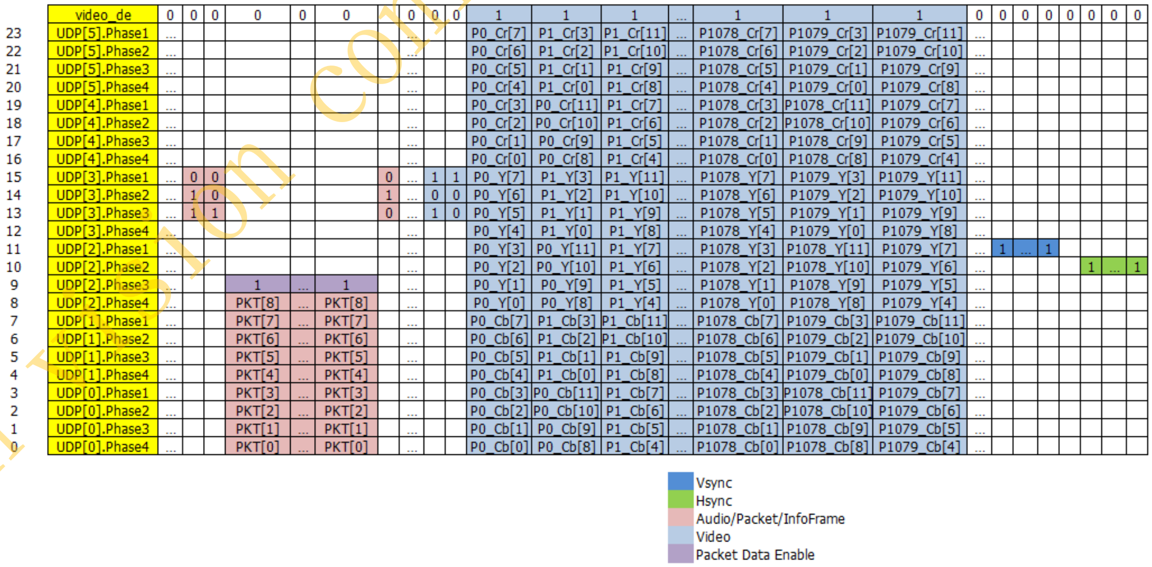
UDP mode Single Pixel 4x LVDS pin mapping demo (YCbCr420 8-bit, 1080p@60Hz)



	video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
23	UDP[5].Phase1	...							...			P1_Y[7]	P3_Y[7]	...	P1077_Y[7]	P1079_Y[7]	...							
22	UDP[5].Phase2	...							...			P1_Y[6]	P3_Y[6]	...	P1077_Y[6]	P1079_Y[6]	...							
21	UDP[5].Phase3	...							...			P1_Y[5]	P3_Y[5]	...	P1077_Y[5]	P1079_Y[5]	...							
20	UDP[5].Phase4	...							...			P1_Y[4]	P3_Y[4]	...	P1077_Y[4]	P1079_Y[4]	...							
19	UDP[4].Phase1	...							...			P1_Y[3]	P3_Y[3]	...	P1077_Y[3]	P1079_Y[3]	...							
18	UDP[4].Phase2	...							...			P1_Y[2]	P3_Y[2]	...	P1077_Y[2]	P1079_Y[2]	...							
17	UDP[4].Phase3	...							...			P1_Y[1]	P3_Y[1]	...	P1077_Y[1]	P1079_Y[1]	...							
16	UDP[4].Phase4	...							...			P1_Y[0]	P3_Y[0]	...	P1077_Y[0]	P1079_Y[0]	...							
15	UDP[3].Phase1	...	0	0				0	...	1	1	P0_Y[7]	P2_Y[7]	...	P1076_Y[7]	P1078_Y[7]	...							
14	UDP[3].Phase2	...	1	0				1	...	0	0	P0_Y[6]	P2_Y[6]	...	P1076_Y[6]	P1078_Y[6]	...							
13	UDP[3].Phase3	...	1	1				0	...	1	0	P0_Y[5]	P2_Y[5]	...	P1076_Y[5]	P1078_Y[5]	...							
12	UDP[3].Phase4	...							...			P0_Y[4]	P2_Y[4]	...	P1076_Y[4]	P1078_Y[4]	...							
11	UDP[2].Phase1	...							...			P0_Y[3]	P2_Y[3]	...	P1076_Y[3]	P1078_Y[3]	...	1	...	1				
10	UDP[2].Phase2	...							...			P0_Y[2]	P2_Y[2]	...	P1076_Y[2]	P1078_Y[2]	...					1	...	1
9	UDP[2].Phase3	...			1	...	1		...			P0_Y[1]	P2_Y[1]	...	P1076_Y[1]	P1078_Y[1]	...							
8	UDP[2].Phase4	...			PKT[8]	...	PKT[8]		...			P0_Y[0]	P2_Y[0]	...	P1076_Y[0]	P1078_Y[0]	...							
7	UDP[1].Phase1	...			PKT[7]	...	PKT[7]		...			P0_Cb[7]	P2_Cb[7]	...	P1076_Cb[7]	P1078_Cb[7]	...							
6	UDP[1].Phase2	...			PKT[6]	...	PKT[6]		...			P0_Cb[6]	P2_Cb[6]	...	P1076_Cb[6]	P1078_Cb[6]	...							
5	UDP[1].Phase3	...			PKT[5]	...	PKT[5]		...			P0_Cb[5]	P2_Cb[5]	...	P1076_Cb[5]	P1078_Cb[5]	...							
4	UDP[1].Phase4	...			PKT[4]	...	PKT[4]		...			P0_Cb[4]	P2_Cb[4]	...	P1076_Cb[4]	P1078_Cb[4]	...							
3	UDP[0].Phase1	...			PKT[3]	...	PKT[3]		...			P0_Cb[3]	P2_Cb[3]	...	P1076_Cb[3]	P1078_Cb[3]	...							
2	UDP[0].Phase2	...			PKT[2]	...	PKT[2]		...			P0_Cb[2]	P2_Cb[2]	...	P1076_Cb[2]	P1078_Cb[2]	...							
1	UDP[0].Phase3	...			PKT[1]	...	PKT[1]		...			P0_Cb[1]	P2_Cb[1]	...	P1076_Cb[1]	P1078_Cb[1]	...							
0	UDP[0].Phase4	...			PKT[0]	...	PKT[0]		...			P0_Cb[0]	P2_Cb[0]	...	P1076_Cb[0]	P1078_Cb[0]	...							

■ Vsync
■ Hsync
■ Audio/Packet/InfoFrame
■ Video
■ Packet Data Enable

UDP mode Single Pixel 4x LVDS pin mapping demo (YCbCr444 12-bit deep color, 1080p@60Hz)



	video_de	0	0	0	0	0	0		0	0	0	1	1	1	...	1	1	1	0	0	0	0	0	0	0	0
23	UDP[5].Phase1	...							...			P0_Cr[7]	P1_Cr[3]	P1_Cr[11]	...	P1078_Cr[7]	P1079_Cr[3]	P1079_Cr[11]	...							
22	UDP[5].Phase2	...							...			P0_Cr[6]	P1_Cr[2]	P1_Cr[10]	...	P1078_Cr[6]	P1079_Cr[2]	P1079_Cr[10]	...							
21	UDP[5].Phase3	...							...			P0_Cr[5]	P1_Cr[1]	P1_Cr[9]	...	P1078_Cr[5]	P1079_Cr[1]	P1079_Cr[9]	...							
20	UDP[5].Phase4	...							...			P0_Cr[4]	P1_Cr[0]	P1_Cr[8]	...	P1078_Cr[4]	P1079_Cr[0]	P1079_Cr[8]	...							
19	UDP[4].Phase1	...							...			P0_Cr[3]	P0_Cr[11]	P1_Cr[7]	...	P1078_Cr[3]	P1078_Cr[11]	P1079_Cr[7]	...							
18	UDP[4].Phase2	...							...			P0_Cr[2]	P0_Cr[10]	P1_Cr[6]	...	P1078_Cr[2]	P1078_Cr[10]	P1079_Cr[6]	...							
17	UDP[4].Phase3	...							...			P0_Cr[1]	P0_Cr[9]	P1_Cr[5]	...	P1078_Cr[1]	P1078_Cr[9]	P1079_Cr[5]	...							
16	UDP[4].Phase4	...							...			P0_Cr[0]	P0_Cr[8]	P1_Cr[4]	...	P1078_Cr[0]	P1078_Cr[8]	P1079_Cr[4]	...							
15	UDP[3].Phase1	...	0	0				0	...	1	1	P0_Y[7]	P1_Y[3]	P1_Y[11]	...	P1078_Y[7]	P1079_Y[3]	P1079_Y[11]	...							
14	UDP[3].Phase2	...	1	0				1	...	0	0	P0_Y[6]	P1_Y[2]	P1_Y[10]	...	P1078_Y[6]	P1079_Y[2]	P1079_Y[10]	...							
13	UDP[3].Phase3	...	1	1				0	...	1	0	P0_Y[5]	P1_Y[1]	P1_Y[9]	...	P1078_Y[5]	P1079_Y[1]	P1079_Y[9]	...							
12	UDP[3].Phase4	...							...			P0_Y[4]	P1_Y[0]	P1_Y[8]	...	P1078_Y[4]	P1079_Y[0]	P1079_Y[8]	...							
11	UDP[2].Phase1	...							...			P0_Y[3]	P0_Y[11]	P1_Y[7]	...	P1078_Y[3]	P1078_Y[11]	P1079_Y[7]	...	1	...	1				
10	UDP[2].Phase2	...							...			P0_Y[2]	P0_Y[10]	P1_Y[6]	...	P1078_Y[2]	P1078_Y[10]	P1079_Y[6]	...					1	...	1
9	UDP[2].Phase3	...			1	...	1		...			P0_Y[1]	P0_Y[9]	P1_Y[5]	...	P1078_Y[1]	P1078_Y[9]	P1079_Y[5]	...							
8	UDP[2].Phase4	...			PKT[8]	...	PKT[8]		...			P0_Y[0]	P0_Y[8]	P1_Y[4]	...	P1078_Y[0]	P1078_Y[8]	P1079_Y[4]	...							
7	UDP[1].Phase1	...			PKT[7]	...	PKT[7]		...			P0_Cb[7]	P1_Cb[3]	P1_Cb[11]	...	P1078_Cb[7]	P1079_Cb[3]	P1079_Cb[11]	...							
6	UDP[1].Phase2	...			PKT[6]	...	PKT[6]		...			P0_Cb[6]	P1_Cb[2]	P1_Cb[10]	...	P1078_Cb[6]	P1079_Cb[2]	P1079_Cb[10]	...							
5	UDP[1].Phase3	...			PKT[5]	...	PKT[5]		...			P0_Cb[5]	P1_Cb[1]	P1_Cb[9]	...	P1078_Cb[5]	P1079_Cb[1]	P1079_Cb[9]	...							
4	UDP[1].Phase4	...			PKT[4]	...	PKT[4]		...			P0_Cb[4]	P1_Cb[0]	P1_Cb[8]	...	P1078_Cb[4]	P1079_Cb[0]	P1079_Cb[8]	...							
3	UDP[0].Phase1	...			PKT[3]	...	PKT[3]		...			P0_Cb[3]	P0_Cb[11]	P1_Cb[7]	...	P1078_Cb[3]	P1078_Cb[11]	P1079_Cb[7]	...							
2	UDP[0].Phase2	...			PKT[2]	...	PKT[2]		...			P0_Cb[2]	P0_Cb[10]	P1_Cb[6]	...	P1078_Cb[2]	P1078_Cb[10]	P1079_Cb[6]	...							
1	UDP[0].Phase3	...			PKT[1]	...	PKT[1]		...			P0_Cb[1]	P0_Cb[9]	P1_Cb[5]	...	P1078_Cb[1]	P1078_Cb[9]	P1079_Cb[5]	...							
0	UDP[0].Phase4	...			PKT[0]	...	PKT[0]		...			P0_Cb[0]	P0_Cb[8]	P1_Cb[4]	...	P1078_Cb[0]	P1078_Cb[8]	P1079_Cb[4]	...							

■ Vsync
■ Hsync
■ Audio/Packet/InfoFrame
■ Video
■ Packet Data Enable

UDP mode Dual Pixel 4x LVDS pin mapping demo
(RGB 8-bit deep color, 1080p@60Hz)

	video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
47	UDP[11].Phase1	...							...			P1_R[7]	P3_R[7]	...	P1077_R[7]	P1079_R[7]	...							
46	UDP[11].Phase2	...							...			P1_R[6]	P3_R[6]	...	P1077_R[6]	P1079_R[6]	...							
45	UDP[11].Phase3	...							...			P1_R[5]	P3_R[5]	...	P1077_R[5]	P1079_R[5]	...							
44	UDP[11].Phase4	...							...			P1_R[4]	P3_R[4]	...	P1077_R[4]	P1079_R[4]	...							
43	UDP[10].Phase1	...							...			P1_R[3]	P3_R[3]	...	P1077_R[3]	P1079_R[3]	...							
42	UDP[10].Phase2	...							...			P1_R[2]	P3_R[2]	...	P1077_R[2]	P1079_R[2]	...							
41	UDP[10].Phase3	...							...			P1_R[1]	P3_R[1]	...	P1077_R[1]	P1079_R[1]	...							
40	UDP[10].Phase4	...							...			P1_R[0]	P3_R[0]	...	P1077_R[0]	P1079_R[0]	...							
39	UDP[9].Phase1	...	0	0				0	...	1	1	P1_G[7]	P3_G[7]	...	P1077_G[7]	P1079_G[7]	...							
38	UDP[9].Phase2	...	1	0				1	...	0	0	P1_G[6]	P3_G[6]	...	P1077_G[6]	P1079_G[6]	...							
37	UDP[9].Phase3	...	1	1				0	...	1	0	P1_G[5]	P3_G[5]	...	P1077_G[5]	P1079_G[5]	...							
36	UDP[9].Phase4	...							...			P1_G[4]	P3_G[4]	...	P1077_G[4]	P1079_G[4]	...							
35	UDP[8].Phase1	...							...			P1_G[3]	P3_G[3]	...	P1077_G[3]	P1079_G[3]	...	1	...	1				
34	UDP[8].Phase2	...							...			P1_G[2]	P3_G[2]	...	P1077_G[2]	P1079_G[2]	...					1	...	1
33	UDP[8].Phase3	...			1	...	1		...			P1_G[1]	P3_G[1]	...	P1077_G[1]	P1079_G[1]	...							
32	UDP[8].Phase4	...			PKT[8]	...	PKT[8]		...			P1_G[0]	P3_G[0]	...	P1077_G[0]	P1079_G[0]	...							
31	UDP[7].Phase1	...			PKT[7]	...	PKT[7]		...			P1_B[7]	P3_B[7]	...	P1077_B[7]	P1079_B[7]	...							
30	UDP[7].Phase2	...			PKT[6]	...	PKT[6]		...			P1_B[6]	P3_B[6]	...	P1077_B[6]	P1079_B[6]	...							
29	UDP[7].Phase3	...			PKT[5]	...	PKT[5]		...			P1_B[5]	P3_B[5]	...	P1077_B[5]	P1079_B[5]	...							
28	UDP[7].Phase4	...			PKT[4]	...	PKT[4]		...			P1_B[4]	P3_B[4]	...	P1077_B[4]	P1079_B[4]	...							
27	UDP[6].Phase1	...			PKT[3]	...	PKT[3]		...			P1_B[3]	P3_B[3]	...	P1077_B[3]	P1079_B[3]	...							
26	UDP[6].Phase2	...			PKT[2]	...	PKT[2]		...			P1_B[2]	P3_B[2]	...	P1077_B[2]	P1079_B[2]	...							
25	UDP[6].Phase3	...			PKT[1]	...	PKT[1]		...			P1_B[1]	P3_B[1]	...	P1077_B[1]	P1079_B[1]	...							
24	UDP[6].Phase4	...			PKT[0]	...	PKT[0]		...			P1_B[0]	P3_B[0]	...	P1077_B[0]	P1079_B[0]	...							
23	UDP[5].Phase1	...							...			P0_R[7]	P2_R[7]	...	P1076_R[7]	P1078_R[7]	...							
22	UDP[5].Phase2	...							...			P0_R[6]	P2_R[6]	...	P1076_R[6]	P1078_R[6]	...							
21	UDP[5].Phase3	...							...			P0_R[5]	P2_R[5]	...	P1076_R[5]	P1078_R[5]	...							
20	UDP[5].Phase4	...							...			P0_R[4]	P2_R[4]	...	P1076_R[4]	P1078_R[4]	...							
19	UDP[4].Phase1	...							...			P0_R[3]	P2_R[3]	...	P1076_R[3]	P1078_R[3]	...							
18	UDP[4].Phase2	...							...			P0_R[2]	P2_R[2]	...	P1076_R[2]	P1078_R[2]	...							
17	UDP[4].Phase3	...							...			P0_R[1]	P2_R[1]	...	P1076_R[1]	P1078_R[1]	...							
16	UDP[4].Phase4	...							...			P0_R[0]	P2_R[0]	...	P1076_R[0]	P1078_R[0]	...							
15	UDP[3].Phase1	...	0	0				0	...	1	1	P0_G[7]	P2_G[7]	...	P1076_G[7]	P1078_G[7]	...							
14	UDP[3].Phase2	...	1	0				1	...	0	0	P0_G[6]	P2_G[6]	...	P1076_G[6]	P1078_G[6]	...							
13	UDP[3].Phase3	...	1	1				0	...	1	0	P0_G[5]	P2_G[5]	...	P1076_G[5]	P1078_G[5]	...							
12	UDP[3].Phase4	...							...			P0_G[4]	P2_G[4]	...	P1076_G[4]	P1078_G[4]	...							
11	UDP[2].Phase1	...							...			P0_G[3]	P2_G[3]	...	P1076_G[3]	P1078_G[3]	...	1	...	1				
10	UDP[2].Phase2	...							...			P0_G[2]	P2_G[2]	...	P1076_G[2]	P1078_G[2]	...					1	...	1
9	UDP[2].Phase3	...			1	...	1		...			P0_G[1]	P2_G[1]	...	P1076_G[1]	P1078_G[1]	...							
8	UDP[2].Phase4	...			PKT[8]	...	PKT[8]		...			P0_G[0]	P2_G[0]	...	P1076_G[0]	P1078_G[0]	...							
7	UDP[1].Phase1	...			PKT[7]	...	PKT[7]		...			P0_B[7]	P2_B[7]	...	P1076_B[7]	P1078_B[7]	...							
6	UDP[1].Phase2	...			PKT[6]	...	PKT[6]		...			P0_B[6]	P2_B[6]	...	P1076_B[6]	P1078_B[6]	...							
5	UDP[1].Phase3	...			PKT[5]	...	PKT[5]		...			P0_B[5]	P2_B[5]	...	P1076_B[5]	P1078_B[5]	...							
4	UDP[1].Phase4	...			PKT[4]	...	PKT[4]		...			P0_B[4]	P2_B[4]	...	P1076_B[4]	P1078_B[4]	...							
3	UDP[0].Phase1	...			PKT[3]	...	PKT[3]		...			P0_B[3]	P2_B[3]	...	P1076_B[3]	P1078_B[3]	...							
2	UDP[0].Phase2	...			PKT[2]	...	PKT[2]		...			P0_B[2]	P2_B[2]	...	P1076_B[2]	P1078_B[2]	...							
1	UDP[0].Phase3	...			PKT[1]	...	PKT[1]		...			P0_B[1]	P2_B[1]	...	P1076_B[1]	P1078_B[1]	...							
0	UDP[0].Phase4	...			PKT[0]	...	PKT[0]		...			P0_B[0]	P2_B[0]	...	P1076_B[0]	P1078_B[0]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
🟦 Video
🟪 Packet Data Enable

1.5 HDMI to TTL in UDP mode

In this HDMI to TTL UDP application mode, HDMI input's raw Video/Audio/Control Packets are time division multiplexed to a group of 36 TTL pins in single pixel mode or 48 TTL pins in dual pixel mode. An extra TTL pin is dedicated to transmit DE signal. Single pixel TTL mode is used for <=300MHz pixel clock timing (4K30Hz 12-bit in maximum bandwidth), refer to Table 14 Single Pixel Mode YCbCr/RGB 4:4:4 Pin Mapping in <GSV2011 datasheet> for the 36 TTL pins. The maximum TTL data lane frequency is 300MHz in this mode.

Dual pixel TTL mode should be used for >300MHz pixel clock timing (4K60Hz 8-bit in maximum bandwidth), please refer to Table 17 Dual Pixel Mode YCbCr/RGB 4:4:4 Pin Mapping in <GSV2011 datasheet> for the 48 TTL pins. The maximum TTL data lane frequency is 300MHz in this mode.

By default, the external receiver should receive these pins of TTL data and a pin of input TTL clock. The external receiver should use the given TTL clock to generate its own sampling clock, then utilizes the pin of DE to decode the rest pins of data into Video/Audio/Control Packets.

The decoding of DE is critical for separating active video pixel data and other valid packets. When DE = 1, video data is present on the pins. When DE = 0, audio/control packets are present. The DE pin behavior follows the active pixel data enable definition of video timings defined by CEA/VESA.

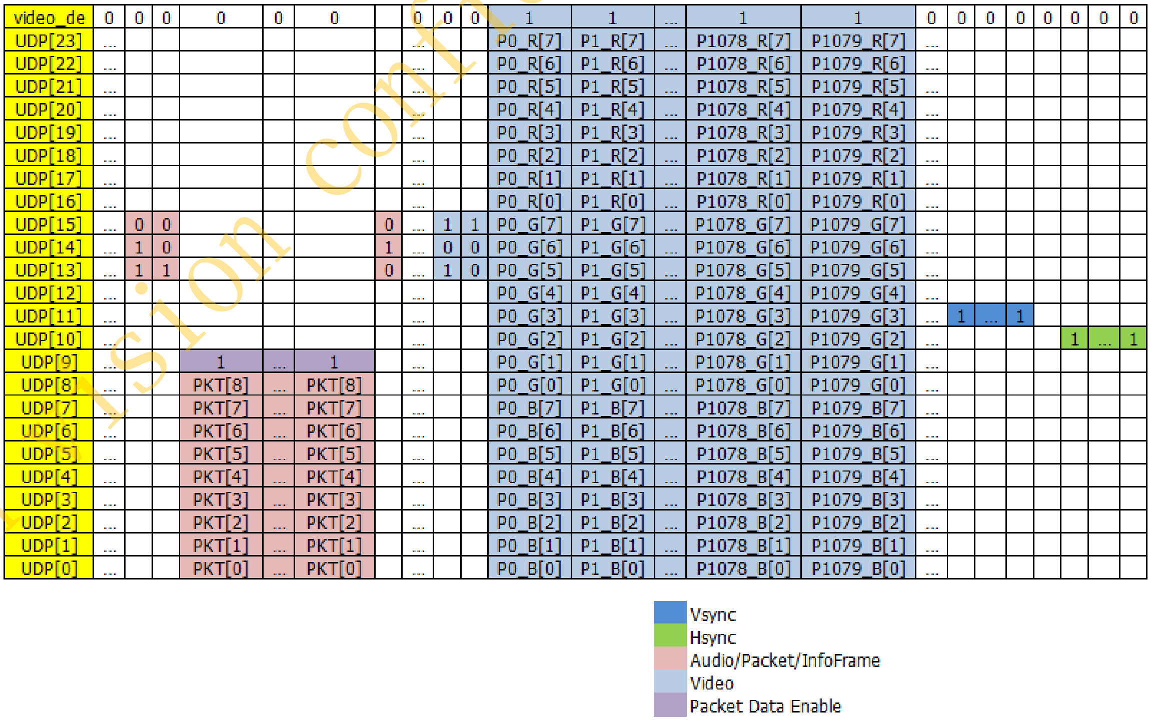
It should be noted that Leading Guard Band and Trailing Guard Band both can be multiple cycls.

In the 3 bytes configuration (refer to Section4.1), the 24-bit UDP TTL pins can be modified with Mode CFG2(Byte 1)[5:4] = i2c_tx_par_sub_mode.

When i2c_tx_par_sub_mode = 01, the lower 24-bits TTL pins will be used. All following pin mapping diagram uses this mode. When i2c_tx_par_sub_mode = 00, TTL[35:28], TTL[23:16], TTL[11:4] will be used for 24-bit UDP TTL pins.

Typical TTL UDP mode pin mapping diagrams are shown below.

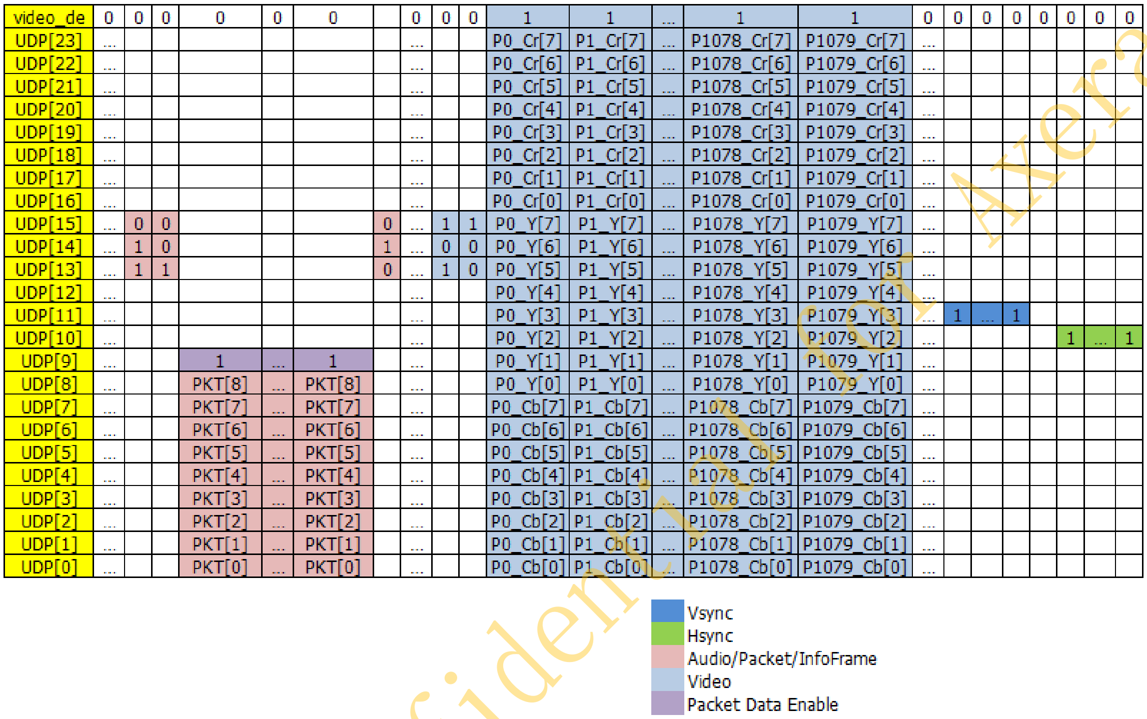
UDP mode Single Pixel TTL pin mapping demo (RGB 8-bit, 1080p@60Hz)



video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
UDP[23]	....							....			P0_R[7]	P1_R[7]	...	P1078_R[7]	P1079_R[7]	...							
UDP[22]	....							....			P0_R[6]	P1_R[6]	...	P1078_R[6]	P1079_R[6]	...							
UDP[21]	....							....			P0_R[5]	P1_R[5]	...	P1078_R[5]	P1079_R[5]	...							
UDP[20]	....							....			P0_R[4]	P1_R[4]	...	P1078_R[4]	P1079_R[4]	...							
UDP[19]	....							....			P0_R[3]	P1_R[3]	...	P1078_R[3]	P1079_R[3]	...							
UDP[18]	....							....			P0_R[2]	P1_R[2]	...	P1078_R[2]	P1079_R[2]	...							
UDP[17]	....							....			P0_R[1]	P1_R[1]	...	P1078_R[1]	P1079_R[1]	...							
UDP[16]	....							....			P0_R[0]	P1_R[0]	...	P1078_R[0]	P1079_R[0]	...							
UDP[15]	....	0	0				0	....	1	1	P0_G[7]	P1_G[7]	...	P1078_G[7]	P1079_G[7]	...							
UDP[14]	....	1	0				1	....	0	0	P0_G[6]	P1_G[6]	...	P1078_G[6]	P1079_G[6]	...							
UDP[13]	....	1	1				0	....	1	0	P0_G[5]	P1_G[5]	...	P1078_G[5]	P1079_G[5]	...							
UDP[12]	....							....			P0_G[4]	P1_G[4]	...	P1078_G[4]	P1079_G[4]	...							
UDP[11]	....							....			P0_G[3]	P1_G[3]	...	P1078_G[3]	P1079_G[3]	...	1	...	1				
UDP[10]	....							....			P0_G[2]	P1_G[2]	...	P1078_G[2]	P1079_G[2]	...					1	...	1
UDP[9]	....			1	...	1		....			P0_G[1]	P1_G[1]	...	P1078_G[1]	P1079_G[1]	...							
UDP[8]	....			PKT[8]	...	PKT[8]		....			P0_G[0]	P1_G[0]	...	P1078_G[0]	P1079_G[0]	...							
UDP[7]	....			PKT[7]	...	PKT[7]		....			P0_B[7]	P1_B[7]	...	P1078_B[7]	P1079_B[7]	...							
UDP[6]	....			PKT[6]	...	PKT[6]		....			P0_B[6]	P1_B[6]	...	P1078_B[6]	P1079_B[6]	...							
UDP[5]	....			PKT[5]	...	PKT[5]		....			P0_B[5]	P1_B[5]	...	P1078_B[5]	P1079_B[5]	...							
UDP[4]	....			PKT[4]	...	PKT[4]		....			P0_B[4]	P1_B[4]	...	P1078_B[4]	P1079_B[4]	...							
UDP[3]	....			PKT[3]	...	PKT[3]		....			P0_B[3]	P1_B[3]	...	P1078_B[3]	P1079_B[3]	...							
UDP[2]	....			PKT[2]	...	PKT[2]		....			P0_B[2]	P1_B[2]	...	P1078_B[2]	P1079_B[2]	...							
UDP[1]	....			PKT[1]	...	PKT[1]		....			P0_B[1]	P1_B[1]	...	P1078_B[1]	P1079_B[1]	...							
UDP[0]	....			PKT[0]	...	PKT[0]		....			P0_B[0]	P1_B[0]	...	P1078_B[0]	P1079_B[0]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
🟦 Video
🟪 Packet Data Enable

UDP mode Single Pixel TTL pin mapping demo (YCbCr444 8-bit, 1080p@60Hz)



video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
UDP[23]	...							...			P0_Cr[7]	P1_Cr[7]	...	P1078_Cr[7]	P1079_Cr[7]	...							
UDP[22]	...							...			P0_Cr[6]	P1_Cr[6]	...	P1078_Cr[6]	P1079_Cr[6]	...							
UDP[21]	...							...			P0_Cr[5]	P1_Cr[5]	...	P1078_Cr[5]	P1079_Cr[5]	...							
UDP[20]	...							...			P0_Cr[4]	P1_Cr[4]	...	P1078_Cr[4]	P1079_Cr[4]	...							
UDP[19]	...							...			P0_Cr[3]	P1_Cr[3]	...	P1078_Cr[3]	P1079_Cr[3]	...							
UDP[18]	...							...			P0_Cr[2]	P1_Cr[2]	...	P1078_Cr[2]	P1079_Cr[2]	...							
UDP[17]	...							...			P0_Cr[1]	P1_Cr[1]	...	P1078_Cr[1]	P1079_Cr[1]	...							
UDP[16]	...							...			P0_Cr[0]	P1_Cr[0]	...	P1078_Cr[0]	P1079_Cr[0]	...							
UDP[15]	...	0	0				0	...	1	1	P0_Y[7]	P1_Y[7]	...	P1078_Y[7]	P1079_Y[7]	...							
UDP[14]	...	1	0				1	...	0	0	P0_Y[6]	P1_Y[6]	...	P1078_Y[6]	P1079_Y[6]	...							
UDP[13]	...	1	1				0	...	1	0	P0_Y[5]	P1_Y[5]	...	P1078_Y[5]	P1079_Y[5]	...							
UDP[12]	...							...			P0_Y[4]	P1_Y[4]	...	P1078_Y[4]	P1079_Y[4]	...							
UDP[11]	...							...			P0_Y[3]	P1_Y[3]	...	P1078_Y[3]	P1079_Y[3]	...	1	...	1				
UDP[10]	...							...			P0_Y[2]	P1_Y[2]	...	P1078_Y[2]	P1079_Y[2]	...					1	...	1
UDP[9]	...			1	...	1		...			P0_Y[1]	P1_Y[1]	...	P1078_Y[1]	P1079_Y[1]	...							
UDP[8]	...			PKT[8]	...	PKT[8]		...			P0_Y[0]	P1_Y[0]	...	P1078_Y[0]	P1079_Y[0]	...							
UDP[7]	...			PKT[7]	...	PKT[7]		...			P0_Cb[7]	P1_Cb[7]	...	P1078_Cb[7]	P1079_Cb[7]	...							
UDP[6]	...			PKT[6]	...	PKT[6]		...			P0_Cb[6]	P1_Cb[6]	...	P1078_Cb[6]	P1079_Cb[6]	...							
UDP[5]	...			PKT[5]	...	PKT[5]		...			P0_Cb[5]	P1_Cb[5]	...	P1078_Cb[5]	P1079_Cb[5]	...							
UDP[4]	...			PKT[4]	...	PKT[4]		...			P0_Cb[4]	P1_Cb[4]	...	P1078_Cb[4]	P1079_Cb[4]	...							
UDP[3]	...			PKT[3]	...	PKT[3]		...			P0_Cb[3]	P1_Cb[3]	...	P1078_Cb[3]	P1079_Cb[3]	...							
UDP[2]	...			PKT[2]	...	PKT[2]		...			P0_Cb[2]	P1_Cb[2]	...	P1078_Cb[2]	P1079_Cb[2]	...							
UDP[1]	...			PKT[1]	...	PKT[1]		...			P0_Cb[1]	P1_Cb[1]	...	P1078_Cb[1]	P1079_Cb[1]	...							
UDP[0]	...			PKT[0]	...	PKT[0]		...			P0_Cb[0]	P1_Cb[0]	...	P1078_Cb[0]	P1079_Cb[0]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
🟦 Video
🟪 Packet Data Enable

UDP mode Single Pixel TTL pin mapping demo (YCbCr422 8-bit, 1080p@60Hz)

video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
UDP[23]	...							...			P0_Cb[11]	P0_Cr[11]	...	P1078_Cb[11]	P1078_Cr[11]	...							
UDP[22]	...							...			P0_Cb[10]	P0_Cr[10]	...	P1078_Cb[10]	P1078_Cr[10]	...							
UDP[21]	...							...			P0_Cb[9]	P0_Cr[9]	...	P1078_Cb[9]	P1078_Cr[9]	...							
UDP[20]	...							...			P0_Cb[8]	P0_Cr[8]	...	P1078_Cb[8]	P1078_Cr[8]	...							
UDP[19]	...							...			P0_Cb[7]	P0_Cr[7]	...	P1078_Cb[7]	P1078_Cr[7]	...							
UDP[18]	...							...			P0_Cb[6]	P0_Cr[6]	...	P1078_Cb[6]	P1078_Cr[6]	...							
UDP[17]	...							...			P0_Cb[5]	P0_Cr[5]	...	P1078_Cb[5]	P1078_Cr[5]	...							
UDP[16]	...							...			P0_Cb[4]	P0_Cr[4]	...	P1078_Cb[4]	P1078_Cr[4]	...							
UDP[15]	...	0	0				0	...	1	1	P0_Y[11]	P1_Y[11]	...	P1078_Y[11]	P1079_Y[11]	...							
UDP[14]	...	1	0				1	...	0	0	P0_Y[10]	P1_Y[10]	...	P1078_Y[10]	P1079_Y[10]	...							
UDP[13]	...	1	1				0	...	1	0	P0_Y[9]	P1_Y[9]	...	P1078_Y[9]	P1079_Y[9]	...							
UDP[12]	...							...			P0_Y[8]	P1_Y[8]	...	P1078_Y[8]	P1079_Y[8]	...							
UDP[11]	...							...			P0_Y[7]	P1_Y[7]	...	P1078_Y[7]	P1079_Y[7]	...	1	...	1				
UDP[10]	...							...			P0_Y[6]	P1_Y[6]	...	P1078_Y[6]	P1079_Y[6]	...					1	...	1
UDP[9]	...			1	...	1		...			P0_Y[5]	P1_Y[5]	...	P1078_Y[5]	P1079_Y[5]	...							
UDP[8]	...			PKT[8]	...	PKT[8]		...			P0_Y[4]	P1_Y[4]	...	P1078_Y[4]	P1079_Y[4]	...							
UDP[7]	...			PKT[7]	...	PKT[7]		...			P0_Cb[3]	P0_Cr[3]	...	P1078_Cb[3]	P1078_Cr[3]	...							
UDP[6]	...			PKT[6]	...	PKT[6]		...			P0_Cb[2]	P0_Cr[2]	...	P1078_Cb[2]	P1078_Cr[2]	...							
UDP[5]	...			PKT[5]	...	PKT[5]		...			P0_Cb[1]	P0_Cr[1]	...	P1078_Cb[1]	P1078_Cr[1]	...							
UDP[4]	...			PKT[4]	...	PKT[4]		...			P0_Cb[0]	P0_Cr[0]	...	P1078_Cb[0]	P1078_Cr[0]	...							
UDP[3]	...			PKT[3]	...	PKT[3]		...			P0_Y[3]	P1_Y[3]	...	P1078_Y[3]	P1079_Y[3]	...							
UDP[2]	...			PKT[2]	...	PKT[2]		...			P0_Y[2]	P1_Y[2]	...	P1078_Y[2]	P1079_Y[2]	...							
UDP[1]	...			PKT[1]	...	PKT[1]		...			P0_Y[1]	P1_Y[1]	...	P1078_Y[1]	P1079_Y[1]	...							
UDP[0]	...			PKT[0]	...	PKT[0]		...			P0_Y[0]	P1_Y[0]	...	P1078_Y[0]	P1079_Y[0]	...							

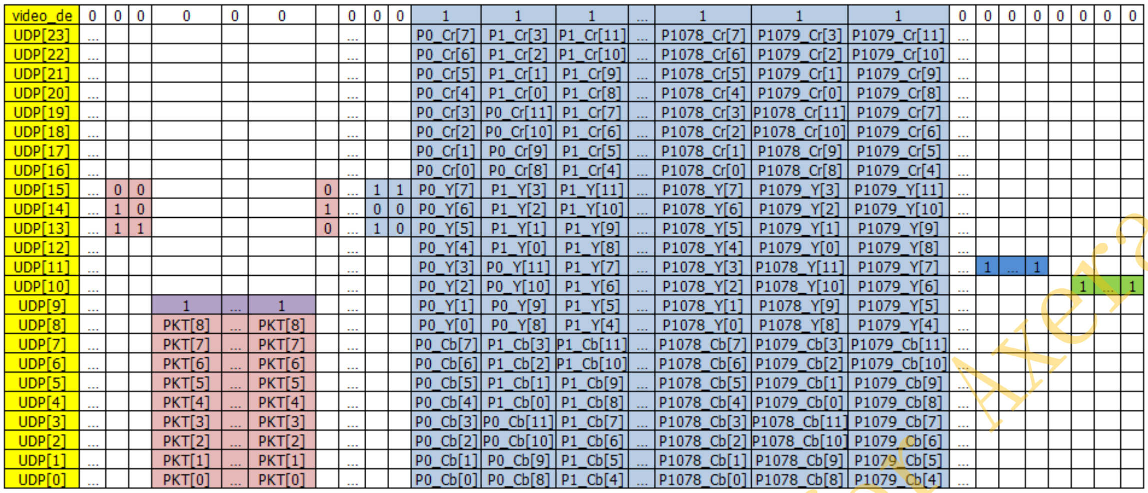
🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
▨ Video
🟪 Packet Data Enable

UDP mode Single Pixel TTL pin mapping demo (YCbCr420 8-bit, 1080p@60Hz)

video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
UDP[23]	...							...			P1_Y[7]	P3_Y[7]	...	P1077_Y[7]	P1079_Y[7]	...							
UDP[22]	...							...			P1_Y[6]	P3_Y[6]	...	P1077_Y[6]	P1079_Y[6]	...							
UDP[21]	...							...			P1_Y[5]	P3_Y[5]	...	P1077_Y[5]	P1079_Y[5]	...							
UDP[20]	...							...			P1_Y[4]	P3_Y[4]	...	P1077_Y[4]	P1079_Y[4]	...							
UDP[19]	...							...			P1_Y[3]	P3_Y[3]	...	P1077_Y[3]	P1079_Y[3]	...							
UDP[18]	...							...			P1_Y[2]	P3_Y[2]	...	P1077_Y[2]	P1079_Y[2]	...							
UDP[17]	...							...			P1_Y[1]	P3_Y[1]	...	P1077_Y[1]	P1079_Y[1]	...							
UDP[16]	...							...			P1_Y[0]	P3_Y[0]	...	P1077_Y[0]	P1079_Y[0]	...							
UDP[15]	...	0	0				0	...	1	1	P0_Y[7]	P2_Y[7]	...	P1076_Y[7]	P1078_Y[7]	...							
UDP[14]	...	1	0				1	...	0	0	P0_Y[6]	P2_Y[6]	...	P1076_Y[6]	P1078_Y[6]	...							
UDP[13]	...	1	1				0	...	1	0	P0_Y[5]	P2_Y[5]	...	P1076_Y[5]	P1078_Y[5]	...							
UDP[12]	...							...			P0_Y[4]	P2_Y[4]	...	P1076_Y[4]	P1078_Y[4]	...							
UDP[11]	...							...			P0_Y[3]	P2_Y[3]	...	P1076_Y[3]	P1078_Y[3]	...	1	...	1				
UDP[10]	...							...			P0_Y[2]	P2_Y[2]	...	P1076_Y[2]	P1078_Y[2]	...					1	...	1
UDP[9]	...			1	...	1		...			P0_Y[1]	P2_Y[1]	...	P1076_Y[1]	P1078_Y[1]	...							
UDP[8]	...			PKT[8]	...	PKT[8]		...			P0_Y[0]	P2_Y[0]	...	P1076_Y[0]	P1078_Y[0]	...							
UDP[7]	...			PKT[7]	...	PKT[7]		...			P0_Cb[7]	P2_Cb[7]	...	P1076_Cb[7]	P1078_Cb[7]	...							
UDP[6]	...			PKT[6]	...	PKT[6]		...			P0_Cb[6]	P2_Cb[6]	...	P1076_Cb[6]	P1078_Cb[6]	...							
UDP[5]	...			PKT[5]	...	PKT[5]		...			P0_Cb[5]	P2_Cb[5]	...	P1076_Cb[5]	P1078_Cb[5]	...							
UDP[4]	...			PKT[4]	...	PKT[4]		...			P0_Cb[4]	P2_Cb[4]	...	P1076_Cb[4]	P1078_Cb[4]	...							
UDP[3]	...			PKT[3]	...	PKT[3]		...			P0_Cb[3]	P2_Cb[3]	...	P1076_Cb[3]	P1078_Cb[3]	...							
UDP[2]	...			PKT[2]	...	PKT[2]		...			P0_Cb[2]	P2_Cb[2]	...	P1076_Cb[2]	P1078_Cb[2]	...							
UDP[1]	...			PKT[1]	...	PKT[1]		...			P0_Cb[1]	P2_Cb[1]	...	P1076_Cb[1]	P1078_Cb[1]	...							
UDP[0]	...			PKT[0]	...	PKT[0]		...			P0_Cb[0]	P2_Cb[0]	...	P1076_Cb[0]	P1078_Cb[0]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
▨ Video
🟪 Packet Data Enable

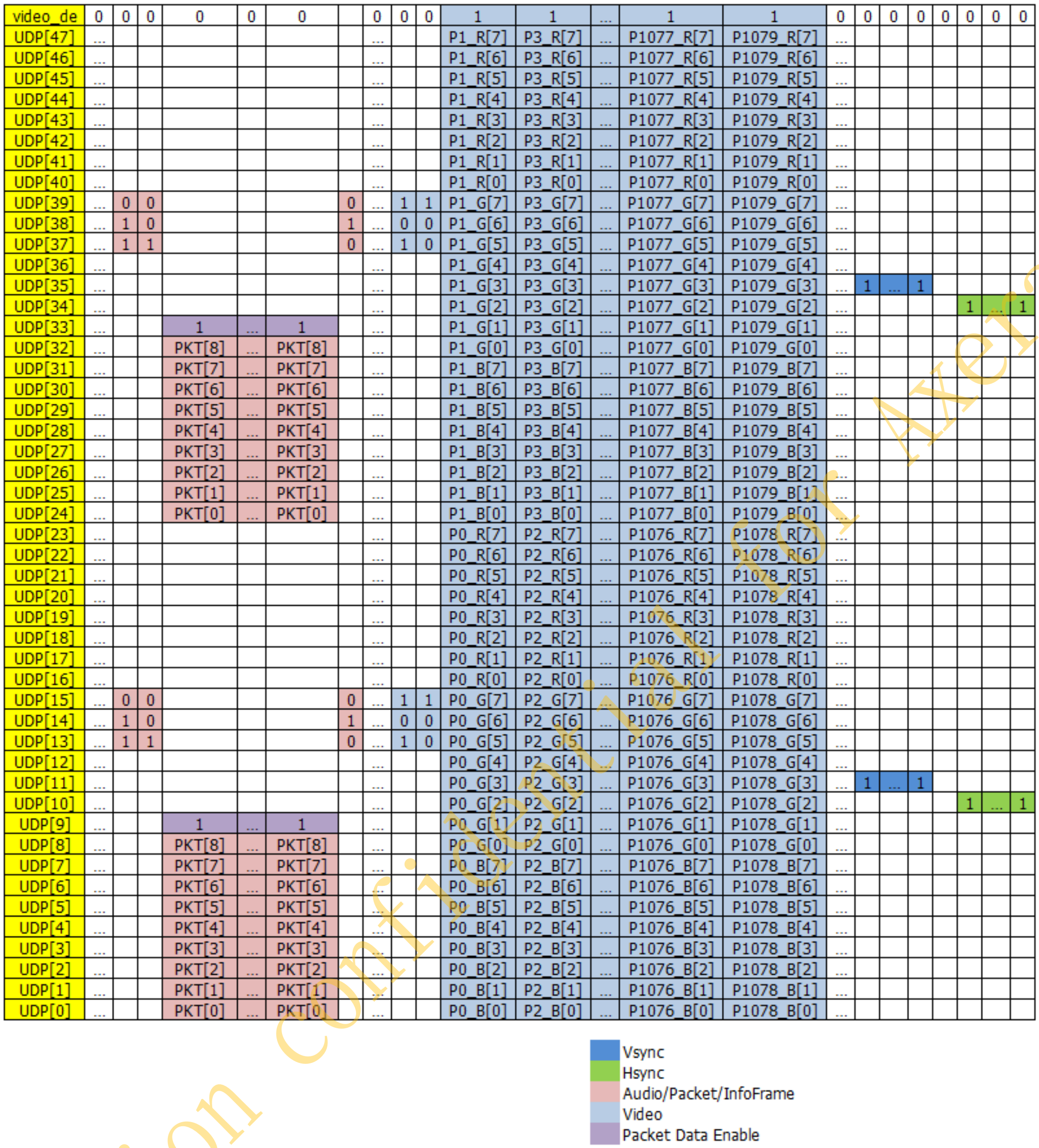
UDP mode Single Pixel TTL pin mapping demo
(YCbCr444 12-bit deep color, 1080p@60Hz)



video_de	0	0	0	0	0	0		0	0	0	1	1	1	...	1	1	1	0	0	0	0	0	0	0	0
UDP[23]	...							...			P0_Cr[7]	P1_Cr[3]	P1_Cr[11]	...	P1078_Cr[7]	P1079_Cr[3]	P1079_Cr[11]	...							
UDP[22]	...							...			P0_Cr[6]	P1_Cr[2]	P1_Cr[10]	...	P1078_Cr[6]	P1079_Cr[2]	P1079_Cr[10]	...							
UDP[21]	...							...			P0_Cr[5]	P1_Cr[1]	P1_Cr[9]	...	P1078_Cr[5]	P1079_Cr[1]	P1079_Cr[9]	...							
UDP[20]	...							...			P0_Cr[4]	P1_Cr[0]	P1_Cr[8]	...	P1078_Cr[4]	P1079_Cr[0]	P1079_Cr[8]	...							
UDP[19]	...							...			P0_Cr[3]	P0_Cr[11]	P1_Cr[7]	...	P1078_Cr[3]	P1078_Cr[11]	P1079_Cr[7]	...							
UDP[18]	...							...			P0_Cr[2]	P0_Cr[10]	P1_Cr[6]	...	P1078_Cr[2]	P1078_Cr[10]	P1079_Cr[6]	...							
UDP[17]	...							...			P0_Cr[1]	P0_Cr[9]	P1_Cr[5]	...	P1078_Cr[1]	P1078_Cr[9]	P1079_Cr[5]	...							
UDP[16]	...							...			P0_Cr[0]	P0_Cr[8]	P1_Cr[4]	...	P1078_Cr[0]	P1078_Cr[8]	P1079_Cr[4]	...							
UDP[15]	...	0	0				0	...	1	1	P0_Y[7]	P1_Y[3]	P1_Y[11]	...	P1078_Y[7]	P1079_Y[3]	P1079_Y[11]	...							
UDP[14]	...	1	0				1	...	0	0	P0_Y[6]	P1_Y[2]	P1_Y[10]	...	P1078_Y[6]	P1079_Y[2]	P1079_Y[10]	...							
UDP[13]	...	1	1				0	...	1	0	P0_Y[5]	P1_Y[1]	P1_Y[9]	...	P1078_Y[5]	P1079_Y[1]	P1079_Y[9]	...							
UDP[12]	...							...			P0_Y[4]	P1_Y[0]	P1_Y[8]	...	P1078_Y[4]	P1079_Y[0]	P1079_Y[8]	...							
UDP[11]	...							...			P0_Y[3]	P0_Y[11]	P1_Y[7]	...	P1078_Y[3]	P1078_Y[11]	P1079_Y[7]	...	1	...	1				
UDP[10]	...							...			P0_Y[2]	P0_Y[10]	P1_Y[6]	...	P1078_Y[2]	P1078_Y[10]	P1079_Y[6]	...					1	...	1
UDP[9]	...			1	...	1		...			P0_Y[1]	P0_Y[9]	P1_Y[5]	...	P1078_Y[1]	P1078_Y[9]	P1079_Y[5]	...							
UDP[8]	...			PKT[8]	...	PKT[8]		...			P0_Y[0]	P0_Y[8]	P1_Y[4]	...	P1078_Y[0]	P1078_Y[8]	P1079_Y[4]	...							
UDP[7]	...			PKT[7]	...	PKT[7]		...			P0_Cb[7]	P1_Cb[3]	P1_Cb[11]	...	P1078_Cb[7]	P1079_Cb[3]	P1079_Cb[11]	...							
UDP[6]	...			PKT[6]	...	PKT[6]		...			P0_Cb[6]	P1_Cb[2]	P1_Cb[10]	...	P1078_Cb[6]	P1079_Cb[2]	P1079_Cb[10]	...							
UDP[5]	...			PKT[5]	...	PKT[5]		...			P0_Cb[5]	P1_Cb[1]	P1_Cb[9]	...	P1078_Cb[5]	P1079_Cb[1]	P1079_Cb[9]	...							
UDP[4]	...			PKT[4]	...	PKT[4]		...			P0_Cb[4]	P1_Cb[0]	P1_Cb[8]	...	P1078_Cb[4]	P1079_Cb[0]	P1079_Cb[8]	...							
UDP[3]	...			PKT[3]	...	PKT[3]		...			P0_Cb[3]	P0_Cb[11]	P1_Cb[7]	...	P1078_Cb[3]	P1078_Cb[11]	P1079_Cb[7]	...							
UDP[2]	...			PKT[2]	...	PKT[2]		...			P0_Cb[2]	P0_Cb[10]	P1_Cb[6]	...	P1078_Cb[2]	P1078_Cb[10]	P1079_Cb[6]	...							
UDP[1]	...			PKT[1]	...	PKT[1]		...			P0_Cb[1]	P0_Cb[9]	P1_Cb[5]	...	P1078_Cb[1]	P1078_Cb[9]	P1079_Cb[5]	...							
UDP[0]	...			PKT[0]	...	PKT[0]		...			P0_Cb[0]	P0_Cb[8]	P1_Cb[4]	...	P1078_Cb[0]	P1078_Cb[8]	P1079_Cb[4]	...							

🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
▨ Video
🟪 Packet Data Enable

UDP mode Dual Pixel TTL pin mapping demo
(RGB 8-bit deep color, 1080p@60Hz)



video_de	0	0	0	0	0	0		0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
UDP[47]	...							...			P1_R[7]	P3_R[7]	...	P1077_R[7]	P1079_R[7]	...							
UDP[46]	...							...			P1_R[6]	P3_R[6]	...	P1077_R[6]	P1079_R[6]	...							
UDP[45]	...							...			P1_R[5]	P3_R[5]	...	P1077_R[5]	P1079_R[5]	...							
UDP[44]	...							...			P1_R[4]	P3_R[4]	...	P1077_R[4]	P1079_R[4]	...							
UDP[43]	...							...			P1_R[3]	P3_R[3]	...	P1077_R[3]	P1079_R[3]	...							
UDP[42]	...							...			P1_R[2]	P3_R[2]	...	P1077_R[2]	P1079_R[2]	...							
UDP[41]	...							...			P1_R[1]	P3_R[1]	...	P1077_R[1]	P1079_R[1]	...							
UDP[40]	...							...			P1_R[0]	P3_R[0]	...	P1077_R[0]	P1079_R[0]	...							
UDP[39]	...	0	0				0	...	1	1	P1_G[7]	P3_G[7]	...	P1077_G[7]	P1079_G[7]	...							
UDP[38]	...	1	0				1	...	0	0	P1_G[6]	P3_G[6]	...	P1077_G[6]	P1079_G[6]	...							
UDP[37]	...	1	1				0	...	1	0	P1_G[5]	P3_G[5]	...	P1077_G[5]	P1079_G[5]	...							
UDP[36]	...							...			P1_G[4]	P3_G[4]	...	P1077_G[4]	P1079_G[4]	...							
UDP[35]	...							...			P1_G[3]	P3_G[3]	...	P1077_G[3]	P1079_G[3]	...	1	...	1				
UDP[34]	...							...			P1_G[2]	P3_G[2]	...	P1077_G[2]	P1079_G[2]	...					1	...	1
UDP[33]	...			1	...	1		...			P1_G[1]	P3_G[1]	...	P1077_G[1]	P1079_G[1]	...							
UDP[32]	...			PKT[8]	...	PKT[8]		...			P1_G[0]	P3_G[0]	...	P1077_G[0]	P1079_G[0]	...							
UDP[31]	...			PKT[7]	...	PKT[7]		...			P1_B[7]	P3_B[7]	...	P1077_B[7]	P1079_B[7]	...							
UDP[30]	...			PKT[6]	...	PKT[6]		...			P1_B[6]	P3_B[6]	...	P1077_B[6]	P1079_B[6]	...							
UDP[29]	...			PKT[5]	...	PKT[5]		...			P1_B[5]	P3_B[5]	...	P1077_B[5]	P1079_B[5]	...							
UDP[28]	...			PKT[4]	...	PKT[4]		...			P1_B[4]	P3_B[4]	...	P1077_B[4]	P1079_B[4]	...							
UDP[27]	...			PKT[3]	...	PKT[3]		...			P1_B[3]	P3_B[3]	...	P1077_B[3]	P1079_B[3]	...							
UDP[26]	...			PKT[2]	...	PKT[2]		...			P1_B[2]	P3_B[2]	...	P1077_B[2]	P1079_B[2]	...							
UDP[25]	...			PKT[1]	...	PKT[1]		...			P1_B[1]	P3_B[1]	...	P1077_B[1]	P1079_B[1]	...							
UDP[24]	...			PKT[0]	...	PKT[0]		...			P1_B[0]	P3_B[0]	...	P1077_B[0]	P1079_B[0]	...							
UDP[23]	...							...			P0_R[7]	P2_R[7]	...	P1076_R[7]	P1078_R[7]	...							
UDP[22]	...							...			P0_R[6]	P2_R[6]	...	P1076_R[6]	P1078_R[6]	...							
UDP[21]	...							...			P0_R[5]	P2_R[5]	...	P1076_R[5]	P1078_R[5]	...							
UDP[20]	...							...			P0_R[4]	P2_R[4]	...	P1076_R[4]	P1078_R[4]	...							
UDP[19]	...							...			P0_R[3]	P2_R[3]	...	P1076_R[3]	P1078_R[3]	...							
UDP[18]	...							...			P0_R[2]	P2_R[2]	...	P1076_R[2]	P1078_R[2]	...							
UDP[17]	...							...			P0_R[1]	P2_R[1]	...	P1076_R[1]	P1078_R[1]	...							
UDP[16]	...							...			P0_R[0]	P2_R[0]	...	P1076_R[0]	P1078_R[0]	...							
UDP[15]	...	0	0				0	...	1	1	P0_G[7]	P2_G[7]	...	P1076_G[7]	P1078_G[7]	...							
UDP[14]	...	1	0				1	...	0	0	P0_G[6]	P2_G[6]	...	P1076_G[6]	P1078_G[6]	...							
UDP[13]	...	1	1				0	...	1	0	P0_G[5]	P2_G[5]	...	P1076_G[5]	P1078_G[5]	...							
UDP[12]	...							...			P0_G[4]	P2_G[4]	...	P1076_G[4]	P1078_G[4]	...							
UDP[11]	...							...			P0_G[3]	P2_G[3]	...	P1076_G[3]	P1078_G[3]	...	1	...	1				
UDP[10]	...							...			P0_G[2]	P2_G[2]	...	P1076_G[2]	P1078_G[2]	...					1	...	1
UDP[9]	...			1	...	1		...			P0_G[1]	P2_G[1]	...	P1076_G[1]	P1078_G[1]	...							
UDP[8]	...			PKT[8]	...	PKT[8]		...			P0_G[0]	P2_G[0]	...	P1076_G[0]	P1078_G[0]	...							
UDP[7]	...			PKT[7]	...	PKT[7]		...			P0_B[7]	P2_B[7]	...	P1076_B[7]	P1078_B[7]	...							
UDP[6]	...			PKT[6]	...	PKT[6]		...			P0_B[6]	P2_B[6]	...	P1076_B[6]	P1078_B[6]	...							
UDP[5]	...			PKT[5]	...	PKT[5]		...			P0_B[5]	P2_B[5]	...	P1076_B[5]	P1078_B[5]	...							
UDP[4]	...			PKT[4]	...	PKT[4]		...			P0_B[4]	P2_B[4]	...	P1076_B[4]	P1078_B[4]	...							
UDP[3]	...			PKT[3]	...	PKT[3]		...			P0_B[3]	P2_B[3]	...	P1076_B[3]	P1078_B[3]	...							
UDP[2]	...			PKT[2]	...	PKT[2]		...			P0_B[2]	P2_B[2]	...	P1076_B[2]	P1078_B[2]	...							
UDP[1]	...			PKT[1]	...	PKT[1]		...			P0_B[1]	P2_B[1]	...	P1076_B[1]	P1078_B[1]	...							
UDP[0]	...			PKT[0]	...	PKT[0]		...			P0_B[0]	P2_B[0]	...	P1076_B[0]	P1078_B[0]	...							

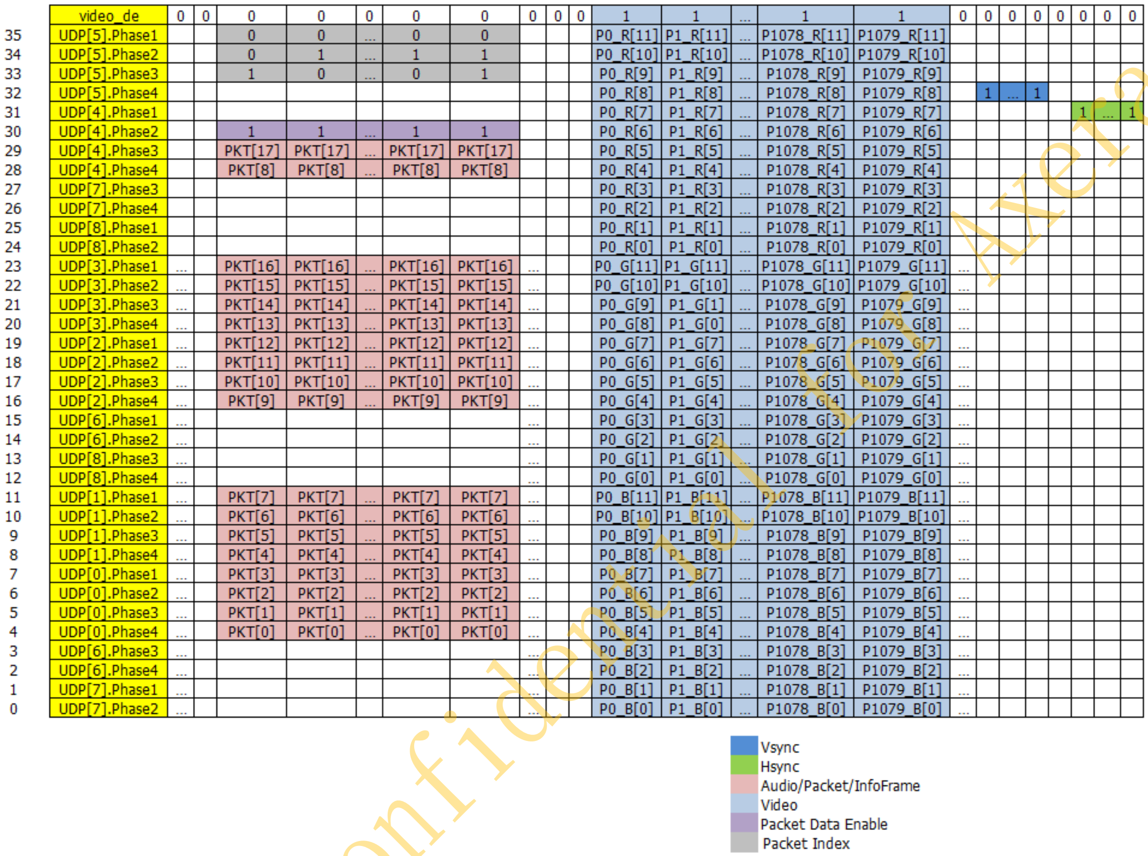
🟦 Vsync
🟩 Hsync
🟥 Audio/Packet/InfoFrame
🟦 Video
🟪 Packet Data Enable

1.6 HDMI to LVDS in UDP deep-color decoding mode

UDP application in Section 1.4 and 1.5 will not decode deep-color to generate 30/36-bit bus width for LVDS/TTL interface. If deep-color decoding is needed for GSV2011 to LVDS/TTL conversion, an extra UDP deep-color decoding mode can be supported.
The limitation of this mode:
1, Maximum supported timing frequency is limited to 4K30.
2, Only support HDMI Rx->LVDS/TTL Tx support. No LVDS/TTL Rx-> HDMI Tx support.

A 36-bit parallel bus is mapped for TTL. If using LVDS 4x mode, a 9-lane parallel bus is mapped for LVDS.

UDP mode Single Pixel 4x LVDS deep-color decoding pin mapping demo
(YCbCr444 12-bit deep color, 1080p@60Hz)



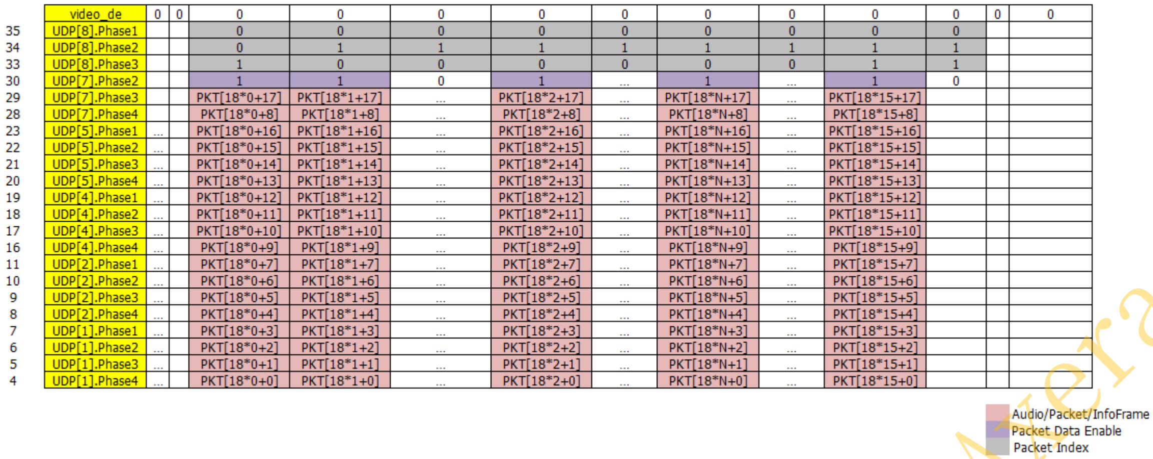
	video_de	0	0	0	0	0	0	0	0	0	0	1	1	...	1	1	0	0	0	0	0	0	0	0
35	UDP[5].Phase1			0	0	...	0	0				P0_R[11]	P1_R[11]	...	P1078_R[11]	P1079_R[11]								
34	UDP[5].Phase2			0	1	...	1	1				P0_R[10]	P1_R[10]	...	P1078_R[10]	P1079_R[10]								
33	UDP[5].Phase3			1	0	...	0	1				P0_R[9]	P1_R[9]	...	P1078_R[9]	P1079_R[9]								
32	UDP[5].Phase4											P0_R[8]	P1_R[8]	...	P1078_R[8]	P1079_R[8]		1	...	1				
31	UDP[4].Phase1											P0_R[7]	P1_R[7]	...	P1078_R[7]	P1079_R[7]						1	...	1
30	UDP[4].Phase2			1	1	...	1	1				P0_R[6]	P1_R[6]	...	P1078_R[6]	P1079_R[6]								
29	UDP[4].Phase3			PKT[17]	PKT[17]	...	PKT[17]	PKT[17]				P0_R[5]	P1_R[5]	...	P1078_R[5]	P1079_R[5]								
28	UDP[4].Phase4			PKT[8]	PKT[8]	...	PKT[8]	PKT[8]				P0_R[4]	P1_R[4]	...	P1078_R[4]	P1079_R[4]								
27	UDP[7].Phase3											P0_R[3]	P1_R[3]	...	P1078_R[3]	P1079_R[3]								
26	UDP[7].Phase4											P0_R[2]	P1_R[2]	...	P1078_R[2]	P1079_R[2]								
25	UDP[8].Phase1											P0_R[1]	P1_R[1]	...	P1078_R[1]	P1079_R[1]								
24	UDP[8].Phase2											P0_R[0]	P1_R[0]	...	P1078_R[0]	P1079_R[0]								
23	UDP[3].Phase1	...		PKT[16]	PKT[16]	...	PKT[16]	PKT[16]	...			P0_G[11]	P1_G[11]	...	P1078_G[11]	P1079_G[11]	...							
22	UDP[3].Phase2	...		PKT[15]	PKT[15]	...	PKT[15]	PKT[15]	...			P0_G[10]	P1_G[10]	...	P1078_G[10]	P1079_G[10]	...							
21	UDP[3].Phase3	...		PKT[14]	PKT[14]	...	PKT[14]	PKT[14]	...			P0_G[9]	P1_G[1]	...	P1078_G[9]	P1079_G[9]	...							
20	UDP[3].Phase4	...		PKT[13]	PKT[13]	...	PKT[13]	PKT[13]	...			P0_G[8]	P1_G[0]	...	P1078_G[8]	P1079_G[8]	...							
19	UDP[2].Phase1	...		PKT[12]	PKT[12]	...	PKT[12]	PKT[12]	...			P0_G[7]	P1_G[7]	...	P1078_G[7]	P1079_G[7]	...							
18	UDP[2].Phase2	...		PKT[11]	PKT[11]	...	PKT[11]	PKT[11]	...			P0_G[6]	P1_G[6]	...	P1078_G[6]	P1079_G[6]	...							
17	UDP[2].Phase3	...		PKT[10]	PKT[10]	...	PKT[10]	PKT[10]	...			P0_G[5]	P1_G[5]	...	P1078_G[5]	P1079_G[5]	...							
16	UDP[2].Phase4	...		PKT[9]	PKT[9]	...	PKT[9]	PKT[9]	...			P0_G[4]	P1_G[4]	...	P1078_G[4]	P1079_G[4]	...							
15	UDP[6].Phase1	...							...			P0_G[3]	P1_G[3]	...	P1078_G[3]	P1079_G[3]	...							
14	UDP[6].Phase2	...							...			P0_G[2]	P1_G[2]	...	P1078_G[2]	P1079_G[2]	...							
13	UDP[8].Phase3	...							...			P0_G[1]	P1_G[1]	...	P1078_G[1]	P1079_G[1]	...							
12	UDP[8].Phase4	...							...			P0_G[0]	P1_G[0]	...	P1078_G[0]	P1079_G[0]	...							
11	UDP[1].Phase1	...		PKT[7]	PKT[7]	...	PKT[7]	PKT[7]	...			P0_B[11]	P1_B[11]	...	P1078_B[11]	P1079_B[11]	...							
10	UDP[1].Phase2	...		PKT[6]	PKT[6]	...	PKT[6]	PKT[6]	...			P0_B[10]	P1_B[10]	...	P1078_B[10]	P1079_B[10]	...							
9	UDP[1].Phase3	...		PKT[5]	PKT[5]	...	PKT[5]	PKT[5]	...			P0_B[9]	P1_B[9]	...	P1078_B[9]	P1079_B[9]	...							
8	UDP[1].Phase4	...		PKT[4]	PKT[4]	...	PKT[4]	PKT[4]	...			P0_B[8]	P1_B[8]	...	P1078_B[8]	P1079_B[8]	...							
7	UDP[0].Phase1	...		PKT[3]	PKT[3]	...	PKT[3]	PKT[3]	...			P0_B[7]	P1_B[7]	...	P1078_B[7]	P1079_B[7]	...							
6	UDP[0].Phase2	...		PKT[2]	PKT[2]	...	PKT[2]	PKT[2]	...			P0_B[6]	P1_B[6]	...	P1078_B[6]	P1079_B[6]	...							
5	UDP[0].Phase3	...		PKT[1]	PKT[1]	...	PKT[1]	PKT[1]	...			P0_B[5]	P1_B[5]	...	P1078_B[5]	P1079_B[5]	...							
4	UDP[0].Phase4	...		PKT[0]	PKT[0]	...	PKT[0]	PKT[0]	...			P0_B[4]	P1_B[4]	...	P1078_B[4]	P1079_B[4]	...							
3	UDP[6].Phase3	...							...			P0_B[3]	P1_B[3]	...	P1078_B[3]	P1079_B[3]	...							
2	UDP[6].Phase4	...							...			P0_B[2]	P1_B[2]	...	P1078_B[2]	P1079_B[2]	...							
1	UDP[7].Phase1	...							...			P0_B[1]	P1_B[1]	...	P1078_B[1]	P1079_B[1]	...							
0	UDP[7].Phase2	...							...			P0_B[0]	P1_B[0]	...	P1078_B[0]	P1079_B[0]	...							

Vsync
Hsync
Audio/Packet/InfoFrame
Video
Packet Data Enable
Packet Index

As every HDMI defined packet/infoframe will be sent via a 18-bit x 16-cycle format. Using TTL/LVDS deep-color decoding UDP mode, the packet_data_enable will always maintain 16 valid cycles for every single packet/infoframe. The packet_data_enable might contain invalid cycles during the 16 valid cycles.

A Packet_index is used to indicate/assist the packet start and packet end.

Packet_index	Description
1	Packet start, this cycle contains the 1st packet_data_enable data, next cycle is started with Packet_index = 2
2	Packet content, packet_data_enable = 1 means valid packet data. A total of 16 packet_data_enable cycles will be within packet_index = 1/2/3 cycles.
3	Packet end, the 1st packet_index = 3 cycle will have packet_data_enable = 1 for indicating the last valid cycle
others	invalid



	video_de	0	0	0	0	0	0	0	0	0	0	0	0	0
35	UDP[8].Phase1			0	0	0	0	0	0	0	0	0		
34	UDP[8].Phase2			0	1	1	1	1	1	1	1	1		
33	UDP[8].Phase3			1	0	0	0	0	0	0	1	1		
30	UDP[7].Phase2			1	1	0	1	...	1	...	1	0		
29	UDP[7].Phase3			PKT[18*0+17]	PKT[18*1+17]	...	PKT[18*2+17]	...	PKT[18*N+17]	...	PKT[18*15+17]			
28	UDP[7].Phase4			PKT[18*0+8]	PKT[18*1+8]	...	PKT[18*2+8]	...	PKT[18*N+8]	...	PKT[18*15+8]			
23	UDP[5].Phase1	...		PKT[18*0+16]	PKT[18*1+16]	...	PKT[18*2+16]	...	PKT[18*N+16]	...	PKT[18*15+16]			
22	UDP[5].Phase2	...		PKT[18*0+15]	PKT[18*1+15]	...	PKT[18*2+15]	...	PKT[18*N+15]	...	PKT[18*15+15]			
21	UDP[5].Phase3	...		PKT[18*0+14]	PKT[18*1+14]	...	PKT[18*2+14]	...	PKT[18*N+14]	...	PKT[18*15+14]			
20	UDP[5].Phase4	...		PKT[18*0+13]	PKT[18*1+13]	...	PKT[18*2+13]	...	PKT[18*N+13]	...	PKT[18*15+13]			
19	UDP[4].Phase1	...		PKT[18*0+12]	PKT[18*1+12]	...	PKT[18*2+12]	...	PKT[18*N+12]	...	PKT[18*15+12]			
18	UDP[4].Phase2	...		PKT[18*0+11]	PKT[18*1+11]	...	PKT[18*2+11]	...	PKT[18*N+11]	...	PKT[18*15+11]			
17	UDP[4].Phase3	...		PKT[18*0+10]	PKT[18*1+10]	...	PKT[18*2+10]	...	PKT[18*N+10]	...	PKT[18*15+10]			
16	UDP[4].Phase4	...		PKT[18*0+9]	PKT[18*1+9]	...	PKT[18*2+9]	...	PKT[18*N+9]	...	PKT[18*15+9]			
11	UDP[2].Phase1	...		PKT[18*0+7]	PKT[18*1+7]	...	PKT[18*2+7]	...	PKT[18*N+7]	...	PKT[18*15+7]			
10	UDP[2].Phase2	...		PKT[18*0+6]	PKT[18*1+6]	...	PKT[18*2+6]	...	PKT[18*N+6]	...	PKT[18*15+6]			
9	UDP[2].Phase3	...		PKT[18*0+5]	PKT[18*1+5]	...	PKT[18*2+5]	...	PKT[18*N+5]	...	PKT[18*15+5]			
8	UDP[2].Phase4	...		PKT[18*0+4]	PKT[18*1+4]	...	PKT[18*2+4]	...	PKT[18*N+4]	...	PKT[18*15+4]			
7	UDP[1].Phase1	...		PKT[18*0+3]	PKT[18*1+3]	...	PKT[18*2+3]	...	PKT[18*N+3]	...	PKT[18*15+3]			
6	UDP[1].Phase2	...		PKT[18*0+2]	PKT[18*1+2]	...	PKT[18*2+2]	...	PKT[18*N+2]	...	PKT[18*15+2]			
5	UDP[1].Phase3	...		PKT[18*0+1]	PKT[18*1+1]	...	PKT[18*2+1]	...	PKT[18*N+1]	...	PKT[18*15+1]			
4	UDP[1].Phase4	...		PKT[18*0+0]	PKT[18*1+0]	...	PKT[18*2+0]	...	PKT[18*N+0]	...	PKT[18*15+0]			

🟥 Audio/Packet/InfoFrame
🟪 Packet Data Enable
▨ Packet Index

Following steps are required for UDP mode Single Pixel 4x LVDS deep-color decoding mode.

1, set "***0x04,0x03,0xA2***" configuration for single pixel UDP 12-bit configuration in *ParallelConfigTable[]* of *av_common.c* file.

2, edit following red code into *AvUapiCheckLogicVideoTx()* of *gsv2k11.c* file.

```
/* UDP Configuration in mux mode */
FromPort = (AvPort*)(port->content.RouteVideoFromPort);
if((ParallelConfigTable[Offset+2] & 0x20) != 0)
{
  ~~//Value = 5;~~
  ~~//AvHalI2cWriteField8(GSV2K11_PAR_MAP_ADDR(port),0x64,0xFF,0,0x80);~~
  ~~//AvHalI2cWriteField8(GSV2K11_PAR_MAP_ADDR(port),0x66,0xFF,0,0x00);~~
  ~~//AvHalI2cWriteField8(GSV2K11_PAR_MAP_ADDR(port),0x69,0xFF,0,0x09);~~
  Value = 0;
  AvHalI2cWriteField8(GSV2K11_PAR_MAP_ADDR(port),0x64,0xFF,0,0x01);
  AvHalI2cWriteField8(GSV2K11_PAR_MAP_ADDR(port),0x66,0xFF,0,0x00);
  AvHalI2cWriteField8(GSV2K11_PAR_MAP_ADDR(port),0x69,0xFF,0,0x09);
}
```

1.7 UDP in 1.2Gbps Fixed Rate mode

To relieve the clock design complexity on the receiving circuit in FPGA, there is a fixed 1.2Gbps data rate mode in GSV2011 UDP application. LVDS data rate is fixed to 1.2Gbps regardless of input timing. LVDS clock can be separately configured according to Section 4.1.

Packet_data_enable and video_de will be discrete in 1.2Gbps fixed lane rate mode, and these signals are indicator of the valid packet/video stream data. Valid phase of UDP data is indicated by the separate HS differential pair.

2. Parallel Pin Characteristic Description

2.1 TTL Tx Characteristic

GSV2011 Tx TTL parallel bus supports slew rate manual control and drive strength manual control. A total 16 steps with 50ps/step manual control is supported by groups. A 0 degree and 90 degree phase relationship between clock and all data TTL pins are supported. To have a better clock receiving performance, DDR mode is recommended to lower the clock frequency by 1/2.

2.2 TTL Rx Characteristic

GSV2011 Rx TTL parallel bus supports 0 degree and 90 degree phase relationship between clock and all data TTL pins.

2.3 TTL Layout Recommendation

TTL pins should have 50 ohm single-ended resistance, 2~3w between TTL pins, the total mismatch should be < 200mil. For GSV2011 GND, it is recommended to have TTL GND and HDMI Rx/Tx GND separated, more decaps around HDMI ports will greatly help relieve TTL SSN impact on HDMI input.

2.4 LVDS Tx Characteristic

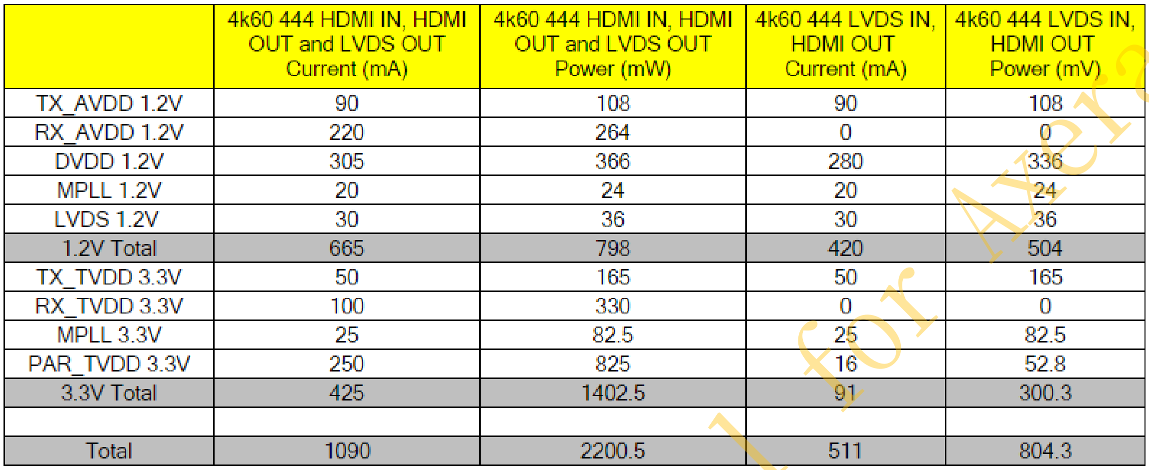
GSV2011 Tx LVDS parallel bus supports slew rate manual control, termination manual control, pre-emphasis control, common mode voltage control and drive strength manual control.
A total 16 steps with 50ps/step manual control is supported per differential pair.
A 0 degree and 90 degree phase relationship between clock and all data differential pairs are supported. To have a better clock receiving performance, DDR mode is recommended to lower the clock frequency by 1/2.
There is a limited +/- 50ps phase uncertainty between LVDS clock and each LVDS data pair. When calculating and determining timing constraints for LVDS Rx side (in FPGA), PCB inter-pair trace length difference of LVDS should also be included in the total phase difference by 6mil->1ps relationship.

2.5 LVDS Rx Characteristic

For LVDS Rx, in typical application, clock input RMS jitter is 10ps. GSV2011 can tolerate +/-200ps data/clock uncertainty for receiving LVDS input data.

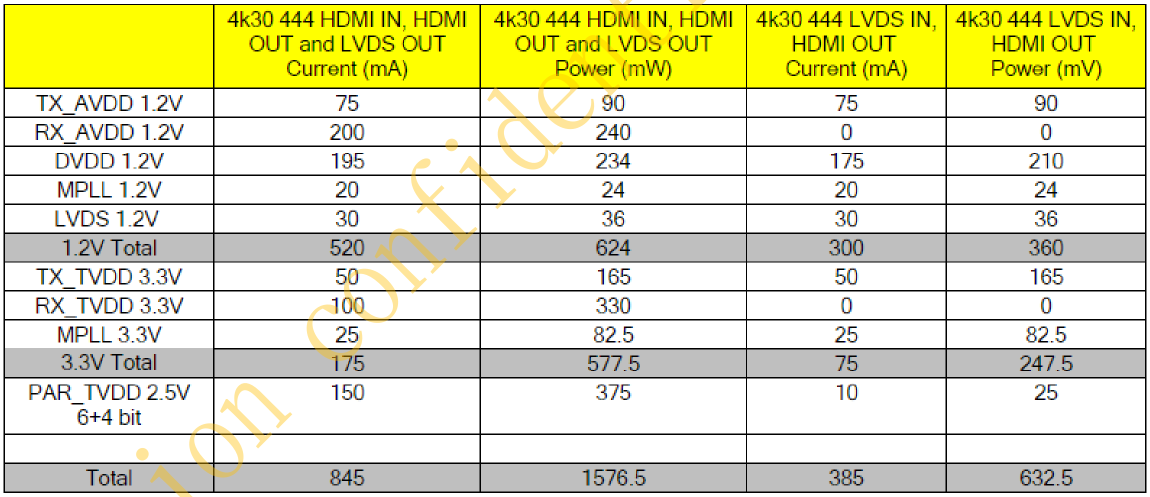
3. Power Consumption Description

The estimated highest power of 4K60 444 is given below.



	4k60 444 HDMI IN, HDMI OUT and LVDS OUT Current (mA)	4k60 444 HDMI IN, HDMI OUT and LVDS OUT Power (mW)	4k60 444 LVDS IN, HDMI OUT Current (mA)	4k60 444 LVDS IN, HDMI OUT Power (mV)
TX_AVDD 1.2V	90	108	90	108
RX_AVDD 1.2V	220	264	0	0
DVDD 1.2V	305	366	280	336
MPLL 1.2V	20	24	20	24
LVDS 1.2V	30	36	30	36
1.2V Total	665	798	420	504
TX_TVDD 3.3V	50	165	50	165
RX_TVDD 3.3V	100	330	0	0
MPLL 3.3V	25	82.5	25	82.5
PAR_TVDD 3.3V	250	825	16	52.8
3.3V Total	425	1402.5	91	300.3
Total	1090	2200.5	511	804.3

The estimated highest power of 4K30 444 is given below.



	4k30 444 HDMI IN, HDMI OUT and LVDS OUT Current (mA)	4k30 444 HDMI IN, HDMI OUT and LVDS OUT Power (mW)	4k30 444 LVDS IN, HDMI OUT Current (mA)	4k30 444 LVDS IN, HDMI OUT Power (mV)
TX_AVDD 1.2V	75	90	75	90
RX_AVDD 1.2V	200	240	0	0
DVDD 1.2V	195	234	175	210
MPLL 1.2V	20	24	20	24
LVDS 1.2V	30	36	30	36
1.2V Total	520	624	300	360
TX_TVDD 3.3V	50	165	50	165
RX_TVDD 3.3V	100	330	0	0
MPLL 3.3V	25	82.5	25	82.5
3.3V Total	175	577.5	75	247.5
PAR_TVDD 2.5V 6+4 bit	150	375	10	25
Total	845	1576.5	385	632.5

4. Software Development Questions

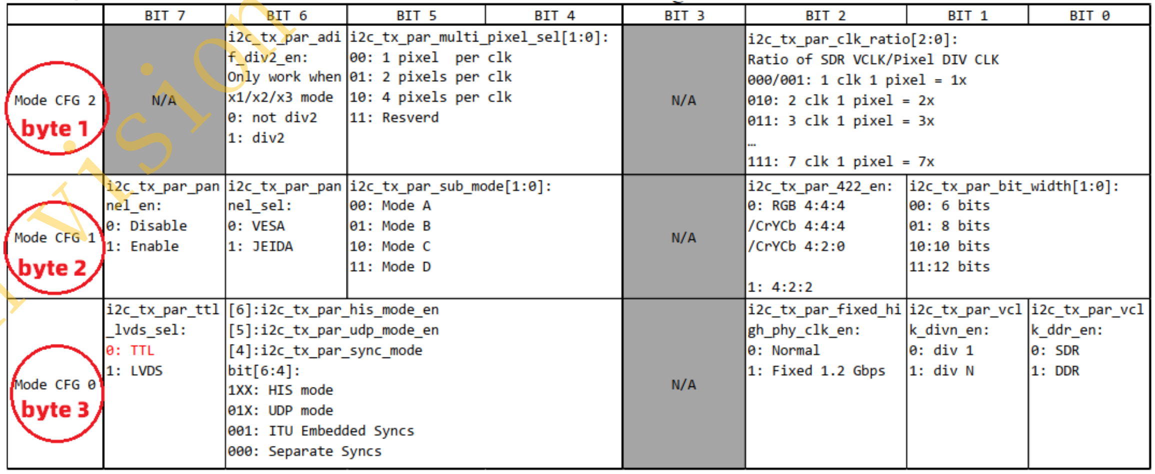
Please refer to <GSV software User Guide> for general GSV software integration questions. The following questions are GSV2011 parallel bus related and not related to other chips.

4.1 How to set specific parallel bus setting in software?

In *apps/av_common.c* file, there is a table called *ParallelConfigTable[]* which lists commonly supported parallel bus settings.
The table looks like below:

```
const uint8 ParallelConfigTable[] = {
    0x00, 0x00, 0x00,    // 0: Invalid Setting for disabled Parallel bus
    0x04, 0x03, 0x82,    // 1: Index 21, 4x single pixel 12-bit 444
    0x14, 0x01, 0x82,    // 2: Index 61, 4x dual pixel 8-bit 444
    0x04, 0x05, 0xC1,    // 3: Index 58-1, HIS mode in DDR mode
    0x14, 0x01, 0xA2,    // 4: 4x dual pixel UDP mode using LVDS
    0x01, 0x07, 0x01,    // 5: TTL 36-bit, DDR mode, YCbCr 444
    0x01, 0x15, 0x10,    // 6: TTL BT.1120 16-bit, SDR mode, YCbCr 422
    0x11, 0x01, 0x01,    // 7: TTL 48-bit, DDR mode
    0x01, 0x15, 0x11,    // 8: TTL BT.1120 16-bit, DDR mode, YCbCr 422
    0x11, 0x01, 0x21,    // 9: dual pixel UDP mode using TTL
    0x01, 0x15, 0x00,    // 10: TTL BT.1120 16-bit, SDR mode, YCbCr 422, separate sync
    0xFF, 0xFF, 0xFF
};
```

Each parallel bus setting is listed in 3 bytes of each line. The index = 0 (first line, 3 bytes of *0x00,0x00,0x00*) are the default setting to disable the parallel bus. Since index = 1 (second line, 3 bytes of *0x04,0x03,0x82*), the index number is used for valid parallel bus setting.



	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Mode CFG 2 byte 1	N/A	i2c_tx_par_adif_div2_en: Only work when x1/x2/x3 mode 0: not div2 1: div2	i2c_tx_par_multi_pixel_sel[1:0]: 00: 1 pixel per clk 01: 2 pixels per clk 10: 4 pixels per clk 11: Resverd		N/A	i2c_tx_par_clk_ratio[2:0]: Ratio of SDR VCLK/Pixel DIV CLK 000/001: 1 clk 1 pixel = 1x 010: 2 clk 1 pixel = 2x 011: 3 clk 1 pixel = 3x … 111: 7 clk 1 pixel = 7x		
Mode CFG 1 byte 2	i2c_tx_par_pannel_en: 0: Disable 1: Enable	i2c_tx_par_pannel_sel: 0: VESA 1: JEIDA	i2c_tx_par_sub_mode[1:0]: 00: Mode A 01: Mode B 10: Mode C 11: Mode D		N/A	i2c_tx_par_422_en: 0: RGB 4:4:4 /CrYCb 4:4:4 /CrYCb 4:2:0 1: 4:2:2	i2c_tx_par_bit_width[1:0]: 00: 6 bits 01: 8 bits 10:10 bits 11:12 bits	
Mode CFG 0 byte 3	i2c_tx_par_ttl_lvds_sel: 0: TTL 1: LVDS	[6]:i2c_tx_par_his_mode_en [5]:i2c_tx_par_udp_mode_en [4]:i2c_tx_par_sync_mode bit[6:4]: 1XX: HIS mode 01X: UDP mode 001: ITU Embedded Syncs 000: Separate Syncs			N/A	i2c_tx_par_fixed_high_phy_clk_en: 0: Normal 1: Fixed 1.2 Gbps	i2c_tx_par_vclk_divn_en: 0: div 1 1: div N	i2c_tx_par_vclk_ddr_en: 0: SDR 1: DDR

To add a dedicated parallel bus setting, refer to <GSV2011 datasheet>. In Section 4.1 and Section 4.2, there are multiple settings regarding Mode CFG 2/1/0. The 3 bytes of each

line in *ParallelConfigTable[]*, directly mapped from Mode CFG 2/1/0. The Pin mapping tables in datasheet have already given example settings for common used parallel bus configuration.
Software designer can feel free to add more 3-byte settings into the *ParallelConfigTable[]* table.

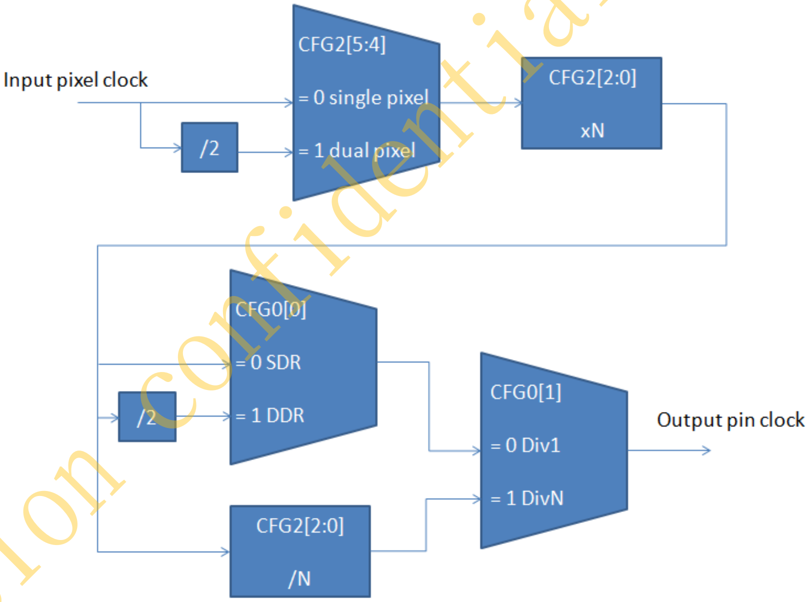
At any position in the code (usually during the *GsvMain()* of *av_main.c*), set the ParallelBusPort setting to accommodate the setting in *ParallelConfigTable[]*.
For example, to set Parallel bus output, set like below in av_main.c:

```
AvApiAddPort(&devices[0],&gsv2k11Ports[2] ,32,LogicVideoTx); // add LogicVideoTxPort
uint8 CommonBusConfig = Index_Number_in_ParallelConfigTable; // Set Index
gsv2k11Ports[2].content.lvtx->Config       = CommonBusConfig; // Set Parallel Port config as index
gsv2k11Ports[2].content.lvtx->Update       = 1; // Update = 1 to update Parallel Port config into chip
```

For example, to set Parallel bus input, set like below in av_main.c:

```
AvApiAddPort(&devices[0],&gsv2k11Ports[7] ,30,LogicVideoRx); // add LogicVideoRxPort
uint8 CommonBusConfig = Index_Number_in_ParallelConfigTable; // Set Index
gsv2k11Ports[7].content.lvtx->Config       = CommonBusConfig; // Set Parallel Port config as index
gsv2k11Ports[7].content.lvtx->Update       = 1; // Update = 1 to update Parallel Port config into chip
```

The output pin clock frequency relationship is given below.



4.1.1 An example of GSV2011 LVDS VESA mode application

1, confirm the current single pixel VESA mode has been added into apps/av_common.c.

```
const uint8 ParallelConfigTable[] = {
    0x00, 0x00, 0x00,    // 0: Invalid Setting for disabled Parallel bus
    0x04, 0x03, 0x82,    // 1: Index 21, 4x single pixel 12-bit 444
    0x14, 0x01, 0x82,    // 2: Index 61, 4x dual pixel 8-bit 444
    0x04, 0x05, 0xC1,     // 3: Index 58-1, HIS mode in DDR mode
    0x14, 0x01, 0xA2,     // 4: 4x dual pixel UDP mode using LVDS
    0x01, 0x07, 0x01,    // 5: TTL 36-bit, DDR mode, YCbCr 444
    0x01, 0x15, 0x10,    // 6: TTL BT.1120 16-bit, SDR mode, YCbCr 422
    0x11, 0x01, 0x01,    // 7: TTL 48-bit, DDR mode
    0x01, 0x15, 0x11,    // 8: TTL BT.1120 16-bit, DDR mode, YCbCr 422
    0x11, 0x01, 0x21,    // 9: dual pixel UDP mode using TTL
    0x01, 0x15, 0x00,    // 10: TTL BT.1120 16-bit, SDR mode, YCbCr 422, separate sync
    0x07, 0x81, 0x82,    // 11: LVDS VESA configuration, 8-bit
    0xFF, 0xFF, 0xFF
};
```

2, choose the new setting in apps/av_main.c file.

```
    /* 3.4.5 Video Parallel Bus Input */
    /* CommonBusConfig = 0 to disable, CommonBusConfig = 1~64 for feature setting */
    uint8 CommonBusConfig = 11; // LVDS VESA configuration, 8-bit
    LvdsTxPort.content.lvtx->Config        = CommonBusConfig;
```

3， set color space in apps/av_main.c file.

```
LogicVideoTxPort->content.video->Y = AV_Y2Y1Y0_RGB;
LogicVideoTxPort->content.video->InCs = AV_CS_RGB;
```

4, Add following settings into *Gsv2k11InitTable[]* of *gsv2k11_table.h* if only if VESA clock phase adjustment is needed.

```
0x14,0x80,0x87,    // manual control clock output with 7x clock ratio
0x14,0x81,0x1E,    // set the clock phase to 00_1111_0, default is 1111_000
0x14,0x69,0x0B,    // set HS to copy CLK output for dual pixel VESA/JEIDA output
```

5, In *Gsv2k11ParTxTable[]* of *gsv2k11_table.h*, if dual pixel VESA/JEIDA output is required, modification should be applied to PAR.0x69 as well.

```
    0x66,0x00, // Tx Par Pll Setting
    0x69,0x0B, // Tx Par Pll Setting, modified from 0x09
```

4.2 How to set Tx parallel bus setting in software?

There are several setting for constraining Tx parallel bus's color space and timing.

LogicVideoTxPort->content.video->Y = Desired_Color_Space;

Desired_Color_Space	Description
AV_Y2Y1Y0_INVALID	Parallel Port output color space matches HDMI Rx input color space, no internal color space conversion
AV_Y2Y1Y0_RGB	Parallel Port output color space is RGB. Because of chip limitation, if HDMI Rx is YCbCr 420, Parallel Port output color space will be *AV_Y2Y1Y0_YCBCR_444* instead.
AV_Y2Y1Y0_YCBCR_422	Parallel Port output color space is YCbCr 422. Default color space setting for BT656 and BT1120.
AV_Y2Y1Y0_YCBCR_444	Parallel Port output color space is YCbCr 444. Recommended default setting.
AV_Y2Y1Y0_YCBCR_420	Parallel Port output color space is YCbCr 420.

LogicVideoTxPort->content.video->InCs = Desired_Color_Range;

Desired_Color_Range	Description
AV_CS_AUTO	Parallel Port output color range matches HDMI Rx input color range, no internal processing.
AV_CS_RGB	Parallel Port output color range is full range RGB.
AV_CS_YUV_601	Parallel Port output color range is full range BT601 YUV.
AV_CS_YUV_709	Parallel Port output color range is full range BT709 YUV.
AV_CS_LIM_RGB	Parallel Port output color range is limited range RGB.
AV_CS_LIM_YUV_601	Parallel Port output color range is limited range BT601 YUV.
AV_CS_LIM_YUV_709	Parallel Port output color range is limited range BT709 YUV.

LogicVideoTxPort->content.video->info.TmdsFreq = Desired_Maxinum_Pixel_Clock;

Desired_Maxinum_Pixel_ Clock	Description
0	Default setting, maximum pixel clock is 600MHz for full HDMI 2.0 timings as HDMI Rx input.
600	Default setting, maximum pixel clock is 600MHz for full HDMI 2.0 timings as HDMI Rx input.
300	4K50/60 will be down-scaled to 1080p50/60.
150	All 4K timings will be downscaled to 2K. 1080p is maximum supported timing.

4.3 How to set Rx parallel bus setting in software?

LogicVideoRxPort->content.video->InCs = Desired_Color_Range;

Desired_Color_Range	Description
AV_CS_RGB	Parallel Port input color range is full range RGB.
AV_CS_YUV_601	Parallel Port input color range is full range BT601 YUV.
AV_CS_YUV_709	Parallel Port input color range is full range BT709 YUV.
AV_CS_LIM_RGB	Parallel Port input color range is limited range RGB.
AV_CS_LIM_YUV_601	Parallel Port input color range is limited range BT601 YUV.
AV_CS_LIM_YUV_709	Parallel Port input color range is limited range BT709 YUV.
AV_CS_SYCC_601	Parallel Port input color range is Full range SYCC 601
AV_CS_ADOBE_YCC_601	Parallel Port input color range is Full range Adobe YCC 601
AV_CS_ADOBE_RGB	Parallel Port input color range is Full range Adobe RGB
AV_CS_YCC_601	Parallel Port input color range is Full range YCC 601
AV_CS_YCC_709	Parallel Port input color range is Full range YCC 709
AV_CS_BT2020_YCC	Parallel Port input color range is Full range BT2020 YCC
AV_CS_BT2020_RGB	Parallel Port input color range is Full range BT2020 RGB
AV_CS_LIM_YCC_601	Parallel Port input color range is Limited range YCC 601
AV_CS_LIM_YCC_709	Parallel Port input color range is Limited range YCC 709
AV_CS_LIM_SYCC_601	Parallel Port input color range is Limited range SYCC 601
AV_CS_LIM_ADOBE_YCC	Parallel Port input color range is Limited Range Adobe

_601	YCC 601
AV_CS_LIM_ADOBE_RGB	Parallel Port input color range is Limited range Adobe RGB
AV_CS_LIM_BT2020_YCC	Parallel Port input color range is Limited Range BT2020 YCC
AV_CS_LIM_BT2020_RGB	Parallel Port input color range is Limited range BT2020 RGB.

LogicVideoRxPort->content.video->Y = Desired_Color_Space;

Desired_Color_Space	Description
AV_Y2Y1Y0_RGB	Parallel Port input color space is RGB.
AV_Y2Y1Y0_YCBCR_422	Parallel Port input color space is YCbCr 422. Default color space setting for BT656 and BT1120.
AV_Y2Y1Y0_YCBCR_444	Parallel Port input color space is YCbCr 444. Recommended default setting.
AV_Y2Y1Y0_YCBCR_420	Parallel Port input color space is YCbCr 420.

LogicVideoRxPort->content.video->timing.Vic = Desired_Vic;
LogicVideoRxPort->content.lvrx->Update   = 1;
If input timing is embeded sync, a separate *Desired_Vic* needs to be set externally. Then use *Update = 1* to update the value into the chip.

LogicVideoRxPort->content.video->AvailableVideoPackets = AV_BIT_AV_INFO_FRAME;
LogicVideoRxPort->content.video->Cd   = AV_CD_24;
Set the desired HDMI output color depth. Enable AVI Infoframe for HDMI output.

4.4 How to tune Parallel bus timing in software?

The registers' value could be modified into the default value in *Gsv2k11ParTxTable[]* and *Gsv2k11ParRxTable[]* of *gsv2k11_table.h*.

4.4.1 TTL Tx Register Control

TTL Tx related registers are shown below.

Register Name	*Register Description*	*Default Value*	*Tweak Value*
PAR.0xE7[4:3]	*TTL Drive Strength* *00 = 2mA* *10 = 4mA* *01 = 8mA* *11 = 12mA*	*11 = 12mA, maximum current*	*10 = 4mA*
PAR.0xE7[6]	*TTL Drive Slew Rate* *0 = Slow Slew* *1 = Fast Slew*	*1 = Fast Slew*	*0 = Slow Slew*
PAR.0xE2[2]	*Output clock phase selection*	*0 = Clock/Data 0*	*1 = Clock/Data 90*

	0 = Clock/Data 0 degree 1 = Clock/Data 90 degree	degree	degree
PAR.0xC8[3:0]	Parallel bus clock phase control, 40ps/step 0000 = 0 delay (minimum delay) 1111 = maximum delay	0000 = no delay based on PAR.0xE2[2]	xxxx = extra delay based on PAR.0xE2[2]

4.4.2 TTL Rx Register Control

TTL Rx related registers are shown below.

Register Name	Register Description	Default Value	Tweak Value
PAR.0xE5[1]	Input clock phase selection 0 = Clock/Data 0 degree 1 = Clock/Data 90 degree	0 = Clock/Data 0 degree	1 = Clock/Data 90 degree
PAR.0xE1[3:0]	Input clock forward delay with 40p/unit PAR.0xE1[7:4]=0 when [3:0] != 0	0x0B = 440 ps forward delay	xxxx
PAR.0xE1[7:4]	Input clock backward delay with 40p/unit PAR.0xE1[3:0]=0 when [7:4] != 0	0 = no backward delay	xxxx

4.4.3 LVDS Tx Register Control

LVDS Tx related registers are shown below.

Register Name	Register Description	Default Value	Tweak Value
PAR.0xC6[6:4]	LVDS Termination Code, TxTerm 001 = 700 ohm, 010 = 330 ohm 011 = 190 ohm, 100 = 130 ohm 101 = 100 ohm, 110 = 90 ohm 111 = 75 ohm, 000 = term off	011 = 190 ohm	000 = TxTerm disabled
PAR.0xC6[3:0]	LVDS Drive Swing, 0.5mA per step, 1001 = default value	1011	xxxx
PAR.0xC4[0]	Output Pre-Emphasis powerdown 0 = Pre-Emphasis Enabled 1 = Pre-Emphasis Disabled	0 = PE enabled	1 = PE disabled
PAR.0xC3[7:4]	Output Pre-Emphasis slew rate 1000 = slow, 0001 = fast, 0000 = open	0001 = fast	xxxx = slow
PAR.0xC5[7:4]	LVDS slew rate control 1000 = slow, 0001 = fast, 0000 = open	0001 = fast	xxxx = slow
PAR.0xC5[3:0]	LVDS output common voltage setting, 1001 for 1.25V, 50mV/step	1001	xxxx
PAR.0xE2[2]	Output clock phase selection 0 = Clock/Data 0 degree 1 = Clock/Data 90 degree	0 = Clock/Data 0 degree	1 = Clock/Data 90 degree
PAR.0xC8[3:0]	Parallel bus clock phase control, 40ps/step	0000 = no delay based on	xxxx = extra delay based on

	0000 = 0 delay (minimum delay) *1111 = maximum delay*	*PAR.0xE2[2]*	*PAR.0xE2[2]*

With typical Tx term's 100ohm by LVDS specification setting, the LVDS output swing can be calculated below.

*LVDS Output Swing = LVDS Drive Swing * 0.5mA * (RxTerm || TxTerm)*

4.4.4 LVDS Rx Register Control

LVDS Rx related registers are shown below.

Register Name	*Register Description*	*Default Value*	*Tweak Value*
PAR.0xE5[1]	*Input clock phase selection* *0 = Clock/Data 0 degree* *1 = Clock/Data 90 degree*	*0 = Clock/Data 0 degree*	*1 = Clock/Data 90 degree*
PAR.0xE1[3:0]	*Input clock forward delay with 40p/unit* *PAR.0xE1[7:4]=0 when [3:0] != 0*	*0x0B = 440 ps forward delay*	*xxxx*
PAR.0xE1[7:4]	*Input clock backward delay with 40p/unit* *PAR.0xE1[3:0]=0 when [7:4] != 0*	*0 = no backward delay*	*xxxx*

4.4.5 Lane Tx/Rx Skew Control

With manual control bit enabled, 4-bit skew value can be separated set per group.
For Parallel bus Rx, input deskew is also using the same scheme and registers. But when data bits are in auto deskew mode, DE/VS/HS must be set to manual mode with fixed setting. Or else, Parallel bus input timing will not be stable due to shifting timing signal.

Register Name	*Register Description*	*Default Value*	*Tweak Value*
PAR.0xC9[3:0] *PAR.0xCA[7:0]*	*LVDS 0~11/ TTL 0~11,24~35 skew manual control*	*All 1 = Manual*	*All 0 =Auto*
PAR.0xD1[7]	*LVDS DE /TTL DE&23 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD2[7]	*LVDS HS /TTL HS&13 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD3[7]	*LVDS VS /TTL VS&12 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD4[7]	*TTL 14~17 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD5[7]	*TTL 18~21 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD6[7]	*TTL 36~39 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD7[7]	*TTL 40~43 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xD8[7]	*TTL 44~47 skew manual control*	*1 = Manual*	*0 =Auto*
PAR.0xCB[3:0]	*LVDS 0/TTL 0~1, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCB[7:4]	*LVDS 1/TTL 2~3, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCC[3:0]	*LVDS 2/TTL 4~5, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCC[7:4]	*LVDS 3/TTL 6~7, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCD[3:0]	*LVDS 4/TTL 8~9, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCD[7:4]	*LVDS 5/TTL 10~11, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCE[3:0]	*LVDS 6/TTL 24~25, 40ps/step skew*	*0001*	*xxxx*
PAR.0xCE[7:4]	*LVDS 7/TTL 26~27, 40ps/step skew*	*0001*	*xxxx*

PAR.0xCF[3:0]	LVDS 8/TTL 28~29, 40ps/step skew	0001	xxxx
PAR.0xCF[7:4]	LVDS 9/TTL 30~31, 40ps/step skew	0001	xxxx
PAR.0xD0[3:0]	LVDS 10/TTL 32~33, 40ps/step skew	0001	xxxx
PAR.0xD0[7:4]	LVDS 11/TTL 34~35, 40ps/step skew	0001	xxxx
PAR.0xD1[3:0]	LVDS DE/TTL DE,23, 40ps/step skew	0001	xxxx
PAR.0xD2[3:0]	LVDS HS/TTL HS,13, 40ps/step skew	0001	xxxx
PAR.0xD3[3:0]	LVDS VS/TTL VS,12, 40ps/step skew	0001	xxxx
PAR.0xD4[3:0]	TTL 14~17, 40ps/step skew	0001	xxxx
PAR.0xD5[3:0]	TTL 18~21, 40ps/step skew	0001	xxxx
PAR.0xD6[3:0]	TTL 36~39, 40ps/step skew	0001	xxxx
PAR.0xD7[3:0]	TTL 40~43, 40ps/step skew	0001	xxxx
PAR.0xD8[3:0]	TTL 44~47, 40ps/step skew	0001	xxxx

4.5 How to tweak pin connection order in detail?

4.5.1 Rx Pin order control registers

Rx related registers are shown below.

Register Name	Register Description	Default Value	Tweak Value
PAR.0x25[6:4]	Bit Endian Swap Bit[4] 0 = Channel 0 [11:0]MSB->LSB Bit[4] 1 = Channel 0 [11:0]LSB->MSB Bit[5] 0 = Channel 1 [11:0]MSB->LSB Bit[5] 1 = Channel 1 [11:0]LSB->MSB Bit[6] 0 = Channel 2 [11:0]MSB->LSB Bit[6] 1 = Channel 2 [11:0]LSB->MSB	000 = MSB->LSB	111 = LSB->MSB
PAR.0x25[3]	In dual pixel mode, swap the order of higher bit and lower bit 0 = Default 1 = Swap	0 = default	1 = swap
PAR.0x25[2:0]	Channel Order Swap 000 = CH2/CH1/CH0 001 = CH2/CH0/CH1 010 = CH1/CH2/CH0 011 = CH1/CH0/CH2 100 = CH0/CH2/CH1 101/110/111 = CH0/CH1/CH2	000 = default	xxx = different order
PAR.0x26[1]	0 = Disable Pin offset (default) 1 = Enable Pin offset In LVDS Mode, when PIN Offset enabled, LVDS6~11 will be used, LVDS 0~5 will not be used. In TTL Mode, when PIN Offset enabled, TTL 24~47 will be used, TTL 0~23 will not be used.	0 = default	1 = enable pin offset

4.5.2 Tx Pin order control registers

Tx related registers are shown below.

Register Name	*Register Description*	*Default Value*	*Tweak Value*
PAR.0x65[6:4]	*Bit Endian Swap* *Bit[4] 0 = Channel 0 [11:0]MSB->LSB* *Bit[4] 1 = Channel 0 [11:0]LSB->MSB* *Bit[5] 0 = Channel 1 [11:0]MSB->LSB* *Bit[5] 1 = Channel 1 [11:0]LSB->MSB* *Bit[6] 0 = Channel 2 [11:0]MSB->LSB* *Bit[6] 1 = Channel 2 [11:0]LSB->MSB*	*000 = MSB->LSB*	*111 = LSB->MSB*
PAR.0x65[3]	*In dual pixel mode, swap the order of higher bits and lower bits, often referred as even/odd pixel swap* *0 = Default* *1 = Swap*	*0 = default*	*1 = swap*
PAR.0x65[2:0]	*Channel Order Swap* *000 = CH2/CH1/CH0* *001 = CH2/CH0/CH1* *010 = CH1/CH2/CH0* *011 = CH1/CH0/CH2* *100 = CH0/CH2/CH1* *101/110/111 = CH0/CH1/CH2*	*000 = default*	*xxx = different order*
PAR.0x66[1]	*0 = Disable Pin offset (default)* *1 = Enable Pin offset* *In LVDS Mode, when PIN Offset enabled, LVDS6~11 will be used, LVDS 0~5 will not be used.* *In TTL Mode, when PIN Offset enabled, TTL 24~47 will be used, TTL 0~23 will not be used.*	*0 = default*	*1 = enable pin offset*
PAR.0x68[2:0]	*2 = Default Cb/Cr 444->422 order* *4 = Swapped Cr/Cb 444->422 order*	*2 = default Cb/Cr order*	*4 = swapped Cr/Cb order*

For example, if Y/C is required to swap order to C/Y, add following code into Gsv2k11InitTable[] of gsv2k11_tables.h file.

0x14,0x65,0x01, // Y/C swap of 4:2:2 output

4.6 How to guarantee TTL input timing stable?

There is an internal CRC check module for checking parallel bus input timing pixel data's CRC value. With fixed pattern as the parallel bus input, the CRC value should be a fixed value.

The detailed CRC check process is shown below. The registers are located inside VSP map (0x12 by default).

Register Name	*Register Description*	*Default Value*	*Tweak Value*
VSP.0xC1[1]	*Writeable, CRC check enable* *0 = Disable, reset error count* *1 = Enable, start error count*	*0 = Disable*	*1 = Enable*

VSP.0xD2[7:0] *VSP.0xD3[7:0]*	*Writable, 16-bit golden CRC value for frame check comparison*	*0 = default*	*The CRC value to compare*
VSP.0xD4[7:0] *VSP.0xD5[7:0]*	*Read ONLY, 16-bit CRC value of current parallel input timing frame*	*0 = default*	
VSP.0xD6[7:0] *VSP.0xD7[7:0]* *VSP.0xD8[7:0]*	*Read ONLY, Total Error frame count, comparing between frame CRC and golden CRC value*	*0 = default*	
VSP.0xBA[7:0] *VSP.0xBB[7:0]* *VSP.0xBC[7:0]*	*Read ONLY, Total checked frame count since CRC check enabled, reset to 0 when CRC check disabled*	*0 = default*	

To use CRC module, follow below steps:
Step 1, wait until ParallelRxPort->content..rx->IsInputStable.
Step 2, disable CRC check enable register(VSP.0xC1[1]), read 16-bit CRC value of current frame (VSP.0xD4~0xD5).
Step 3, fill the 16-bit value into golden CRC value registers (VSP.0xD2~0xD3)
Step 4, enable CRC check enable register
Step 5, Read total checked frame and error frame to get result.
Step 6, when timing unlock, return to Step 1.

4.7 Why does Apple TV 4K60 420 not work under Dual Pixel UDP mode?

For Dual Pixel UDP mode, the DE must match even valid DE pixels. If there are odd valid DE pixels, the last odd pixel will be missing from the Dual Pixel UDP DE valid signal. So in this means, using Single Pixel UDP mode is the best way to match this kind of timing. We noticed that Apple TV 4K60 420 will have this issue when using Dual Pixel UDP mode. So Dual Pixel UDP mode is only recommended for 4K60 444 timing. For all lower pixel frequency timing, Single Pixel UDP mode is preferred for better compatibility.

4.8 How to set Parallel bus input timing parameter of embedded sync timing?

When using embedded sync timing (no separate Hsync/Vsync), software designer would need to set timing information to match the parallel bus video stream.
1, If the parallel bus input timing is formal CEA timing (including all CEA interlaced timings), designer only needs to set its CEA Vic and clear rest detailed timing information.
For example, for 4K@60 RGB, set the following information.

```
ParallelInputPort.content.video->timing.Vic       = 0x61; // 0x61 = 4K60 Vic
ParallelInputPort.content.video->timing.HPolarity = 0;
ParallelInputPort.content.video->timing.HActive   = 0;
ParallelInputPort.content.video->timing.HTotal    = 0;
ParallelInputPort.content.video->timing.HBack     = 0;
ParallelInputPort.content.video->timing.HSync     = 0;
ParallelInputPort.content.video->timing.VPolarity = 0;
ParallelInputPort.content.video->timing.VActive   = 0;
ParallelInputPort.content.video->timing.VTotal    = 0;
ParallelInputPort.content.video->timing.VBack     = 0;
ParallelInputPort.content.video->timing.VSync     = 0;
```

2, If the parallel bus input timing is in YCbCr 420 format, or does not have a formal Vic (including all VESA timings), designer needs to set its detailed timing information. If internal processing (420-444/downscaler) is required, Vic must also be set correctly.
For example, for 4K@60 YCbCr420, set the following information.

```
ParallelInputPort.content.video->timing.Vic       = 0x61; // Vic must be set correctly if it exists
ParallelInputPort.content.video->timing.HPolarity = 1;
ParallelInputPort.content.video->timing.HActive   = 3840/2; // horizontal pixel is in half for 420
ParallelInputPort.content.video->timing.HTotal    = 4400/2; // horizontal pixel is in half for 420
ParallelInputPort.content.video->timing.HBack     = 296/2; // horizontal pixel is in half for 420
ParallelInputPort.content.video->timing.HSync     = 88/2; // horizontal pixel is in half for 420
ParallelInputPort.content.video->timing.VPolarity = 1;
ParallelInputPort.content.video->timing.VActive   = 2160;
ParallelInputPort.content.video->timing.VTotal    = 2250;
ParallelInputPort.content.video->timing.VBack     = 72;
ParallelInputPort.content.video->timing.VSync     = 10;
```

4.9 How to loose LVDS/TTL input clock lock detection threshold?

When LVDS/TTL input is fed to GSV2011, it is possible that input LVDS/TTL clock has a larger jitter range than GSV2011 Rx setting. In this means, allowing a larger range of jitter tolerance threshold is required.
Add following code into *Gsv2k11InitTable[]* of *gsv2k11_tables.h*. By default, the *value* is 0x05 as threshold. Enlarging the value would make the input clock more tolerable. For example, setting the *value* to *0x0A* would enlarge the threshold.

```
0x06,0x85,value, // default is 0x05
```

4.9 Why does LPCM work but compressed audio fail in GSV2011 cascading?

Current source code is for I2S audio codec inter-connection. When HDMI input is compressed audio, source code has intentionally muted the TMDS output. To enable compressed audio output, modify code as below.

In *AvUapiCheckLogicAudioTx()* of *gsv2k11.c* file, edit below.

```
/* Step 1. Setup New Status */
TempPort = (AvPort*)(port->content.RouteAudioFromPort);
if(TempPort != NULL)
{
    if(TempPort->type == HdmiRx)
    {
        if(TempPort->content.rx->Lock.AudioLock == 0)
            NewValue = 0xff;
        else if((TempPort->content.audio->AudCoding == AV_AUD_FORMAT_LINEAR_PCM) ||
                (TempPort->content.audio->AudCoding == 0x00))
            NewValue = 0;
        else
            NewValue = 0;//0xef;
    }
    else if(TempPort->type == HdmiTx)
        NewValue = 0xef;
}
```

4.10 Why does 480i60/576i50 fail in ParallelPort->HDMI connection?

480i@60Hz and 576i@50Hz needs pixel repetition for HDMI output. So an extra configuration for the following registers are needed for these timings.

```
HdmiTxPort->content.video->ClockMultiplyFactor = 1; // extra pixel repetition, default = 0
HdmiTxPort->content.video->PixelRepeatValue = 1; // extra pixel repetition, default = 0
```

For normal timings above 480i@60Hz and 576i@50Hz, use the following setting.

```
HdmiTxPort->content.video->ClockMultiplyFactor = 0; // extra pixel repetition, default = 0
HdmiTxPort->content.video->PixelRepeatValue = 0; // extra pixel repetition, default = 0
```

4.11 How to set deep color in ParallelPort->HDMI connection?

Set color depth with General Control Packet enabled in ParallelRxPort->content.video->AvailableVideoPackets.

For example:

When HDMI Tx Port is not streaming at configuration time, add the following code.

```
ParallelRxPort.content.video->timing.Vic  = 0x10; /* 1080p60 */
ParallelRxPort.content.video->AvailableVideoPackets = AV_BIT_GC_PACKET |
AV_BIT_AV_INFO_FRAME;    // enable color depth control
ParallelRxPort.content.video->Cd          = AV_CD_36; // 12-bit color depth
```

When HDMI Tx Port is already streaming at configuration time, add the following code.

```
ParallelRxPort.content.video->timing.Vic  = 0x10; /* 1080p60 */
ParallelRxPort.content.video->AvailableVideoPackets = AV_BIT_GC_PACKET |
AV_BIT_AV_INFO_FRAME;    // enable color depth control
```

```
ParallelRxPort.content.video->Cd         = AV_CD_30; // new 10-bit color depth
HdmiTxPort.content.tx->Hpd = AV_HPD_FORCE_LOW;
```

4.12 How to pull-up/down ParallelPort pins in bootup?

In system bootup, the parallel pins might be needed to pull-up/down to avoid impact on the far-end device's boot up sequence. GSV2011's LVDS/TTL pins are default to be tri-stated. If desired, an internal pull-up/down resistor can be implemented.

Register Name	*Register Description*	*Default Value*	*Tweak Value*
PAR.0xE7[2:1]	*LVDS/TTL Pin Drive State* *00 = Hi-Z state* *01 = Pull-down ~100k ohm* *10 = Pull-up ~100k ohm to VDD1833* *11 = Reserved*	*00 = Hi-Z state*	*10 = Pull-Up* *01 = Pull-Down*

4.13 How to mute ParallelTxPort output clock when HDMI input is not plugged?

ParallelTxPort->content.lvtx->Config is used to define the Parallel Port output stability. Designer can wait until (*HdmiRxPort->content.rx->IsInputStable == 1*) to enable the ParallelPort output configuration.

```
ParallelTxPort->content.lvtx->Config = xx; // xx is a non-zero configuration value
ParallelTxPort ->content.lvtx->Update = 1;
```

When (*HdmiRxPort->content.rx->IsInputStable == 0*), if *ParallelTxPort->content.lvtx->Config* is not 0, reset the configuration to completely shut down ParallelTxPort output clock.

```
ParallelTxPort ->content.lvtx->Config = 0; // reset the ParallelTxPort
ParallelTxPort ->content.lvtx->Update = 1;
```

4.14 How to change color depth in Parallel->HDMI?

In *AvUapiTxSetHdmiDeepColor()*, add extra setting for dedicated color depth.

```
value = desired_color_depth; /* 5 = 10-bit, 6 = 12-bit, 7 = 16-bit */
if(value != 4)
   Gsv2k11AvUapiTxEnableInfoFrames(port, AV_BIT_GC_PACKET, 1);
/* DVI protection */
if(port->content.tx->HdmiMode == 0)
   value = 4;
GSV2K11_TXDIG_set_TX_GC_CD(port, value);
```

```
    return ret;
}
```

4.15 How to extend PLL/timing compliance in Parallel->HDMI?

Add following code into Gsv2k11InitTable[] of gsv2k11_tables.h file will greatly help on the input clock SSC and timing drift tolerance.

```
0x02,0x83,0x70,
0x02,0xB3,0x70,
0x14,0x19,0xFF,
0x14,0x1B,0xFF,
```

4.16 How to lower chip temperature in full power application?

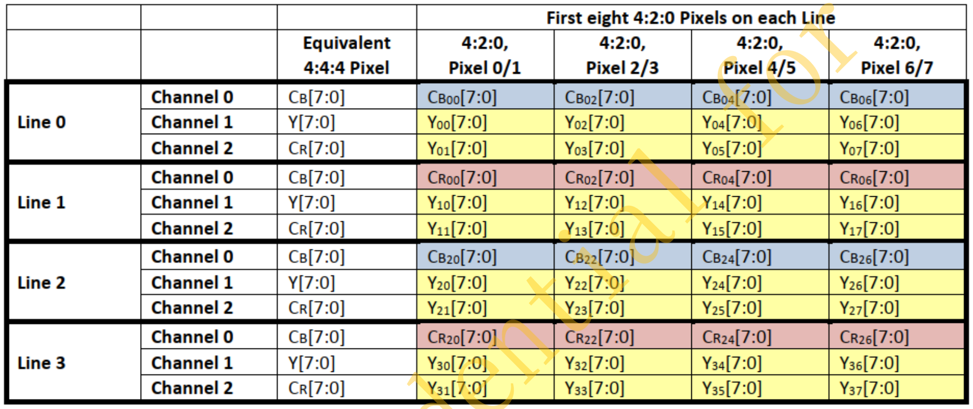
By default GSV2011 software, chip's LVDS/TTL drive strength is set to higher value for better compatibility. For chips' interconnection on the same board (especially with FPGA), a lower drive strength would reduce power and temperature drastically.
For example, if LVDS is used for GSV2011->FPGA application, do the following edit.
In Gsv2k11ParTxTable[] of gsv2k11_tables.h file, change the following setting.
For example, change "0xC6,0x3B," to "0xC6,0x04". This will disable the internal termination (which consumes power on LVDS) and lower the LVDS current to 2mA (which consumes less power).

5. Pin Mapping

5.1 What is the pin mapping of YCbCr 420 mode?

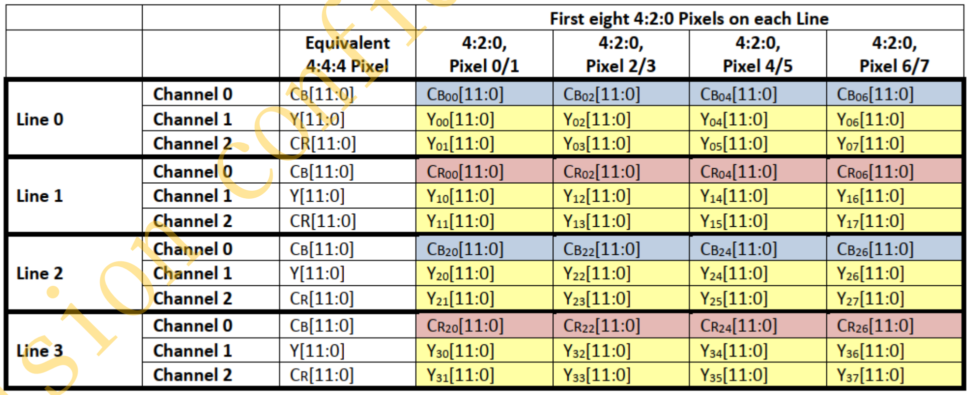
A perfect example of YCbCr 420 8-bit layout is shown in HDMI 2.0b specification Section 7.1. As shown in HDMI 2.0 specification, YCbCr 420 3 channels' TMDS layout is mirrored to YCbCr 444's 3 channels, by placing odd Pixel Y onto Channel 2. The example of 444<->420 channel mapping of HDMI 2.0b specification is shown below.

Table 7-2: Mapping Two 8-bit per component 4:2:0 Pixels to one 24-bit 4:4:4 Pixel prior to Deep Color Packing



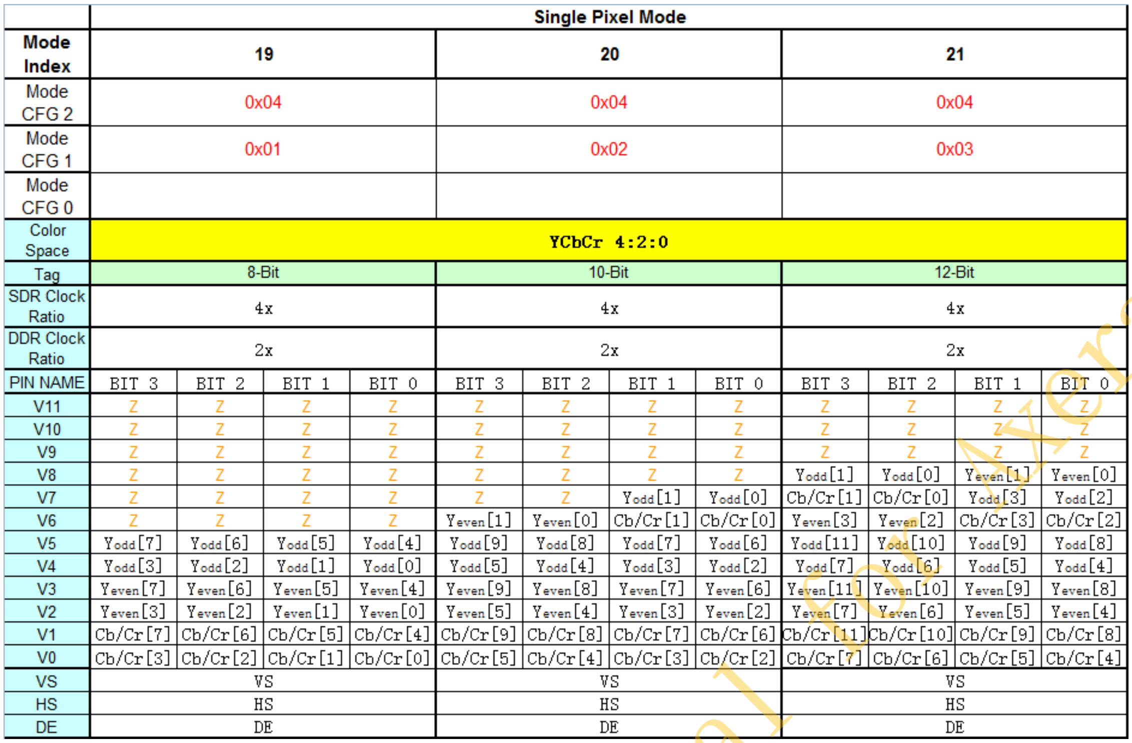
			First eight 4:2:0 Pixels on each Line			
		Equivalent 4:4:4 Pixel	4:2:0, Pixel 0/1	4:2:0, Pixel 2/3	4:2:0, Pixel 4/5	4:2:0, Pixel 6/7
Line 0	Channel 0	CB[7:0]	CB$_{00}$[7:0]	CB$_{02}$[7:0]	CB$_{04}$[7:0]	CB$_{06}$[7:0]
	Channel 1	Y[7:0]	Y$_{00}$[7:0]	Y$_{02}$[7:0]	Y$_{04}$[7:0]	Y$_{06}$[7:0]
	Channel 2	CR[7:0]	Y$_{01}$[7:0]	Y$_{03}$[7:0]	Y$_{05}$[7:0]	Y$_{07}$[7:0]
Line 1	Channel 0	CB[7:0]	CR$_{00}$[7:0]	CR$_{02}$[7:0]	CR$_{04}$[7:0]	CR$_{06}$[7:0]
	Channel 1	Y[7:0]	Y$_{10}$[7:0]	Y$_{12}$[7:0]	Y$_{14}$[7:0]	Y$_{16}$[7:0]
	Channel 2	CR[7:0]	Y$_{11}$[7:0]	Y$_{13}$[7:0]	Y$_{15}$[7:0]	Y$_{17}$[7:0]
Line 2	Channel 0	CB[7:0]	CB$_{20}$[7:0]	CB$_{22}$[7:0]	CB$_{24}$[7:0]	CB$_{26}$[7:0]
	Channel 1	Y[7:0]	Y$_{20}$[7:0]	Y$_{22}$[7:0]	Y$_{24}$[7:0]	Y$_{26}$[7:0]
	Channel 2	CR[7:0]	Y$_{21}$[7:0]	Y$_{23}$[7:0]	Y$_{25}$[7:0]	Y$_{27}$[7:0]
Line 3	Channel 0	CB[7:0]	CR$_{20}$[7:0]	CR$_{22}$[7:0]	CR$_{24}$[7:0]	CR$_{26}$[7:0]
	Channel 1	Y[7:0]	Y$_{30}$[7:0]	Y$_{32}$[7:0]	Y$_{34}$[7:0]	Y$_{36}$[7:0]
	Channel 2	CR[7:0]	Y$_{31}$[7:0]	Y$_{33}$[7:0]	Y$_{35}$[7:0]	Y$_{37}$[7:0]

Table 7-4: Mapping Two 12-bit per component 4:2:0 Pixels to one 36-bit 4:4:4 Pixel prior to Deep Color Packing



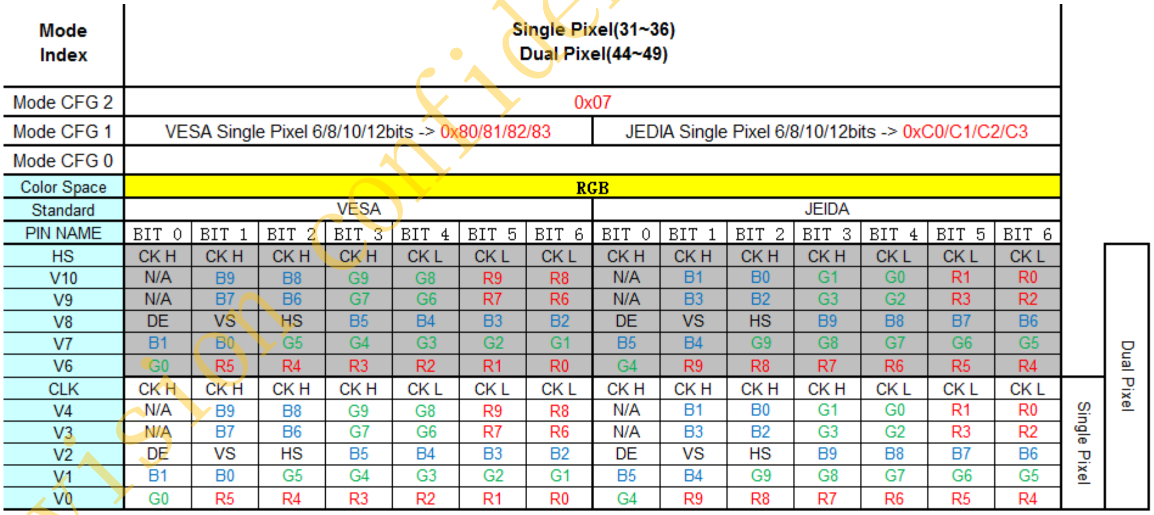
			First eight 4:2:0 Pixels on each Line			
		Equivalent 4:4:4 Pixel	4:2:0, Pixel 0/1	4:2:0, Pixel 2/3	4:2:0, Pixel 4/5	4:2:0, Pixel 6/7
Line 0	Channel 0	CB[11:0]	CB$_{00}$[11:0]	CB$_{02}$[11:0]	CB$_{04}$[11:0]	CB$_{06}$[11:0]
	Channel 1	Y[11:0]	Y$_{00}$[11:0]	Y$_{02}$[11:0]	Y$_{04}$[11:0]	Y$_{06}$[11:0]
	Channel 2	CR[11:0]	Y$_{01}$[11:0]	Y$_{03}$[11:0]	Y$_{05}$[11:0]	Y$_{07}$[11:0]
Line 1	Channel 0	CB[11:0]	CR$_{00}$[11:0]	CR$_{02}$[11:0]	CR$_{04}$[11:0]	CR$_{06}$[11:0]
	Channel 1	Y[11:0]	Y$_{10}$[11:0]	Y$_{12}$[11:0]	Y$_{14}$[11:0]	Y$_{16}$[11:0]
	Channel 2	CR[11:0]	Y$_{11}$[11:0]	Y$_{13}$[11:0]	Y$_{15}$[11:0]	Y$_{17}$[11:0]
Line 2	Channel 0	CB[11:0]	CB$_{20}$[11:0]	CB$_{22}$[11:0]	CB$_{24}$[11:0]	CB$_{26}$[11:0]
	Channel 1	Y[11:0]	Y$_{20}$[11:0]	Y$_{22}$[11:0]	Y$_{24}$[11:0]	Y$_{26}$[11:0]
	Channel 2	CR[11:0]	Y$_{21}$[11:0]	Y$_{23}$[11:0]	Y$_{25}$[11:0]	Y$_{27}$[11:0]
Line 3	Channel 0	CB[11:0]	CR$_{20}$[11:0]	CR$_{22}$[11:0]	CR$_{24}$[11:0]	CR$_{26}$[11:0]
	Channel 1	Y[11:0]	Y$_{30}$[11:0]	Y$_{32}$[11:0]	Y$_{34}$[11:0]	Y$_{36}$[11:0]
	Channel 2	CR[11:0]	Y$_{31}$[11:0]	Y$_{33}$[11:0]	Y$_{35}$[11:0]	Y$_{37}$[11:0]

From the perspective of TMDS channel mapping, the GSV2011 LVDS/TTL pin layout of YCbCr 444 is the same for YCbCr 420. In GSV2011 datasheet, YCbCr 444 pin mapping is given. Here an example of 4x LVDS YCbCr 420 pin mapping is shown to demonstrate the YCbCr 420 pin mapping for 8-bit/10-bit/12-bit.



Mode Index	Single Pixel Mode											
	19				20				21			
Mode CFG 2	0x04				0x04				0x04			
Mode CFG 1	0x01				0x02				0x03			
Mode CFG 0												
Color Space	YCbCr 4:2:0											
Tag	8-Bit				10-Bit				12-Bit			
SDR Clock Ratio	4x				4x				4x			
DDR Clock Ratio	2x				2x				2x			
PIN NAME	BIT 3	BIT 2	BIT 1	BIT 0	BIT 3	BIT 2	BIT 1	BIT 0	BIT 3	BIT 2	BIT 1	BIT 0
V11	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
V10	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
V9	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
V8	Z	Z	Z	Z	Z	Z	Z	Z	Yodd[1]	Yodd[0]	Yeven[1]	Yeven[0]
V7	Z	Z	Z	Z	Z	Z	Yodd[1]	Yodd[0]	Cb/Cr[1]	Cb/Cr[0]	Yodd[3]	Yodd[2]
V6	Z	Z	Z	Z	Yeven[1]	Yeven[0]	Cb/Cr[1]	Cb/Cr[0]	Yeven[3]	Yeven[2]	Cb/Cr[3]	Cb/Cr[2]
V5	Yodd[7]	Yodd[6]	Yodd[5]	Yodd[4]	Yodd[9]	Yodd[8]	Yodd[7]	Yodd[6]	Yodd[11]	Yodd[10]	Yodd[9]	Yodd[8]
V4	Yodd[3]	Yodd[2]	Yodd[1]	Yodd[0]	Yodd[5]	Yodd[4]	Yodd[3]	Yodd[2]	Yodd[7]	Yodd[6]	Yodd[5]	Yodd[4]
V3	Yeven[7]	Yeven[6]	Yeven[5]	Yeven[4]	Yeven[9]	Yeven[8]	Yeven[7]	Yeven[6]	Yeven[11]	Yeven[10]	Yeven[9]	Yeven[8]
V2	Yeven[3]	Yeven[2]	Yeven[1]	Yeven[0]	Yeven[5]	Yeven[4]	Yeven[3]	Yeven[2]	Yeven[7]	Yeven[6]	Yeven[5]	Yeven[4]
V1	Cb/Cr[7]	Cb/Cr[6]	Cb/Cr[5]	Cb/Cr[4]	Cb/Cr[9]	Cb/Cr[8]	Cb/Cr[7]	Cb/Cr[6]	Cb/Cr[11]	Cb/Cr[10]	Cb/Cr[9]	Cb/Cr[8]
V0	Cb/Cr[3]	Cb/Cr[2]	Cb/Cr[1]	Cb/Cr[0]	Cb/Cr[5]	Cb/Cr[4]	Cb/Cr[3]	Cb/Cr[2]	Cb/Cr[7]	Cb/Cr[6]	Cb/Cr[5]	Cb/Cr[4]
VS	VS				VS				VS			
HS	HS				HS				HS			
DE	DE				DE				DE			

5.2 What is the pin mapping of VESA/JEIDA LVDS mode?



Mode Index	Single Pixel(31~36) Dual Pixel(44~49)															
Mode CFG 2	0x07															
Mode CFG 1	VESA Single Pixel 6/8/10/12bits -> 0x80/81/82/83							JEDIA Single Pixel 6/8/10/12bits -> 0xC0/C1/C2/C3								
Mode CFG 0																
Color Space	RGB															
Standard	VESA							JEIDA								
PIN NAME	BIT 0	BIT 1	BIT 2	BIT 3	BIT 4	BIT 5	BIT 6	BIT 0	BIT 1	BIT 2	BIT 3	BIT 4	BIT 5	BIT 6		
HS	CK H	CK H	CK H	CK H	CK L	CK L	CK L	CK H	CK H	CK H	CK H	CK L	CK L	CK L		Dual Pixel
V10	N/A	B9	B8	G9	G8	R9	R8	N/A	B1	B0	G1	G0	R1	R0		
V9	N/A	B7	B6	G7	G6	R7	R6	N/A	B3	B2	G3	G2	R3	R2		
V8	DE	VS	HS	B5	B4	B3	B2	DE	VS	HS	B9	B8	B7	B6		
V7	B1	B0	G5	G4	G3	G2	G1	B5	B4	G9	G8	G7	G6	G5		
V6	G0	R5	R4	R3	R2	R1	R0	G4	R9	R8	R7	R6	R5	R4		
CLK	CK H	CK H	CK H	CK H	CK L	CK L	CK L	CK H	CK H	CK H	CK H	CK L	CK L	CK L	Single Pixel	
V4	N/A	B9	B8	G9	G8	R9	R8	N/A	B1	B0	G1	G0	R1	R0		
V3	N/A	B7	B6	G7	G6	R7	R6	N/A	B3	B2	G3	G2	R3	R2		
V2	DE	VS	HS	B5	B4	B3	B2	DE	VS	HS	B9	B8	B7	B6		
V1	B1	B0	G5	G4	G3	G2	G1	B5	B4	G9	G8	G7	G6	G5		
V0	G0	R5	R4	R3	R2	R1	R0	G4	R9	R8	R7	R6	R5	R4		

In single pixel mode, CLK should be used as VESA/JEIDA input/output clock. In dual pixel mode, CLK/HS share the same output timing of VESA/JEIDA clock. By default, CLK High and Low should strictly follow the declared phase of VESA/JEIDA specification.

If different clock phase is required for the monitor's input timing, use following registers to modify the phase.

GSV2011 supports any CLK phase and duty cycle adjustment of the 7 phases of the clock.

Register Name	*Register Description*	*Default Value*	*Tweak Value*
PAR.0x80[7]	*Writeable, manual clock phase enable* *0 = Disable* *1 = Enable, enable manual clock*	*0 = Disable*	*1 = Enable*
PAR.0x81[6:0]	*Writable, 7-bit clock phase setting* *1111_000 as default phase*	*0 = default*	*Desired phase value*
PAR.0x69[1]	*Writeable, HS mirrors CLK enable* *0 = Disable* *1 = Enable, HS mirrors CLK*	*0 = default*	*1 = HS/CLK shares the same output phase*

For example, to push a 1-bit phase delay, the PAR.0x81 is supposed to set 0x3C (011_1100). To push a 2-bit phase delay, the PAR.0x81 is supposed to set 0x1E (001_1110). To push a 6-bit phase delay, PAR.0x81 is supposed to set 0x79 (111_1001).

Refer to Section 4.1.1 for example configuration.

5.3 Can GSV2011 TTL do x2~x4 pin mapping like LVDS mode?

GSV2011 TTL can do pin overlapping with x2/x3/x4 like LVDS pin mapping. The limitation is that TTL maximum single pin frequency cannot exceed 300Mbps.
Here is the step for TTL x2~x4 pin mapping software control method.
Step 1, Find corresponding LVDS pin mapping 3-byte configuration. And replace it with TTL by setting 3rd_byte[7] to 0.
For example, *0x42, 0x11, 0x00,   // 11: TTL x2 12-bit, SDR mode, separate sync*
Step 2, Modify the setting for TTL x2~x4 in Gsv2k11ParTxTable[] of gsv2k11_tables.h file.
Replace *“0x66,0x00,”* by *“0x66,0x04,”*.

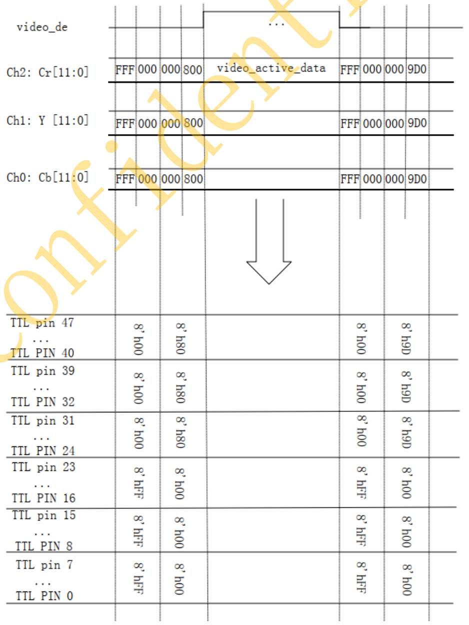
6. Embedded Sync Timing

6.1 Dual pixel SAV/EAV mapping

When using SAV/EAV to embed in dual pixel mode, the SAV/EAV layout has 2 different formats: YCbCr 422 and non-YCbCr 422 mode.

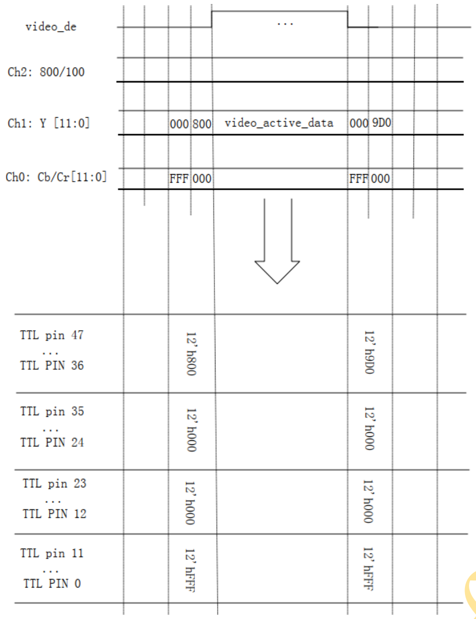
6.1.1 Non-YCbCr 422 mode

Non-YCbCr 422 mode means data is in YCbCr 444/ YCbCr 420/ RGB color space.
In dual pixel mode, non-YCbCr 422 parallel bus only supports this 'same SAV/EAV layout on 3 channels' mode. Using dual pixel mode, each channel only can support 8-bit data width, and SAV/EAV code is in on the higher 8-bit of the 12-bit channel data.
An example of TTL dual pixel non-YCbCr 422 parallel bus layout is shown below. If the parallel bus is using LVDS, the synchronization code will be packed according to data/clock ratio in the same format.



6.1.2 YCbCr 422 mode

In YCbCr 422 mode, the SAV/EAV follows the Y-C layout style.
An example of TTL dual pixel YCbCr 422 parallel bus layout is shown below. If the parallel bus is using LVDS, the synchronization code will be packed according to data/clock ratio in the same format.



6.2 SAV/EAV code value

SAV/EAV code could vary for blanking lines and active lines. SAV/EAV code also varies for interlaced timing as below table.

Video format	field	period	Sav/eav	1st word	2nd word	3rd word	4th word	8bits	10bits	12bits
Progressive Video Format	Same	V blanking line	Start sync code(SAV)	FFFh	000h	000h	AB0h	ABh	2ACh	AB0h
			End sync code(EAV)				B60h	B6h	2D8h	B60h
		Active line	Start sync code(SAV)				800h	80h	200h	800h
			End sync code(EAV)				9D0h	9Dh	274h	9D0h
Interlaced Video Format	Field=0	V blanking line	Start sync code(SAV)	FFFh	000h	000h	AB0h	ABh	2ACh	AB0h
			End sync code(EAV)				B60h	B6h	2D8h	B60h
		Active line	Start sync code(SAV)				800h	80h	200h	800h
			End sync code(EAV)				9D0h	9Dh	274h	9D0h
	Field=1	V blanking line	Start sync code(SAV)	FFFh	000h	000h	EC0h	ECh	3B0h	EC0h
			End sync code(EAV)				F10h	F1h	3C4h	F10h
		Active line	Start sync code(SAV)				C70h	C7h	31Ch	C70h
			End sync code(EAV)				DA0h	DAh	368h	DA0h

To avoid the conflict of active data and SAV/EAV, 000h and FFFh in the active data could be automatically replaced. A register *i2c_tx_itu_eliminate_ff_mode* is used to control the conflict solution.

i2c_tx_itu_eliminate_ff_mode (PAR.0x71[3])	When active data = 000h	When active data = FFFh
= 1	010h	FEFh
=0 (default)	010h	{FEh，orginal_data[3:0]}

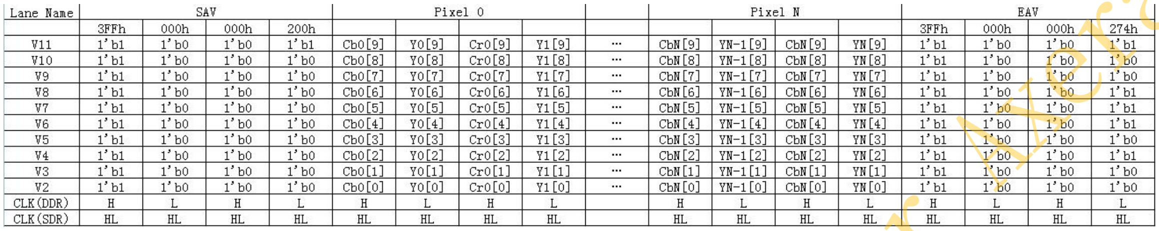
An extra configuration of default setting for SAV/EAV duplication is needed.
If YC is overlapped on the same pins with x2 mode, a single bit SAV/EAV for each value is needed use below setting.

0x14,0x30,0x08, // Rx Av Split Code = 1 to place on 2 channels
0x14,0x71,0x84, // Tx Av Split Code = 1 to place on 2 channels

If splitted Y/C bits are implemented on interface, SAV/EAV for each value is needed use below setting.

0x14,0x30,0x00, // Rx Av Split Code = 0 to place on 1 channel
0x14,0x71,0x80, // Tx Av Split Code = 0 to place on 1 channel

An example of YCbCr 4:2:2 10-bit SAV/EAV code and pixel data relationship is given below.



Lane Name	SAV				Pixel 0					Pixel N				EAV			
	3FFh	000h	000h	200h										3FFh	000h	000h	274h
V11	1’b1	1’b0	1’b0	1’b1	Cb0[9]	Y0[9]	Cr0[9]	Y1[9]	…	CbN[9]	YN-1[9]	CbN[9]	YN[9]	1’b1	1’b0	1’b0	1’b1
V10	1’b1	1’b0	1’b0	1’b0	Cb0[8]	Y0[8]	Cr0[8]	Y1[8]	…	CbN[8]	YN-1[8]	CbN[8]	YN[8]	1’b1	1’b0	1’b0	1’b0
V9	1’b1	1’b0	1’b0	1’b0	Cb0[7]	Y0[7]	Cr0[7]	Y1[7]	…	CbN[7]	YN-1[7]	CbN[7]	YN[7]	1’b1	1’b0	1’b0	1’b0
V8	1’b1	1’b0	1’b0	1’b0	Cb0[6]	Y0[6]	Cr0[6]	Y1[6]	…	CbN[6]	YN-1[6]	CbN[6]	YN[6]	1’b1	1’b0	1’b0	1’b1
V7	1’b1	1’b0	1’b0	1’b0	Cb0[5]	Y0[5]	Cr0[5]	Y1[5]	…	CbN[5]	YN-1[5]	CbN[5]	YN[5]	1’b1	1’b0	1’b0	1’b1
V6	1’b1	1’b0	1’b0	1’b0	Cb0[4]	Y0[4]	Cr0[4]	Y1[4]	…	CbN[4]	YN-1[4]	CbN[4]	YN[4]	1’b1	1’b0	1’b0	1’b1
V5	1’b1	1’b0	1’b0	1’b0	Cb0[3]	Y0[3]	Cr0[3]	Y1[3]	…	CbN[3]	YN-1[3]	CbN[3]	YN[3]	1’b1	1’b0	1’b0	1’b0
V4	1’b1	1’b0	1’b0	1’b0	Cb0[2]	Y0[2]	Cr0[2]	Y1[2]	…	CbN[2]	YN-1[2]	CbN[2]	YN[2]	1’b1	1’b0	1’b0	1’b1
V3	1’b1	1’b0	1’b0	1’b0	Cb0[1]	Y0[1]	Cr0[1]	Y1[1]	…	CbN[1]	YN-1[1]	CbN[1]	YN[1]	1’b1	1’b0	1’b0	1’b0
V2	1’b1	1’b0	1’b0	1’b0	Cb0[0]	Y0[0]	Cr0[0]	Y1[0]	…	CbN[0]	YN-1[0]	CbN[0]	YN[0]	1’b1	1’b0	1’b0	1’b0
CLK(DDR)	H	L	H	L	H	L	H	L		H	L	H	L	H	L	H	L
CLK(SDR)	HL	HL	HL	HL	HL	HL	HL	HL		HL	HL	HL	HL	HL	HL	HL	HL

7. GSV2011 Demo Board code

7.1 How to configure the code into source or sink version?

In apps/av_event_handler.c file, there is a variable called *LogicOutputSel*;
When setting *LogicOutputSel* to 1, the code is configured to source side (HDMI Rx->Parallel Out + HDMI Tx).
When setting *LogicOutputSel* to 0, the code is configured to sink side (Parallel In->HDMI Tx).

7.2 How to configure the parallel bus mode?

By default, GSV2011 demo board uses LVDS UDP dual pixel mode for demonstration. If other modes are required for performance comparison, it is recommended to modify mode configuration *CommonBusConfig* by detailed description from Section 4.1.