TMS320C64x DSP
Video Port/VCXO Interpolated Control (VIC) Port
Reference Guide

Literature Number: SPRU629A
September 2003

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Preface

Read This First

About This Manual

This document describes the video port and VCXO interpolated control (VIC) port in the digital signal processors (DSPs) of the TMS320C6000™ DSP family.

The following modes/features of the video port are currently not supported at this time. The video port is capable of using these modes/features but no technical assistance will be given at this time.

Modes:

- **8-, 10-, 16-, and 20-bit raw capture mode**
- **TSI capture mode**
- **Dual-sync display mode**

Features:

- **Noncontinuous capture or display**
- **Single frame capture or display**

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.

Related Documentation From Texas Instruments

The following documents describe the C6000™ devices and related support tools. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

TMS320C6000 CPU and Instruction Set Reference Guide (literature number SPRU189) describes the TMS320C6000™ CPU architecture, instruction set, pipeline, and interrupts for these digital signal processors.

TMS320C6000 DSP Peripherals Overview Reference Guide (literature number SPRU190) describes the peripherals available on the TMS320C6000™ DSPs.

TMS320C64x Technical Overview (SPRU395) gives an introduction to the TMS320C64x™ DSP and discusses the application areas that are enhanced by the TMS320C64x VelociTI™.

TMS320C6000 Programmer's Guide (literature number SPRU198) describes ways to optimize C and assembly code for the TMS320C6000™ DSPs and includes application program examples.

TMS320C6000 Code Composer Studio Tutorial (literature number SPRU301) introduces the Code Composer Studio™ integrated development environment and software tools.

Code Composer Studio Application Programming Interface Reference Guide (literature number SPRU321) describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.

TMS320C6x Peripheral Support Library Programmer's Reference (literature number SPRU273) describes the contents of the TMS320C6000™ peripheral support library of functions and macros. It lists functions and macros both by header file and alphabetically, provides a complete description of each, and gives code examples to show how they are used.

TMS320C6000 Chip Support Library API Reference Guide (literature number SPRU401) describes a set of application programming interfaces (APIs) used to configure and control the on-chip peripherals.

Trademarks

Code Composer Studio, C6000, C62x, C64x, C67x, TMS320C6000, TMS320C62x, TMS320C64x, TMS320C67x, and VelociTI are trademarks of Texas Instruments.

Contents

Figures

Tables

Chapter 1

Overview

This chapter provides an overview of the video port peripheral in the digital signal processors (DSPs) of the TMS320C6000™ DSP family. Included are an overview of the video port functions, FIFO configurations, and signal mapping.

Topic **Page**

1.1 Video Port

The following modes/features of the video port are currently not supported at this time. The video port is capable of using these modes/features but no technical assistance will be given at this time.

Modes:

- **8-, 10-, 16-, and 20-bit raw capture mode**
- **TSI capture mode**
- **Dual-sync display mode**

Features:

- **Noncontinuous capture or display**
- **Single frame capture or display**

The video port peripheral can operate as a video capture port, video display port, or transport stream interface (TSI) capture port. It provides the following functions:

- Video capture mode:
  - Capture rate up to 80 MHz.
  - Two channels of 8/10-bit digital video input from a digital camera or analog camera (using a video decoder). Digital video input is in YCbCr 4:2:2 format with 8-bit or 10-bit resolution multiplexed in ITU-R BT.656 format.
  - One channel of Y/C 16/20-bit digital video input in YCbCr 4:2:2 format on separate Y and Cb/Cr inputs. Supports SMPTE 260M, SMPTE 274M, SMPTE 296M, ITU-BT.1120, etc., as well as older CCIR601 interfaces.
  - YCbCr 4:2:2 to YCbCr 4:2:0 horizontal conversion and ½ scaling in 8-bit 4:2:2 modes.
  - Direct interface for two channels of up to 10-bit or one channel of up to 20-bit raw video from A/D converters.

- Video display mode:
  - Display rate up to 110 MHz.
  - One channel of continuous digital video output. Digital video output is YCbCr 4:2:2 co-sited pixel data with 8/10-bit resolution multiplexed in ITU-R BT.656 format.
  - One channel of Y/C 16/20-bit digital video output in YCbCr 4:2:2 format on separate Y and Cb/Cr outputs. (Supports SMPTE 260M, SMPTE 274M, SMPTE 296M, ITU-BT.1120, etc.)
  - YCbCr 4:2:0 to YCbCr 4:2:2 horizontal conversion and 2× scaling of output in 8-bit 4:2:2 modes.
  - Programmable clipping of BT.656 and Y/C mode output values.
  - One channel of raw data output up to 20-bits for interface to RAM-DACs. Two channel synchronized raw data output.
  - Synchronizes to external video controller or another video display port.
  - Using the external clock, the frame timing generator provides programmable image timing including horizontal and vertical blanking, start of active video (SAV) and end of active video (EAV) code insertion, and horizontal and frame timing pulses.
  - Generates horizontal and vertical synchronization and blanking signals and a frame synchronization signal.
- TSI capture mode: Transport stream interface (TSI) from a front-end device such as demodulator or forward error correction device in 8-bit parallel format at up to 30 Mbytes/sec.
- The port generates up to three events per channel and one interrupt to the DSP.

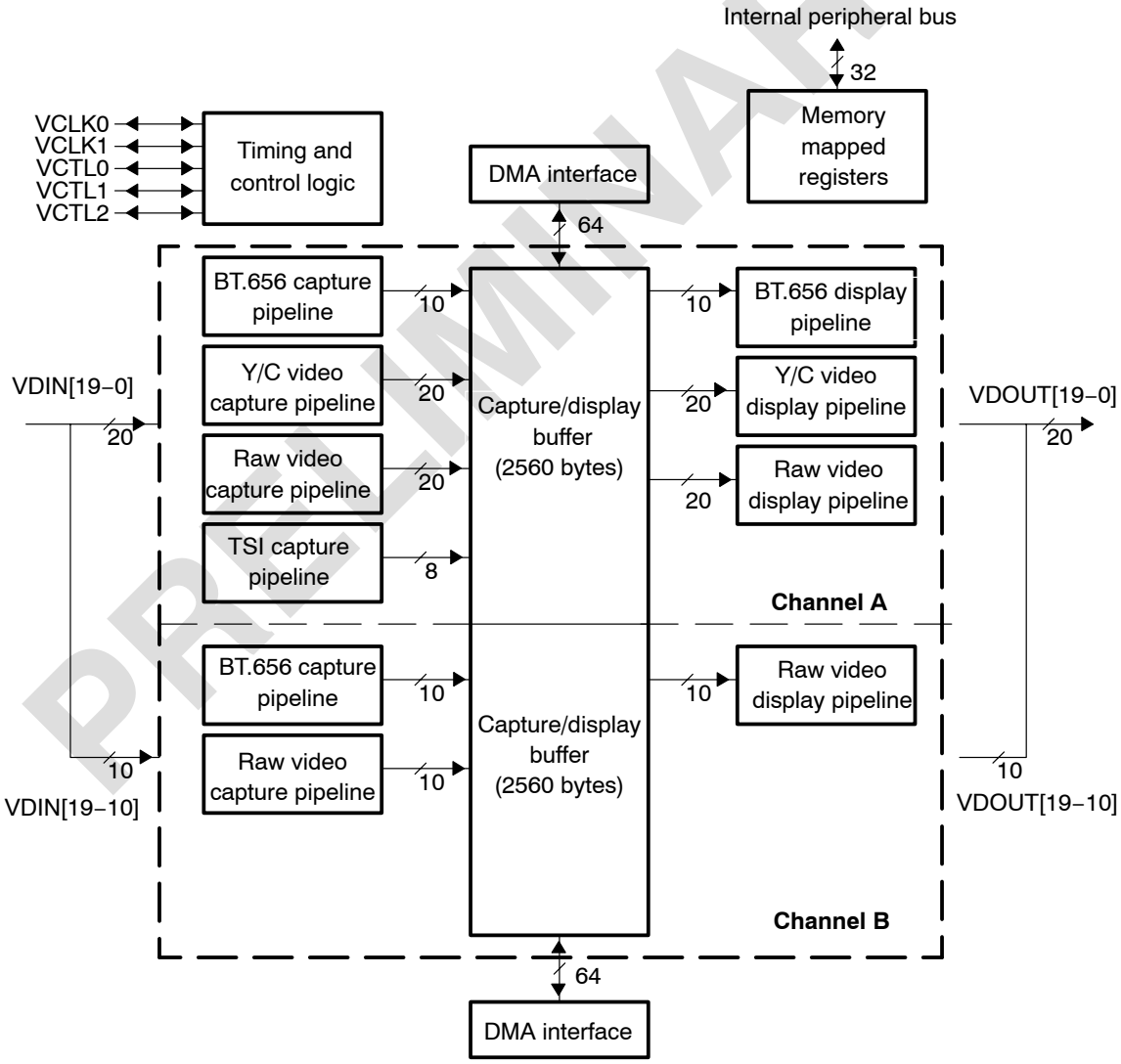
A high-level block diagram of the video port is shown in Figure 1–1. The port consists of two channels: A and B. A 5120-byte capture/display buffer is splittable between the two channels. The entire port (both channels) is always configured for either video capture or display only. Separate data pipelines control the parsing and formatting of video capture or display data for each of the BT.656, Y/C, raw video, and TSI modes.

For video capture operation, the video port may operate as two 8/10-bit channels of BT.656 or raw video capture; or as a single channel of 8/10-bit BT.656, 8/10-bit raw video, 16/20-bit Y/C video, 16/20-bit raw video, or 8-bit TSI.

For video display operation, the video port may operate as a single channel of 8/10-bit BT.656, 8/10-bit raw video, 16/20 bit Y/C video, or 16/20-bit raw video. It may also operate in a two channel 8/10-bit raw mode in which the two channels are locked to the same timing. Channel B is not used during single channel operation.

This document describes the full feature set offered by a 20-bit video port implementation. Some devices may offer a subset of features such as video capture only or video display only. Also, some devices may limit the video port width to 8 or 10 bits. In this case, modes requiring wider video port widths such as 16-bit raw, 20-bit raw, and Y/C are not supported. See the device-specific datasheet for details and for I/O timing information.

Figure 1−1. Video Port Block Diagram



1.2 Video Port FIFO

The video port includes a FIFO to store data coming into or out from the video port. The video port operates in conjunction with DMA transfers to move data between the video port FIFO and external or on-chip memory. You can program threshold settings so DMA events are generated when the video port FIFO reaches a certain fullness (for capture) or goes below a certain fullness (for display). DMAs required to service the FIFO are set up independently by you and are key to correct operation of the video port. The FIFO size is relatively large to allow time for DMAs to service the transfer requests, since devices typically have many peripheral interfaces often including multiple video ports.

The following sections briefly describe the interaction with the DMA and different FIFO configurations used to support the various modes of the video port.

1.2.1 DMA Interface

Video port data transfers take place using DMAs. DMA requests are based on buffer thresholds. Since the video port does not directly source the transfer, it can not adjust the transfer size based on buffer empty/full status. This means the DMA transfer size is essentially fixed in the user-programmed DMA parameter table. The preferred transfer size is often one entire line of data, because this allows the most flexibility in terms of frame buffer line pitch (in RAM). Some modes of operation for the highest display rates may require more frequent DMA requests such as on a half or quarter line basis.

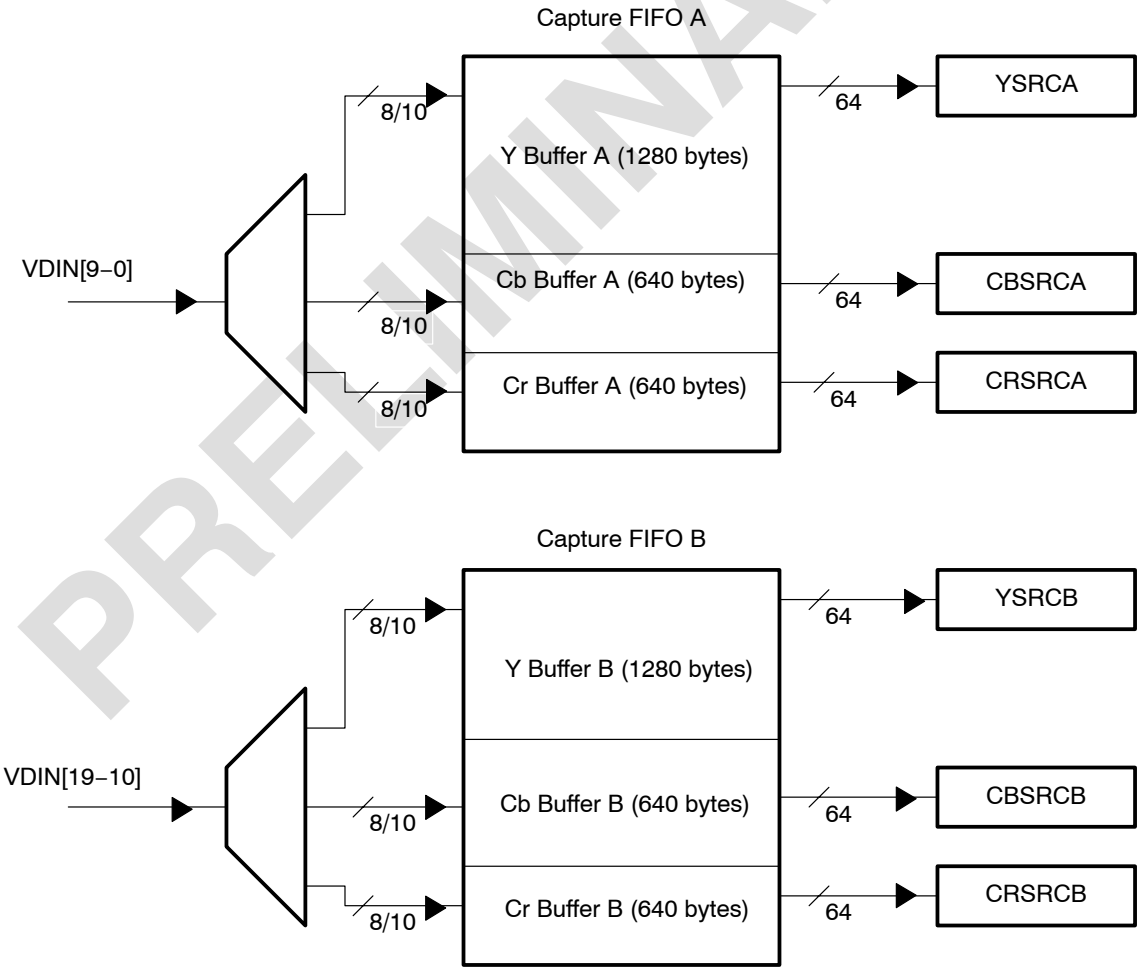
All requests are based on buffer thresholds. In video capture mode, DMA requests are made whenever the number of samples in the buffer reaches the threshold value. In order to ensure that all data from a capture field/frame gets emptied from the buffer, the transfer size must be equal to the threshold and the total amount of field/frame data must be a multiple of the transfer size.

For video display operation, DMA requests are made whenever there is at least the threshold number of doublewords free in the FIFO. This means that the transfer size must be equal to the threshold so that it fits into the available space. The field/frame size must still be a multiple of the transfer size or there are pixels left in the buffer at the end of the field (which appear at the start of the next field).

1.2.2 Video Capture FIFO Configurations

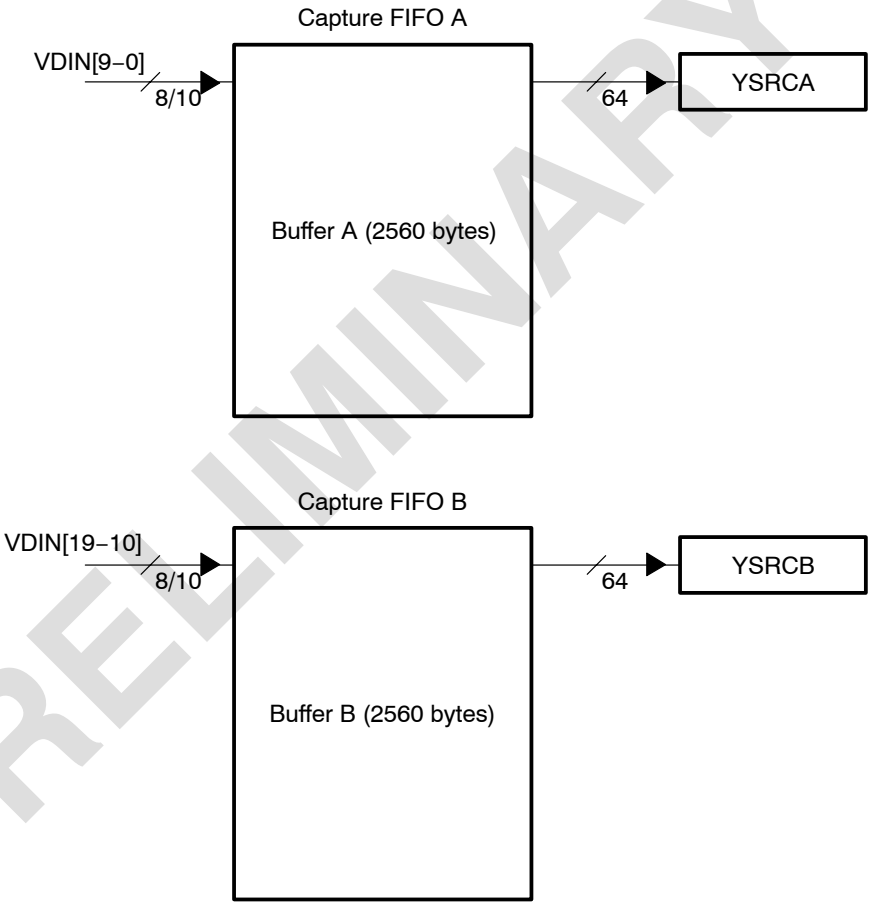
During video capture operation, the video port FIFO has one of four configurations depending on the capture mode. For BT.656 operation, the FIFO is split into channel A and B, as shown in Figure 1–2. Each FIFO is clocked independently with the channel A FIFO receiving data from the VDIN[9–0] half of the bus and the channel B FIFO receiving data from the VDIN[19–10] half of the bus. Each channel's FIFO is further split into Y, Cb, and Cr buffers with separate write pointers and read registers (YSRC*x*, CBSRC*x*, and CRSRC*x*).

Figure 1–2. BT.656 Video Capture FIFO Configuration



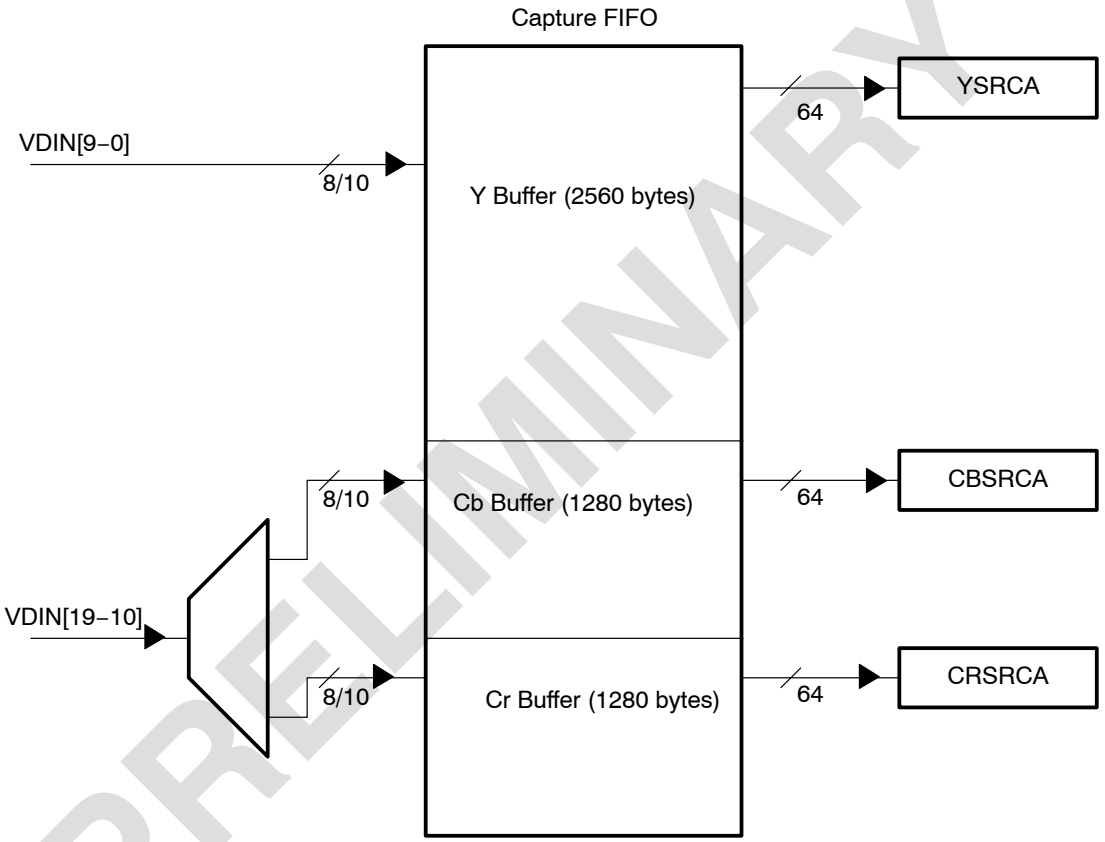
For 8/10-bit raw video, the FIFO is split into channel A and B, as shown in Figure 1–3. Each FIFO is clocked independently with the channel A FIFO receiving data from the VDIN[9–0] half of the bus and the channel B FIFO receiving data from the VDIN[19–10] half of the bus. Each channel's FIFO has a separate write pointer and read register (YSRC*x*). The FIFO configuration is identical for TSI capture, but channel B is disabled.

Figure 1–3. 8/10-Bit Raw Video Capture and TSI Video Capture FIFO Configuration



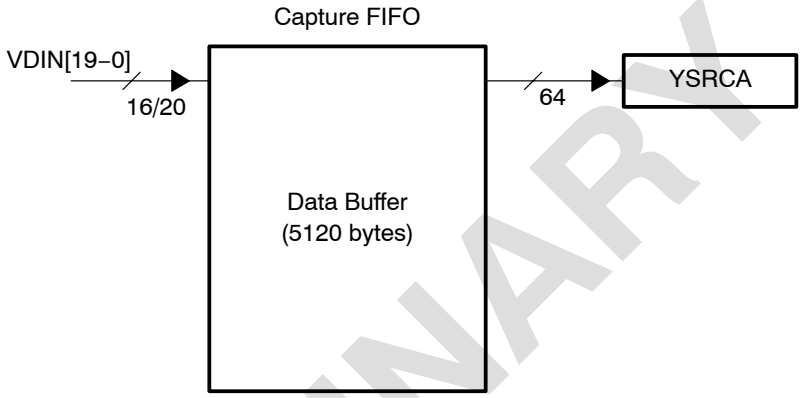
For Y/C video capture, the FIFO is configured as a single channel split into separate Y, Cb, and Cr buffers with separate write pointers and read registers (YSRCA, CBSRCA, and CRSRCA). Figure 1−4 shows how Y data is received on the VDIN[9−0] half of the bus and Cb/Cr data is received on the VDIN[19−10] half of the bus and demultiplexed into the Cb and Cr buffers.

Figure 1−4. Y/C Video Capture FIFO Configuration



For 16/20-bit raw video, the FIFO is configured as a single buffer, as shown in Figure 1–5. The FIFO receives 16/20-bit data from the VDIN[19–0] bus. The FIFO has a single write pointer and read register (YSRCA).

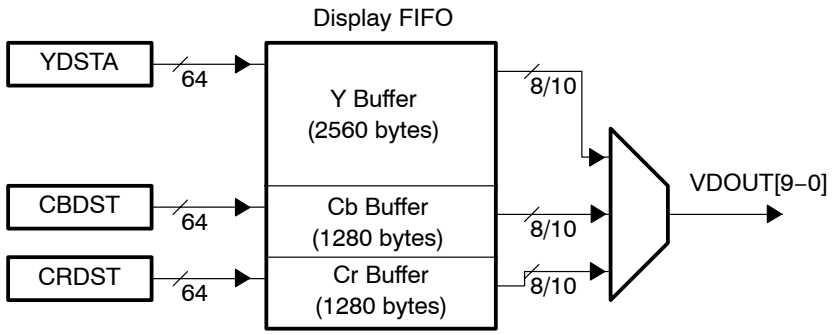
Figure 1–5. 16/20-Bit Raw Video Capture FIFO Configuration



1.2.3 Video Display FIFO Configurations

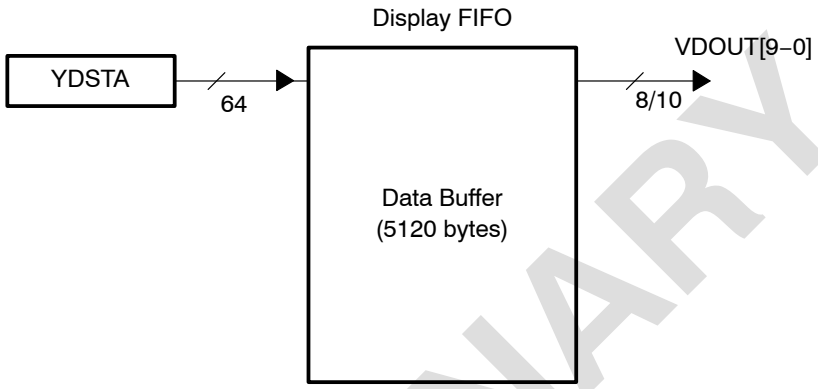
During video display operation, the video port FIFO has one of five configurations depending on the display mode. For BT.656 operation, a single output is provided on channel A, as shown in Figure 1–6, with data output on VDOUT[9–0]. The channel's FIFO is split into Y, Cb, and Cr buffers with separate read pointers and write registers (YDSTA, CBDST, and CRDST).

Figure 1–6. BT.656 Video Display FIFO Configuration



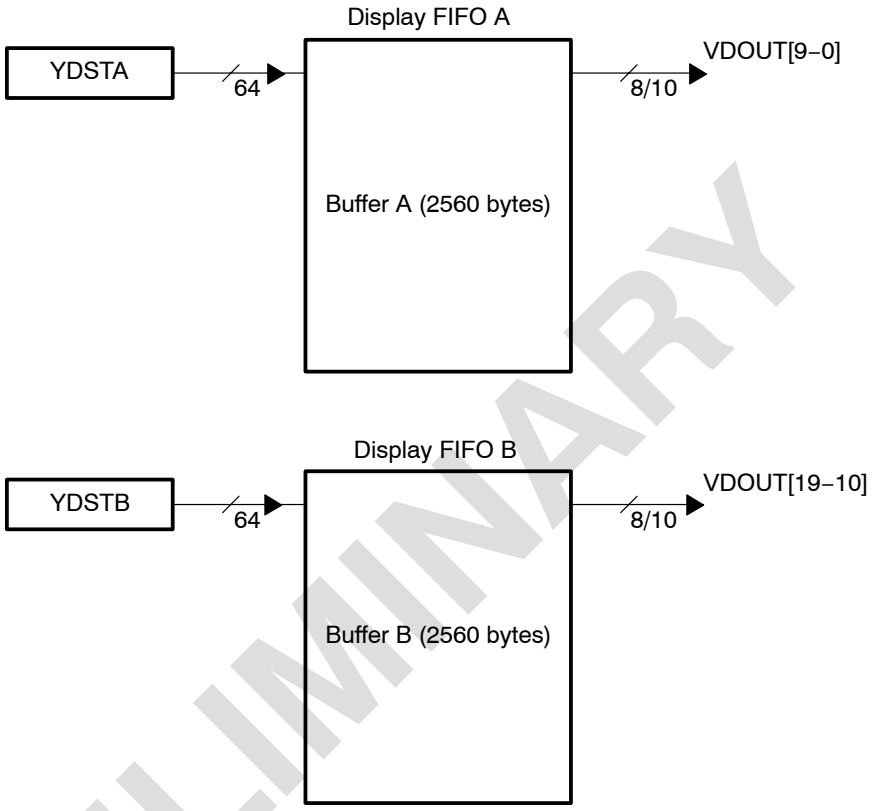
For 8/10-bit raw video, the FIFO is configured as a single buffer as shown in Figure 1−7. The FIFO outputs data on the VDOUT[9−0] half of the bus. The FIFO has a single read pointer and write register (YDSTA).

Figure 1−7. 8/10-Bit Raw Video Display FIFO Configuration



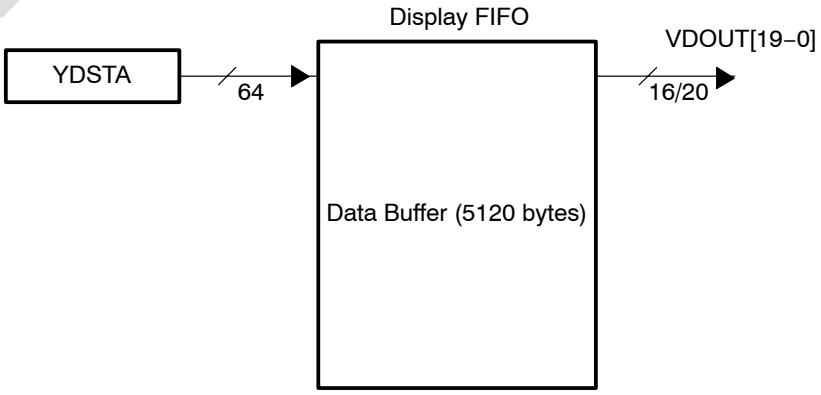
For locked raw video, the FIFO is split into channel A and B. The channels are locked together and use the same clock and control signals. Each channel uses a single buffer and write register (YDST*x*) as shown in Figure 1−8.

Figure 1–8. 8/10 Bit Locked Raw Video Display FIFO Configuration



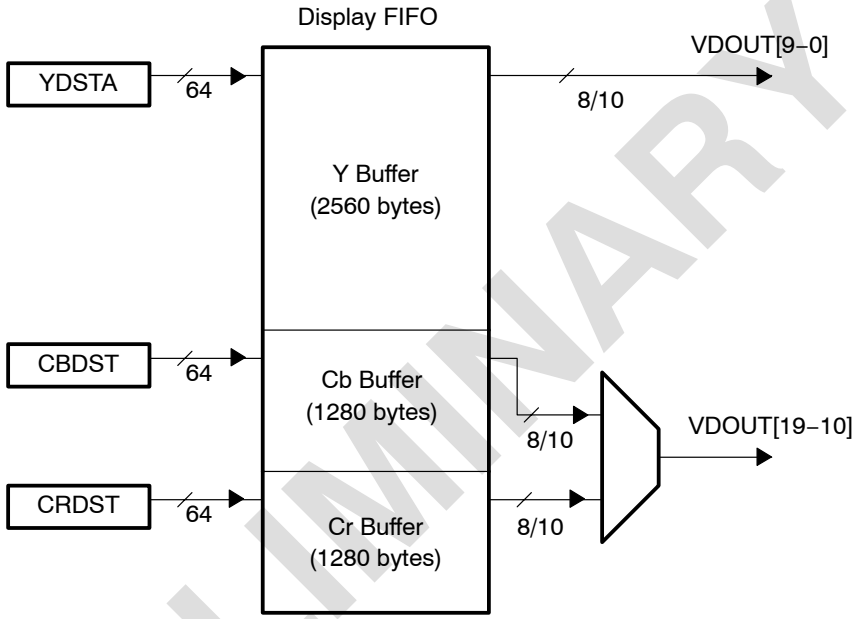
For 16/20-bit raw video, the FIFO is configured as a single buffer, as shown in Figure 1–9. The FIFO outputs data on VDOUT[19–0]. The FIFO has a single read pointer and write register (YDSTA).

Figure 1–9. 16/20-Bit Raw Video Display FIFO Configuration



For Y/C video display, the FIFO is configured as a single channel split into separate Y, Cb, and Cr buffers with separate read pointers and write registers (YDSTA, CBDST, and CRDST). Figure 1−10 shows how Y data is output on the VDOUT[9−0] half of the bus and Cb/Cr data is multiplexed and output on the VDOUT[19−10] half of the bus.

Figure 1−10. Y/C Video Display FIFO Configuration



1.3 Video Port Registers

The video port configuration register space is divided into several different sections with registers grouped by function including top-level video port control, video capture control, video display control, and GPIO.

The registers for controlling the video port are in section 2.4.

The registers for controlling the video capture mode of operation are shown in section 3.13. An additional space is dedicated for FIFO read pseudo-registers as shown in section 3.14. This space requires high-speed access and is not mapped to the register access bus.

The registers for controlling the video display mode of operation are shown in section 4.12. An additional space is dedicated for FIFO write pseudo-registers as shown in section 4.14. This space requires high-speed access and is not mapped to the register access bus.

The registers for controlling the general-purpose input/output (GPIO) are shown in section 5.1.

1.4 Video Port Pin Mapping

The video port requires 21 external signal pins for full functionality. Pin usage and direction changes depend on the selected operating mode. Pin functionality detail for video capture mode is listed in Table 1–1. Pin functionality detail for video display mode is listed in Table 1–2. All unused port signals (except VCLK0 and VCLK1) can be configured as general-purpose I/O (GPIO) pins.

Table 1–1. Video Capture Signal Mapping

		Usage					
		BT.656 Capture Mode			Raw Data Capture Mode		
Video Port Signal	I/O	Dual Channel	Single Channel	Y/C Capture Mode	8/10-Bit	16/20-Bit	TSI Capture Mode
VDATA[9–0]	I/O	VDIN[9–0] (In) Ch A	VDIN[9–0] (In) Ch A	VDIN[9–0] (In) (Y)	VDIN[9–0] (In) Ch A	VDIN[9–0] (In)	VDIN[7–0] (In)
VDATA[19–10]	I/O	VDIN[19–10] (In) Ch B	Not Used	VDIN[19–10] (In) (Cb/Cr)	VDIN[19–10] (In) Ch B	VDIN[19–10] (In)	Not Used
VLCK1	I	VCLKINA (In)	VCLKINA (In)	VCLKINA (In)	VCLKINA (In)	VCLKINA (In)	VCLKINA (In)
VCLK1	I/O	VCLKINB (In)	Not Used	Not Used	VCLKINB (In)	Not Used	Not Used
VCTL0	I/O	CAPENA (In)	CAPENA/ AVID/HSYNC (In)	CAPENA/ AVID/HSYNC (In)	CAPENA (In)	CAPENA (In)	CAPENA (In)
VCTL1	I/O	CAPENB (In)	VBLNK/ VSYNC (In)	VBLNK/ VSYNC (In)	CAPENB (In)	Not Used	PACSTRT (In)
VCTL2	I/O	Not Used	FID (In)	FID (In)	FID (In) Ch A	FID (In) Ch A	PACERR (In)

Legend: VCLKINA – Channel A capture clock; CAPENA – Channel A capture enable; VCLKINB – Channel B capture clock; CAPENB – Channel B capture enable; AVID – Active video; HSYNC – Horizontal synchronization; VBLNK – Vertical blanking; VSYNC – Vertical synchronization; FID – Field identification; PACSTRT – Packet start; PACERR – Packet error

Table 1–2. Video Display Signal Mapping

Video Port Signal	I/O	Usage				
				Raw Data Display Mode		
		BT.656 Display Mode	Y/C Display Mode	8/10-Bit	16/20-Bit	8/10-Bit Dual Sync
VDATA[9–0]	I/O	VDOUT[9–0] (Out)	VDOUT[9–0] (Out) (Y)	VDOUT[9–0] (Out)	VDOUT[9–0] (Out)	VDOUT[9–0] (Out) (Ch A)
VDATA[19–10]	I/O	Not Used	VDOUT[19–10] (Out) (Cb/Cr)	Not Used	VDOUT[19–10] (Out)	VDOUT[9–0] (Out) (Ch B)
VCLK0	I	VCLKIN (In)	VCLKIN (In)	VCLKIN (In)	VCLKIN (In)	VCLKIN (In)
VCLK1	I/O	VCLKOUT (Out)	VCLKOUT (Out)	VCLKOUT (Out)	VCLKOUT (Out)	VCLKOUT (Out)
VCTL0	I/O	HSYNC/HBLNK/ AVID/FLD (Out) or HSYNC (In)	HSYNC/HBLNK/ AVID/FLD (Out) or HSYNC (In)	HSYNC/HBLNK/ AVID/FLD (Out) or HSYNC (In)	HSYNC/HBLNK/ AVID/FLD (Out) or HSYNC (In)	HSYNC/HBLNK/ AVID/FLD (Out) or HSYNC (In)
VCTL1	I/O	VSYNC/VBLNK/ CSYNC/FLD (Out) or VSYNC (In)	VSYNC/VBLNK/ CSYNC/FLD (Out) or VSYNC (In)	VSYNC/VBLNK/ CSYNC/FLD (Out) or VSYNC (In)	VSYNC/VBLNK/ CSYNC/FLD (Out) or VSYNC (In)	VSYNC/VBLNK/ CSYNC/FLD (Out) or VSYNC (In)
VCTL2	I/O	CBLNK/FLD (Out) or FLD (In)	CBLNK/FLD (Out) or FLD (In)	CBLNK/FLD (Out) or FLD (In)	CBLNK/FLD (Out) or FLD (In)	CBLNK/FLD (Out) or FLD (In)

1.4.1 VDIN Bus Usage for Capture Modes

The alignment and usage of data on the VDIN bus depends on the capture mode as shown in Table 1−3.

Table 1−3. VDIN Data Bus Usage for Capture Modes

	Capture Mode								
	BT.656		Y/C		Raw Data				
Data Bus	10-Bit	8-Bit	10-Bit	8-Bit	8-Bit	10-Bit	16-Bit	20-Bit	TSI Mode
VDIN19	B	B	A (C)	A (C)	B	B	A	A	
VDIN18	B	B	A (C)	A (C)	B	B	A	A	
VDIN17	B	B	A (C)	A (C)	B	B	A	A	
VDIN16	B	B	A (C)	A (C)	B	B	A	A	
VDIN15	B	B	A (C)	A (C)	B	B	A	A	
VDIN14	B	B	A (C)	A (C)	B	B	A	A	
VDIN13	B	B	A (C)	A (C)	B	B	A	A	
VDIN12	B	B	A (C)	A (C)	B	B	A	A	
VDIN11	B		A (C)			B		A	
VDIN10	B		A (C)			B		A	
VDIN9	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN8	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN7	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN6	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN5	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN4	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN3	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN2	A	A	A (Y)	A (Y)	A	A	A	A	A
VDIN1	A		A (Y)			A		A	
VDIN0	A		A (Y)			A		A	

Legend: A – Channel A capture; A(C) Channel A chroma; A(Y) Channel A luma; B – Channel B capture

1.4.2 VDOUT Data Bus Usage for Display Modes

The alignment and usage of data on the VDOUT bus depends on the display mode as shown in Table 1–4.

Table 1–4. VDOUT Data Bus Usage for Display Modes

	Display Mode							
	BT.656		Y/C		Dual Sync Raw Data		Raw Data	
Data Bus	10-Bit	8-Bit	10-Bit	8-Bit	8-Bit	10-Bit	16-Bit	20-Bit
VDOUT19			A (C)	A (C)	(B)	(B)	A	A
VDOUT18			A (C)	A (C)	(B)	(B)	A	A
VDOUT17			A (C)	A (C)	(B)	(B)	A	A
VDOUT16			A (C)	A (C)	(B)	(B)	A	A
VDOUT15			A (C)	A (C)	(B)	(B)	A	A
VDOUT14			A (C)	A (C)	(B)	(B)	A	A
VDOUT13			A (C)	A (C)	(B)	(B)	A	A
VDOUT12			A (C)	A (C)	(B)	(B)	A	A
VDOUT11			A (C)			(B)		A
VDOUT10			A (C)			(B)		A
VDOUT9	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT8	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT7	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT6	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT5	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT4	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT3	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT2	A	A	A (Y)	A (Y)	A	A	A	A
VDOUT1	A		A (Y)			A		A
VDOUT0	A		A (Y)			A		A

Legend: A – Channel A display; A(C) Channel A chroma; A(Y) Channel A luma; B – Optional locked channel B display

Chapter 2

Video Port

This chapter discusses the basic operation of the video port. Included is a discussion of the sources and types of resets, interrupt operation, DMA operation, external clock inputs, video port throughput and latency, and the video port control registers.

Topic **Page**

2.1 Reset Operation

The video port has several sources and types of resets. The actions performed by these resets and the state of the port following the resets is described in the following sections.

2.1.1 Power-On Reset

Power-on reset is an asynchronous hardware reset caused by a chip-level reset operation. The reset is initiated by a power-on reset input to the video port. When the input is active, the port places all I/Os (VD[19−0], VCTL0, VCTL1, VCTL2, and VCLK1) in a high-impedance state.

2.1.2 Peripheral Bus Reset

Peripheral bus reset is a synchronous hardware reset caused by a chip-level reset operation. The reset is initiated by a peripheral bus reset input to the video port. This reset can be used internally (continuously asserted) to disable the video port for low-power operation. When the input is active, the port does the following:

- ❏ Places (keeps) all I/Os (VD[19−0], VCTL0, VCTL1, VCTL2, and VCLK1) in a high-impedance state.
- ❏ Flushes the FIFOs (resets pointers)
- ❏ Resets all port, capture, display, and GPIO registers to their default values. These may not complete until the appropriate module clock (VCLK0, VCLK1, STCLK) edges occur to synchronously release the logic from reset.
- ❏ Clears PEREN bit in PCR to 0.
- ❏ Sets VPHLT bit in VPCTL to 1.

While the peripheral remains disabled (PEREN = 0):

- ❏ VCLK0, VCLK1, and STCLK are gated off to save peripheral power.
- ❏ Peripheral bus accesses are acknowledged (RREADY/WREADY returned) to prevent DMA lock-up. (Any value returned on reads, data accepted or discarded on writes.)
- ❏ Peripheral bus MMR interface allows access to GPIO registers only (PID, PCR, PFUNC, PDIR, PIN, PDOUT, PDSET, PDCLR, PIEN, PIPOL, PISTAT, and PICLR).
- ❏ Port I/Os (VD[19−0], VCTL0, VCTL1, VCTL2, and VCLK1) remain in a high-impedance state unless enabled as GPIO by the PFUNC bits.

If software sets the PEREN bit in PCR but the VPHLT bit in VPCTL remains set:

- ❏ VCLK0, VCLK1, and STCLK are enabled to the port (allowing logic reset to complete).
- ❏ Peripheral bus accesses are acknowledged (RREADY/WREADY returned) to prevent DMA lock-up. (Any value returned on reads, data accepted or discarded on writes.)
- ❏ Peripheral bus MMR interface allows access to all registers.
- ❏ Port I/Os (VD[19−0], VCTL0, VCTL1, VCTL2, and VCLK1) remain in a high-impedance state unless enabled as GPIO by the PFUNC bits.
- ❏ VPCTL bits may be set (until the VPHLT bit is cleared).

2.1.3 Software Port Reset

A software port reset may be performed on the entire video port by setting the VPRST bit in VPCTL. This behaves identically to the peripheral bus reset except that it does not clear the PEREN bit in PCR. This reset:

- ❏ Performs an asynchronous reset on all port logic (channel logic may stay in reset until port input clock pulses occur).
- ❏ Self-clears the VPRTS bit to 0 but leaves the VPHLT bit set.

Once the port is configured and the VPHLT bit is cleared, the setting of other VPCTL bits (except VPRST) is disabled. The VCLK1 output may also be driven at this time, if display mode is selected. VCTL0−2 must remain in a high-impedance state unless enabled as GPIO, since internal/external sync is selected through VDCTL.

2.1.4 Capture Channel Reset

A software reset may be performed on a single capture channel by setting the RSTCH bit in VC*x*CTL. This reset requires that the channel VCLKIN be transitioning. On capture channel reset:

- ❏ No new DMA events are generated.
- ❏ Peripheral bus accesses are acknowledged (RREADY returned) to prevent DMA lock-up. (Any value returned on reads)
- ❏ Channel capture registers are set to their default values.
- ❏ Channel capture FIFO is flushed (pointers reset).
- ❏ The VCEN bit in VCxCTL is cleared to 0.
- ❏ The RSTCH bit self-clears to 0 after completion of the above.

Once the port is configured and the VCEN bit is set, the setting of other VC*x*CTL bits (except VCEN, RSTCH, and BLKCAP) is prohibited and the capture counters begin counting. When BLKCAP is cleared, data capture and event generation may begin.

2.1.5 Display Channel Reset

A software reset may be performed on the display channel by setting the RSTCH bit in VDCTL. This reset requires that the channel VCLKIN be transitioning. On display channel reset:

- ❑ No new DMA events are generated.
- ❑ Peripheral bus accesses are acknowledged (WREADY returned) to prevent DMA lock-up. (Write data may be written into the FIFO or discarded.)
- ❑ Channel display registers are set to their default values.
- ❑ Channel display FIFO is flushed (pointers reset).
- ❑ The VDEN bit in VDCTL is cleared to 0.
- ❑ The RSTCH bit self-clears to 0 after completion of the above.

Once the port is configured and the VDEN bit is set, the setting of other VDCTL bits (except VDEN, RSTCH, and BLKDIS) is prohibited and the display counters begin counting. Data outputs are driven (with default value, blanking, and control codes as appropriate and any control outputs are driven). When the BLKDIS bit is cleared, event generation may begin and FIFO data displayed.

2.2 Interrupt Operation

The video port generates an interrupt to the DSP core after any of the following events occur:

- ❏ Capture complete (CCMP*x*) bit is set.
- ❏ Capture overrun (COVR*x*) bit is set.
- ❏ Synchronization byte error (SERR*x*) bit is set.
- ❏ Vertical interrupt (VINT*xn*) bit is set.
- ❏ Short field detect (SFD*x*) bit is set.
- ❏ Long field detect (LFD*x*) bit is set.
- ❏ STC absolute time (STC) bit is set.
- ❏ STC tick counter expired (TICK) bit is set.
- ❏ Display complete (DCMP) bit is set.
- ❏ Display underrun (DUND) bit is set.
- ❏ Display complete not acknowledged (DCNA) bit is set.
- ❏ GPIO interrupt (GPIO) bit is set.

The interrupt signal is a pulse only and does not hold state. The interrupt pulse is generated only when the number of set flags in VPIS transitions from none to one or more. Another interrupt pulse is not generated by setting additional flag bits.

Interrupts can be masked via the video port interrupt enable register (VPIE) using individual interrupt enables and the VIE global enable bit. The interrupts are cleared in the video port interrupt status register (VPIS) using the individual status bits. Writing a 1 to the appropriate bit clears the interrupt. The clearing of an interrupt flag reenables the generation of another interrupt pulse, if other flags are still set. In other words, pulse generation is reenabled by writing a 1 to any set bit of VPIS.

Upon receiving an interrupt you should:

1) Read VPIS.

2) Perform the service routine for whatever bits are set.

3) Clear appropriate bits by writing a 1 to their VPIS locations.

4) Upon return from the ISR, if VPIS bits have been (or remain) set, then another interrupt will occur.

2.3 DMA Operation

The video port uses up to three DMA events per channel for a total of six possible events. Each DMA event uses a dedicated event output. The outputs are:

- VPYEVTA
- VPCbEVTA
- VPCrEVTA
- VPYEVTB
- VPCbEVTB
- VPCrEVTB

2.3.1 Capture DMA Event Generation

Capture DMA events are generated based on the state of the capture FIFO(s). If no DMA event is currently pending and the FIFO crosses the value specified by VCTHRLD*n*, a DMA event is generated. Once an event has been requested, another DMA event may not be generated until the servicing of the outstanding event has begun (as indicated by the first read of the FIFO by the DMA event service). If the capture FIFO level exceeds 2× the VCTHRLD*n* value before the requested DMA event completes, then another DMA event may be generated. Thus, up to one DMA event may be outstanding.

An outgoing data counter counts data read by the DMA. This counter is loaded with the VCTHRLD*n* value whenever a new DMA service begins. The counter then counts down for each double-word read from the FIFO by the DMA. The DMA is complete when the counter reaches zero.

For BT.656 and Y/C modes, there are three FIFOs, one for each of the Y, Cb, and Cr color components. Each FIFO generates its own DMA event; therefore, the DMA event state and FIFO thresholds for each FIFO are tracked independently. The Cb and Cr FIFOs use a threshold value of ½ (VCTHRLD*n* + VCTHRLD*n* mod 2).

Because the capture FIFOs may hold multiple thresholds worth of data, a problem arises at the boundaries between fields. Since Field 1 and Field 2 may have different threshold values, the amount of data in the FIFO required to generate the DMA event changes depending on the current capture field and the field of any outstanding DMA requests. Similarly, the threshold value loaded in the outgoing data counter needs to change depending on which field's DMA event is being serviced (not which field is currently being captured). To prevent confusion at the field boundaries, the VC*x*EVTCT register is programmed to indicate the number of events to generate for each field. An event counter tracks how many events have been generated and indicates which threshold value to use in event generation and in the outgoing data counter. After the last Field 1 event has been generated, the DMA logic looks for FIFO > THRSHLD1 + THRSHLD2 to pregenerate the first Field 2 event. Once the last Field 1 event completes, the logic looks for FIFO > 2 × THRSHLD2 (assuming a Field 2 event is outstanding).

Some initial devices may require THRSHLD1 and THRSHLD2 to be set to the same value. Check the latest device errata, if you want to use different thresholds for the two fields.

2.3.2 Display DMA Event Generation

Display DMA events are generated based on the amount of room available in the FIFO. The VDTHRLD*n* value indicates the level at which the FIFO has room to receive another DMA. If the FIFO has at least VDTHRLD*n* locations available, a DMA event is generated. Once a DMA event has been requested, another DMA event may not be generated until the servicing of the first DMA event has begun (as indicated by the first write to the FIFO by the DMA event service). If there is at least 2× the threshold space still available in the FIFO after the first DMA service is begun (and the display event counter has not expired) then another DMA event may be generated. Thus, up to one DMA request may be outstanding.

An incoming data counter is loaded with the VDTHRLD*n* (or VDTHRLD*n*/2 for Cb and Cr FIFOs) value at the beginning of each DMA event service and counts down the incoming DMA doublewords When the counter reaches 0, the DMA event is complete.

A DMA event counter is used to track the number of DMA events generated in each field as programmed in the VDDISPEVT register. The DISPEVT1 or DISPEVT2 value (depending on the current display field) is loaded at the start of each field. The event counter then decrements with each DMA event generation until it reaches 0, at which point no more DMA events are generated until the next field begins. Once the last line of data for a field has been requested, the DMA logic stops generating events until the field is complete in case the CPU needs to modify the DMA address pointers.

For BT.656 and Y/C modes, there are three FIFOs, one for each of the Y, Cb, and Cr color components. Each FIFO generates its own DMA event; therefore, the DMA event state and FIFO thresholds for each FIFO are tracked independently. (The Cb and Cr FIFOs use a threshold value of ½ VDTHRLD).

2.3.3 DMA Size and Threshold Restrictions

The video port FIFOs are 64-bits wide and always read or write 64 bits at a time. For this reason, DMA accesses must always be an even number of words in length. It is expected that in most cases the threshold size is set to the line length (rounded up to the next doubleword). This always works because different lines are not packed together within a doubleword and the Cb and Cr thresholds (½ VCTHRLD*x*/VDTHRLD) are always rounded up to the doubleword.

For example, in 8-bit BT.656 capture mode with a line length of 712 (Y), setting the threshold to the line length results in a VCTHRLD of 712 pixels × 1 bytes/pixel × doubleword/8 bytes = 89 doublewords. The Cb and Cr FIFOs contain half the data (44.5 doublewords) so their thresholds are set to 45 doublewords. Therefore, the Cb and Cr DMAs each transmit an extra 4 bytes at the end of each line.

If a multihorizontal line length threshold is desired (2 lines, for example) then the chosen line length must round up to an even number of doublewords so that it is evenly divisible by 2. If this is not the case, then the Cb and Cr FIFO transfers are corrupted. For the multiline case, consider the same 8-bit BT.656 capture mode with a line length of 712 (Y). If the threshold is set for 2 lines, this results in a VCTHRLD value of 2 × 89 = 178 doublewords. The actual Cb/Cr line length is 44.5 doublewords that requires a length of 45. To transfer 2 lines requires 2 × 45 = 90 doublewords. However, for this VCTHRLD, the DMA logic would calculate the Cb/Cr threshold size as 178/2 = 89 doublewords, which is 1 doubleword off. This can be corrected by increasing the line length to 720 pixels (and ignoring the extra captured pixels) or decreasing it to 704 pixels.

Similarly if a subhorizontal line length is desired (½ line, for example), then the line length and threshold must be chosen such that the threshold is divisible by 2. (This can also be stated as the line length must be an even multiple of #DMAs/line × 8). For the subline case, consider the 8-bit BT.656 capture mode with a line length of 624 (Y). If the threshold is set for ½ the line length, this results in VCTHRLD = (624/2)/8 = 39 doublewords. The DMA logic would calculate the Cb/Cr threshold as 39/2 = 20 doublewords. However, two such Cb/Cr DMA events would result in a transfer of 40 doublewords, which is larger than the actual Cb/Cr line length of (624/2)/8 = 39 doublewords. This can be corrected by changing the line size to 640 pixels or 608 pixels, or by changing the threshold to be 1/3 the line length (VCTHRLD = (624/3)/8 = 26 doublewords and the Cb/Cr threshold is 26/2 = 13 doublewords. 3 × 13 = 39 doublewords, which is exactly the Cb/Cr line length.)

2.3.4 DMA Interface Operation

When the video port is configured for capture (or TSI) mode, it only accepts read requests from the DMA interface. Write requests are false acknowledged (so the bus does not stall) and the data is discarded. When the video port is configured for display mode, it only accepts write requests. Read requests are false acknowledged (so the bus does not stall) and an arbitrary data value is returned.

When the video port is in reset, is not enabled (PEREN bit cleared), halted (VPHALT bit is set), or the active mode is not enabled (VCEN or VDEN bit is cleared), then the port will false acknowledge all DMA accesses to prevent bus lockup.

The video port DMA event generation logic is very tightly coupled to the DMA interface accesses. An incorrectly programmed DMA size causes the DMA and FIFO to become misaligned causing aberrations in the captured or displayed data and likely resulting in an eventual FIFO overflow or underflow. In the same manner, if another system DMA incorrectly addresses the video port during active capture or display, the video port has no way of determining that this is an errant DMA because all it monitors is a DMA access so it must perform the FIFO read or write. Such an errant DMA eventually causes the FIFO to be overread or overwritten.

2.4 Video Port Control Registers

The video port control registers are listed in Table 2−1. See the device-specific datasheet for the memory address of these registers.

After enabling the video port in the peripheral configuration register (PERCFG), there should be a delay of 64 CPU cycles before accessing the video port registers.

Table 2−1. Video Port Control Registers

Offset Address†	Acronym	Register Name	Section
C0h	VPCTL	Video Port Control Register	2.4.1
C4h	VPSTAT	Video Port Status Register	2.4.2
C8h	VPIE	Video Port Interrupt Enable Register	2.4.3
CCh	VPIS	Video Port Interrupt Status Register	2.4.4

† The absolute address of the registers is device/port specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

2.4.1 Video Port Control Register (VPCTL)

The video port control register (VPCTL) determines the basic operation of the video port. The VPCTL is shown in Figure 2−1 and described in Table 2−2.

Not all combinations of the port control bits are unique. The control bit encoding is shown in Table 2−3. Additional mode options are selected using the video capture channel A control register (VCACTL) and video display control register (VDCTL).

Figure 2−1. Video Port Control Register (VPCTL)

31							16
Reserved							
R-0							

15	14	13					8
VPRST	VPHLT	Reserved					
R/WS-0	R/WC-1	R-0					

7	6	5	4	3	2	1	0
VCLK1P	VCT2P	VCT1P	VCT0P	Reserved	TSI	DISP	DCHNL
R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; R/W = Read/Write; WC = Write a 1 to clear; WS = Write 1 to set, write of 0 has no effect; *-n* = value after reset

Table 2−2. Video Port Control Register (VPCTL) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−16	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15	VPRST			Video port software reset enable bit. VPRST is set by writing a 1. Writing 0 has no effect.
		NO	0	
		RESET	1	Flush all FIFOs and set all port registers to their initial values. VCLK0 and VCLK1 are configured as inputs and all VDATA and VCTL pins are placed in high impedance. Auto-cleared after reset is complete.

† For CSL implementation, use the notation VP_VPCTL_*field_symval*

Table 2–2. Video Port Control Register (VPCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
14	VPHLT			Video port halt bit. This bit is set upon hardware or software reset. The other VPCTL bits (except VPRST) can only be changed when VPHLT is 1. VPHLT is cleared by writing a 1. Writing 0 has no effect.
		NONE	0	
		CLEAR	1	
13–6	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
7	VCLK1P			VCLK1 pin polarity bit. Has no effect in capture mode.
		NONE	0	
		REVERSE	1	Inverts the VCLK1 output clock polarity in display mode.
6	VCT2P			VCTL2 pin polarity. Does not affect GPIO operation. If VCTL2 pin is used as a FLD input on the video capture side, then the VCTL2 polarity is not considered; the field inverse is controlled by the FINV bit in the video capture channel *x* control register (VC*x*CTL).
		NONE	0	
		ACTIVELOW	1	Indicates the VCTL2 control signal (input or output) is active low.
5	VCT1P			VCTL1 pin polarity bit. Does not affect GPIO operation.
		NONE	0	
		ACTIVELOW	1	Indicates the VCTL1 control signal (input or output) is active low.
4	VCT0P			VCTL0 pin polarity bit. Does not affect GPIO operation.
		NONE	0	
		ACTIVELOW	1	Indicates the VCTL0 control signal (input or output) is active low.
3	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation VP_VPCTL_*field_symval*

Table 2–2. Video Port Control Register (VPCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
2	TSI			TSI capture mode select bit.
		NONE	0	TSI capture mode is disabled.
		CAPTURE	1	TSI capture mode is enabled.
1	DISP			Display mode select bit. VDATA pins are configured for output. VCLK1 pin is configured as VCLKOUT output.
		CAPTURE	0	Capture mode is enabled.
		DISPLAY	1	Display mode is enabled.
0	DCHNL			Dual channel operation select bit. If the DCDIS bit in VPSTAT is set, this bit is forced to 0.
		SINGLE	0	Single-channel operation is enabled.
		DUAL	1	Dual-channel operation is enabled.

† For CSL implementation, use the notation VP_VPCTL_*field_symval*

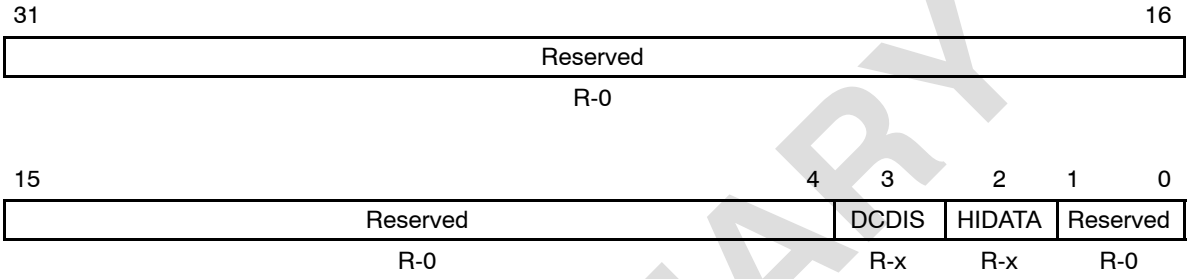
Table 2–3. Video Port Operating Mode Selection

VPCTL Bit			
TSI	**DISP**	**DCHNL**	**Operating Mode**
0	0	0	Single channel video capture. BT.656, Y/C or raw mode as selected in VCACTL. Video capture B channel not used.
0	0	1	Dual channel video capture. Either BT.656 or raw 8/10-bit as selected in VCACTL and VCBCTL. Option is available only if DCDIS is 0.
0	1	x	Single channel video display. BT.656, Y/C or raw mode as selected in VDCTL. Video display B channel is only used for dual channel sync raw mode.
1	x	x	Single channel TSI capture.

2.4.2 Video Port Status Register (VPSTAT)

The video port status register (VPSTAT) indicates the current condition of the video port. The VPSTAT is shown in Figure 2−2 and described in Table 2−4.

Figure 2−2. Video Port Status Register (VPSTAT)



Legend: R = Read only; -*n* = value after reset; -x = value is determined by chip-level configuration

Table 2−4. Video Port Status Register (VPSTAT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−4	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3	DCDIS			Dual-channel disable bit. The default value is determined by the chip-level configuration.
		ENABLE	0	Dual-channel operation is enabled.
		DISABLE	1	Port muxing selections prevent dual-channel operation.
2	HIDATA			High data bus half. HIDATA does not affect video port operation but is provided to inform you which VDATA pins may be controlled by the video port GPIO registers. HIDATA is never set unless DCDIS is also set. The default value is determined by the chip-level configuration.
		NONE	0	
		USE	1	Indicates that another peripheral is using VDATA[9−0] and the video port channel A (VDIN[9−0] or VDOUT[9−0]) is muxed onto VDATA[19−10].
1−0	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation VP_VPSTAT_*field_symval*

2.4.3 Video Port Interrupt Enable Register (VPIE)

The video port interrupt enable register (VPIE) enables sources of the video port interrupt to the DSP. The VPIE is shown in Figure 2–3 and described in Table 2–5.

Figure 2–3. Video Port Interrupt Enable Register (VPIE)

31							24
Reserved							
R-0							
23	22	21	20	19	18	17	16
LFDB	SFDB	VINTB2	VINTB1	SERRB	CCMPB	COVRB	GPIO
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8
Reserved	DCNA	DCMP	DUND	TICK	STC	Reserved	
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	
7	6	5	4	3	2	1	0
LFDA	SFDA	VINTA2	VINTA1	SERRA	CCMPA	COVRA	VIE
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 2–5. Video Port Interrupt Enable Register (VPIE) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–24	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
23	LFDB			Long field detected on channel B interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
22	SFDB			Short field detected on channel B interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
21	VINTB2			Channel B field 2 vertical interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.

† For CSL implementation, use the notation VP_VPIE_*field_symval*

Table 2–5. Video Port Interrupt Enable Register (VPIE) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
20	VINTB1			Channel B field 1 vertical interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
19	SERRB			Channel B synchronization error interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
18	CCMPB			Capture complete on channel B interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
17	COVRB			Capture overrun on channel B interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
16	GPIO			Video port general purpose I/O interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
15	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
14	DCNA			Display complete not acknowledged bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
13	DCMP			Display complete interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
12	DUND			Display underrun interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
11	TICK			System time clock tick interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.

† For CSL implementation, use the notation VP_VPIE_*field_symval*

Table 2–5. Video Port Interrupt Enable Register (VPIE) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
10	STC			System time clock interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
9–8	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
7	LFDA			Long field detected on channel A interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
6	SFDA			Short field detected on channel A interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
5	VINTA2			Channel A field 2 vertical interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
4	VINTA1			Channel A field 1 vertical interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
3	SERRA			Channel A synchronization error interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
2	CCMPA			Capture complete on channel A interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
1	COVRA			Capture overrun on channel A interrupt enable bit.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.
0	VIE			Video port global interrupt enable bit. Must be set for interrupt to be sent to DSP.
		DISABLE	0	Interrupt is disabled.
		ENABLE	1	Interrupt is enabled.

† For CSL implementation, use the notation VP_VPIE_*field_symval*

2.4.4 Video Port Interrupt Status Register (VPIS)

The video port interrupt status register (VPIS) displays the status of video port interrupts to the DSP. The interrupt is only sent to the DSP if the corresponding enable bit in VPIE is set. All VPIS bits are cleared by writing a 1, writing a 0 has no effect. The VPIS is shown in Figure 2−4 and described in Table 2−6.

Figure 2−4. Video Port Interrupt Status Register (VPIS)

31							24
Reserved							
R-0							
23	22	21	20	19	18	17	16
LFDB	SFDB	VINTB2	VINTB1	SERRB	CCMPB	COVRB	GPIO
R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0
15	14	13	12	11	10	9	8
Reserved	DCNA	DCMP	DUND	TICK	STC	Reserved	
R-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R-0	
7	6	5	4	3	2	1	0
LFDA	SFDA	VINTA2	VINTA1	SERRA	CCMPA	COVRA	Reserved
R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R-0

Legend: R = Read only; WC = Write 1 to clear, write of 0 has no effect; *-n* = value after reset

Table 2−6. Video Port Interrupt Status Register (VPIS) Field Descriptions

Bit	*field*	*symval*	Value	Description
31−24	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
23	LFDB			Long field detected on channel B interrupt detected bit. (A long field is only detected when the VRST bit in VCBCTL is cleared to 0; when VRST = 1, a long field is always detected.) BT.656 or Y/C capture mode − LFDB is set when long field detection is enabled and VCOUNT is not reset before VCOUNT = YSTOP + 1. Raw data mode, or TSI capture mode or display mode − Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.

† For CSL implementation, use the notation VP_VPIS_*field_symval*

Table 2–6. Video Port Interrupt Status Register (VPIS) Field Descriptions (Continued)

Bit	*field*	*symval*	Value	Description
22	SFDB			Short field detected on channel B interrupt detected bit.
				BT.656 or Y/C capture mode – SFDB is set when short field detection is enabled and VCOUNT is reset before VCOUNT = YSTOP.
				Raw data mode, or TSI capture mode or display mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
21	VINTB2			Channel B field 2 vertical interrupt detected bit.
				BT.656 or Y/C capture mode – VINTB2 is set when a vertical interrupt occurred in field 2.
				Raw data mode or TSI capture mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
20	VINTB1			Channel B field 1 vertical interrupt detected bit.
				BT.656 or Y/C capture mode – VINTB1 is set when a vertical interrupt occurred in field 1.
				Raw data mode or TSI capture mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
19	SERRB			Channel B synchronization error interrupt detected bit.
				BT.656 or Y/C capture mode – Synchronization parity error on channel B. An SERRB typically requires resetting the channel (RSTCH) or the port (VPRST).
				Raw data mode or TSI capture mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.

† For CSL implementation, use the notation VP_VPIS_*field_symval*

Table 2–6. Video Port Interrupt Status Register (VPIS) Field Descriptions (Continued)

Bit	*field*	*symval*	Value	Description
18	CCMPB			Capture complete on channel B interrupt detected bit. (Data is not in memory until the DMA transfer is complete.)
				BT.656 or Y/C capture mode – CCMPB is set after capturing an entire field or frame (when F1C, F2C, or FRMC in VCBSTAT are set) depending on the CON, FRAME, CF1, and CF2 control bits in VCBCTL.
				Raw data mode – RDFE is not set, CCMPB is set when FRMC in VCBSTAT is set (when the data counter = the combined VCYSTOP/VCXSTOP value).
				TSI capture mode – CCMPB is set when FRMC in VCBSTAT is set (when the data counter = the combined VCYSTOP/VCXSTOP value).
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
17	COVRB			Capture overrun on channel B interrupt detected bit. COVRB is set when data in the FIFO was overwritten before being read out (by the DMA).
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
16	GPIO			Video port general purpose I/O interrupt detected bit.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
15	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
14	DCNA			Display complete not acknowledged. Indicates that the F1D, F2D, or FRMD bit that caused the display complete interrupt was not cleared prior to the start of the next gating field or frame.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.

† For CSL implementation, use the notation VP_VPIS_*field_symval*

Table 2–6. Video Port Interrupt Status Register (VPIS) Field Descriptions (Continued)

Bit	*field*	*symval*	Value	Description
13	DCMP			Display complete. Indicates that the entire frame has been driven out of the port. The DMA complete interrupt can be used to determine when the last data has been transferred from memory to the FIFO. DCMP is set after displaying an entire field or frame (when F1D, F2D or FRMD in VDSTAT are set) depending on the CON, FRAME, DF1, and DF2 control bits in VDCTL.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
12	DUND			Display underrun. Indicates that the display FIFO ran out of data.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
11	TICK			System time clock tick interrupt detected bit. BT.656, Y/C capture mode or raw data mode – Not used. TSI capture mode –TICK is set when the TCKEN bit in TSICTL is set and the desired number of system time clock ticks has occurred as programmed in TSITICKS.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
10	STC			System time clock interrupt detected bit. BT.656, Y/C capture mode or raw data mode – Not used. TSI capture mode – STC is set when the system time clock reaches an absolute time as programmed in TSISTCMPL and TSISTCMPM registers and the STEN bit in TSICTL is set.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
9–8	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation VP_VPIS_*field_symval*

Table 2–6. Video Port Interrupt Status Register (VPIS) Field Descriptions (Continued)

Bit	*field*	*symval*	Value	Description
7	LFDA			Long field detected on channel A interrupt detected bit. (A long field is only detected when the VRST bit in VCACTL is cleared to 0; when VRST = 1, a long field is always detected.)
				BT.656 or Y/C capture mode – LFDA is set when long field detection is enabled and VCOUNT is not reset before VCOUNT = YSTOP + 1.
				Raw data mode, or TSI capture mode or display mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
6	SFDA			Short field detected on channel A interrupt detected bit.
				BT.656 or Y/C capture mode – SFDA is set when short field detection is enabled and VCOUNT is reset before VCOUNT = YSTOP.
				Raw data mode, or TSI capture mode or display mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
5	VINTA2			Channel A field 2 vertical interrupt detected bit.
				BT.656, or Y/C capture mode or any display mode – VINTA2 is set when a vertical interrupt occurred in field 2.
				Raw data mode or TSI capture mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
4	VINTA1			Channel A field 1 vertical interrupt detected bit.
				BT.656, or Y/C capture mode or any display mode – VINTA1 is set when a vertical interrupt occurred in field 1.
				Raw data mode or TSI capture mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.

† For CSL implementation, use the notation VP_VPIS_*field_symval*

Table 2–6. Video Port Interrupt Status Register (VPIS) Field Descriptions (Continued)

Bit	*field*	*symval*	Value	Description
3	SERRA			Channel A synchronization error interrupt detected bit.
				BT.656 or Y/C capture mode – Synchronization parity error on channel A. An SERRA typically requires resetting the channel (RSTCH) or the port (VPRST).
				Raw data mode or TSI capture mode – Not used.
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
2	CCMPA			Capture complete on channel A interrupt detected bit. (Data is not in memory until the DMA transfer is complete.)
				BT.656 or Y/C capture mode – CCMPA is set after capturing an entire field or frame (when F1C, F2C, or FRMC in VCASTAT are set) depending on the CON, FRAME, CF1, and CF2 control bits in VCACTL.
				Raw data mode – If RDFE bit is set, CCMPA is set when F1C, F2C, or FRMC in VCASTAT is set (when the data counter = the combined VCYSTOP/VCXSTOP value) depending on the CON, FRAME, CF1, and CF2 control bits in VCACTL. If RDFE bit is not set, CCMPA is set when FRMC in VCASTAT is set (when the data counter = the combined VCYSTOP/VCXSTOP value).
				TSI capture mode – CCMPA is set when FRMC in VCASTAT is set (when the data counter = the combined VCYSTOP/VCXSTOP value).
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
1	COVRA			Capture overrun on channel A interrupt detected bit. COVRA is set when data in the FIFO was overwritten before being read out (by the DMA).
		NONE	0	No interrupt is detected.
		CLEAR	1	Interrupt is detected. Bit is cleared.
0	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation VP_VPIS_*field_symval*

Chapter 3

Video Capture Port

Video capture works by sampling video data on the input pins and saving it to the video port FIFO. When the amount of captured data reaches a programmed threshold level, a DMA is performed to move data from the FIFO into DSP memory. In some cases, color separation is performed on the incoming video data requiring multiple FIFOs and DMAs to be used.

The video port enables capture of both interlaced and progressive scan data. Interlaced capture can be performed on either a field-by-field or a frame-by-frame basis. A capture window specifies the data to be captured within each field. Frame and field synchronization can be performed using embedded sync codes or configurable control inputs allowing glueless interface to various encoders and ADCs.

Topic **Page**

3.1 Video Capture Mode Selection

The following modes/features of the video port are currently not supported at this time. The video port is capable of using these modes/features but no technical assistance will be given at this time.

Modes:

- **8-, 10-, 16-, and 20-bit raw capture mode**
- **TSI capture mode**

Features:

- **Noncontinuous capture**
- **Single frame capture**

The video capture module operates in one of nine modes as listed in Table 3–1. The transport stream interface (TSI) selection is made using the TSI bit in the video port control register (VPCTL). The CMODE bits are in the video capture channel *x* control register (VC*x*CTL). The Y/C and 16/20-bit raw capture modes may only be selected for channel A and only if the DCDIS bit in VPCTL is cleared to 0.

When operating as a raw video capture channel, no data selection or data interpretation is performed. The 16/20-bit raw capture mode is designed to accept data from A/D converters with resolution higher than eight bits (used, for example, in medical imaging).

Table 3−1. Video Capture Mode Selection

TSI Bit	CMODE Bits	Mode	Description
0	000	8-Bit ITU-R BT.656 Capture	Digital video input is in YCbCr 4:2:2 with 8-bit resolution multiplexed in ITU-R BT.656 format.
0	001	10-Bit ITU-R BT.656 Capture	Digital video input is in YCbCr 4:2:2 with 10-bit resolution multiplexed in ITU-R BT.656 format.
0	010	8-Bit Raw Capture†	Raw 8-bit data capture at sampling rates up to 80 MHz.
0	011	10-Bit Raw Capture†	Raw 8-bit or 10-bit data capture at sampling rates up to 80 MHz.
0	100	8-Bit Y/C Capture	Digital video input is in YCbCr 4:2:2 with 8-bit resolution on parallel Y and Cb/Cr multiplexed channels.
0	101	10-Bit Y/C Capture	Digital video input is in YCbCr 4:2:2 with 10-bit resolution on parallel Y and Cb/Cr multiplexed channels.
0	110	16-Bit Raw Capture†	Raw 16-bit data capture at sampling rates up to 80 MHz.
0	111	20-Bit Raw Capture†	Raw 20-bit data capture at sampling rates up to 80 MHz.
1	010	TSI Capture†	8-bit parallel TSI capture at rates up to 30 MHz.

† This mode of the video port is currently not supported at this time.

3.2 BT.656 Video Capture Mode

The BT.656 capture mode captures 8-bit or 10-bit 4:2:2 luma and chroma data multiplexed into a single data stream. Video data is conveyed in the order Cb,Y,Cr,Y,Cb,Y,Cr, etc. where the sequence Cb,Y,Cr refers to co-sited luma and chroma samples and the following Y value corresponds to the next luminance sample. The data stream is demultiplexed and each component is written in packed form into separate FIFOs for transfer into Y, Cb, and Cr buffers in DSP memory. (This is commonly called planar format.) The packing and order of the samples is determined by the sample size (8-bit or 10-bit) and the selected endianess of the DSP.

The ITU-BT.656 standard provides for either 8-bit or 10-bit component samples. When 10-bit samples are used, the 2 least significant bits are considered fractional values. Thus for 8-bit operation, input data is aligned to the most significant bits (9–2) of the input and the two least-significant bits are ignored.

In BT.656 video capture mode, data bytes in which the 8 most significant bits are all set to 1 (FF.0h, FF.4h, FF.8h, FF.Ch) or are all set to 0 (00.0h, 00.4h, 00.8h, 00.Ch) are reserved for data identification purposes and consequently, only 254 of the possible 256 8-bit words (or 1016 of 1024 10-bit words) may be used to express signal value.

3.2.1 BT.656 Capture Channels

In dual channel operation, the video port can support capture of two BT.656 data streams or one BT.656 data stream and one raw data stream. In the latter case, the BT.656 stream may occur on either Channel A or Channel B. In either case, the BT.656 stream(s) must have embedded timing reference codes and the appropriate VCTL input must be used as a CAPEN signal.

If the port is configured for single channel operation, capture will take place on Channel A only. The unused half of the VDATA bus may be used for GPIO or for another peripheral function. For single channel operation, non-standard BT.656 data streams without embedded timing reference codes are supported through the use of the timing control (VCTL) input signals.

3.2.2 BT.656 Timing Reference Codes

For standard digital video, there are two reference signals, one at the beginning of each video data block (start of active video, SAV), and one at the end of each video block (end of active video, EAV). (Technically each line begins with the EAV code and ends just before the subsequent EAV code.) Each timing reference signal consists of a four sample sequence in the following format: FF.Ch 00.0h 00.0h XY.0h. (The FFh and 00h values are reserved for use in these timing reference signals.) The first three bytes are a fixed preamble. The fourth byte contains information defining field identification, the state of field blanking and state of line blanking. The assignment of these bits within the timing reference signal is listed in Table 3–2. Note that the two least-significant bits should be ignored even during 10-bit operation.

Table 3–2. BT.656 Video Timing Reference Codes

Data Bit	1st Byte (FFh)	2nd Byte (00h)	3rd Byte (00h)	4th Byte (XYh)
9 (MSB)	1	0	0	1
8	1	0	0	F (field)†
7	1	0	0	V (vertical blanking)‡
6	1	0	0	H (horizontal blanking)§
5	1	0	0	P3 (protection bit 3)¶
4	1	0	0	P2 (protection bit 2)¶
3	1	0	0	P1 (protection bit 1)¶
2	1	0	0	P0 (protection bit 0)¶
1	x	x	x	x
0	x	x	x	x

† F = 0 during Field 1; F = 1 during Field 2
‡ V = 0 elsewhere; V = 1 during field blanking
§ H = 0 in SAV; H = 1 in EAV
¶ P0, P1, P2, and P3: Depends on F, V, and H state.

Bits P0, P1, P2, and P3 have different states depending on the state of bits F, V, and H as shown in Table 3–3.

Table 3–3. BT.656 Protection Bits

Line Information Bits			Protection Bits			
F	**V**	**H**	**P3**	**P2**	**P1**	**P0**
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	0	0	0	1

The protection bits allow the port to implement a DEDSEC (double error detection, single error correction) function on the received video timing reference code. The corrected values for the F, H, and V bits based on the protection bit values are shown in Table 3–4. The – entries indicate detected double bit errors that cannot be corrected. Detection of these errors causes the SERR*x* bit in the video port interrupt status register (VPIS) to be set.

Table 3–4. Error Correction by Protection Bits

Received P_3–P_0 Bits	Received F, V, and H Bits							
	000	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	000	000	000	–	000	–	–	111
0001	000	–	–	111	–	111	111	111
0010	000	–	–	011	–	101	–	–
0011	–	–	010	–	100	–	–	111
0100	000	–	–	011	–	–	110	–
0101	–	001	–	–	100	–	–	111
0110	–	011	011	011	100	–	–	011

Table 3–4. Error Correction by Protection Bits (Continued)

Received P_3–P_0 Bits	Received F, V, and H Bits							
	000	001	010	011	100	101	110	111
0111	100	–	–	011	100	100	100	–
1000	000	–	–	–	–	101	110	–
1001	–	001	010	–	–	–	–	111
1010	–	101	010	–	101	101	–	101
1011	010	–	010	010	–	101	010	–
1100	–	001	110	–	110	–	110	110
1101	001	001	–	001	–	001	110	–
1110	–	–	–	011	–	101	110	–
1111	–	001	010	–	100	–	–	–

3.2.3 BT.656 Image Window and Capture

The BT.656 format is an interlaced format consisting of two fields. The video port allows capture of one or both fields. The captured image is a subset of each field and can be larger or smaller than the active video region. The captured image position is defined by the VC*x*STRT1 and VC*x*STOP1 registers for field 1, and the VC*x*STRT2 and VC*x*STOP2 registers for field 2. The VCXSTART and VCXSTOP bits set the horizontal window position for the field relative to the HCOUNT pixel counter. The VCYSTART and VCYSTOP bits set the vertical position relative to the VCOUNT line counter. This is shown in Figure 3–1.

HCOUNT increments on every chroma sample period (every other VCLKIN rising edge) for which capture is enabled. Once VCOUNT = YSTART, line capture begins when HCOUNT = XSTART. It continues until HCOUNT = XSTOP. A field's capture is complete when HCOUNT = VCXSTOP and VCOUNT = VCYSTOP.

Figure 3−1. Video Capture Parameters

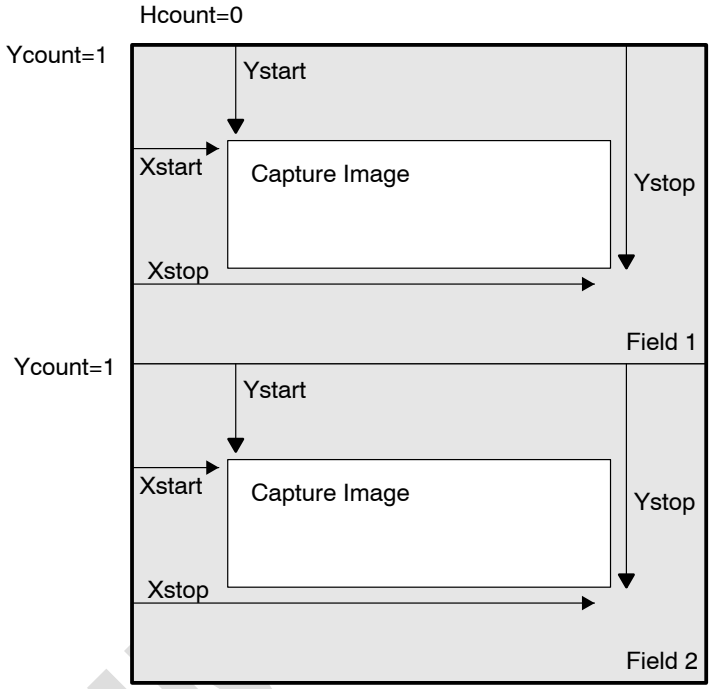


Table 3−5 shows common digital camera standards and the number of fields per second, number of active lines per field, and the number of active pixels per line.

Table 3−5. Common Video Source Parameters

Video Source	Number of Active Lines (Field 1/Field 2)	Number of Active Pixels	Field Rate (Hz)
square pixel 60Hz/525 lines	240/240	640	60
BT.601 60 Hz/525 lines	244/243	720	60
square pixel 50Hz/625 lines	288/288	768	50
BT.601 50 Hz/625 lines	288/288	720	50

For the BT.656 video capture mode, the FIFO buffer is divided into three sections (three buffers). One section is 1280 bytes deep and is dedicated for storage of Y data samples. The other two sections are dedicated for storage of Cb and Cr data samples, respectively. The buffers for Cb and Cr samples are each 640 bytes deep. The incoming video data stream is separated into Y, Cb, and Cr data streams, scaled (if selected), and the Y, Cb, and Cr buffers are filled. Each of the three buffers has a memory-mapped location associated with it; YSRC, CBSRC, and CRSRC. The YSRC, CBSRC, and CRSRC locations are read only and are used by DMAs to access video data samples stored in the FIFOs.

If video capture is enabled (BLKCAP bit in VC*x*CTL is cleared), pixels in the capture window are captured in the Y, Cb, and Cr buffers. The video capture module uses the YEVT, CbEVT, and CrEVT events to notify the DMA controller to copy data from the capture buffers to the DSP memory. The number of doublewords required to generate the events is set by the VCTHRLD*n* bits in VC*x*THRLD. On every YEVT, the DMA should move data from the Y buffer to DSP memory using the YSRC location as the source address. On every CbEVT, the DMA should move data from the Cb buffer to DSP memory using the CBSRC location as the source address. On every CrEVT, the DMA should move data from the Cr buffer to DSP memory using the CRSRC location as the source address. Note that transfer size from the Cb and Cr buffers is half of the transfer size from the Y buffer since for every four Y samples, there are two Cb and two Cr samples.

3.2.4 BT.656 Data Sampling

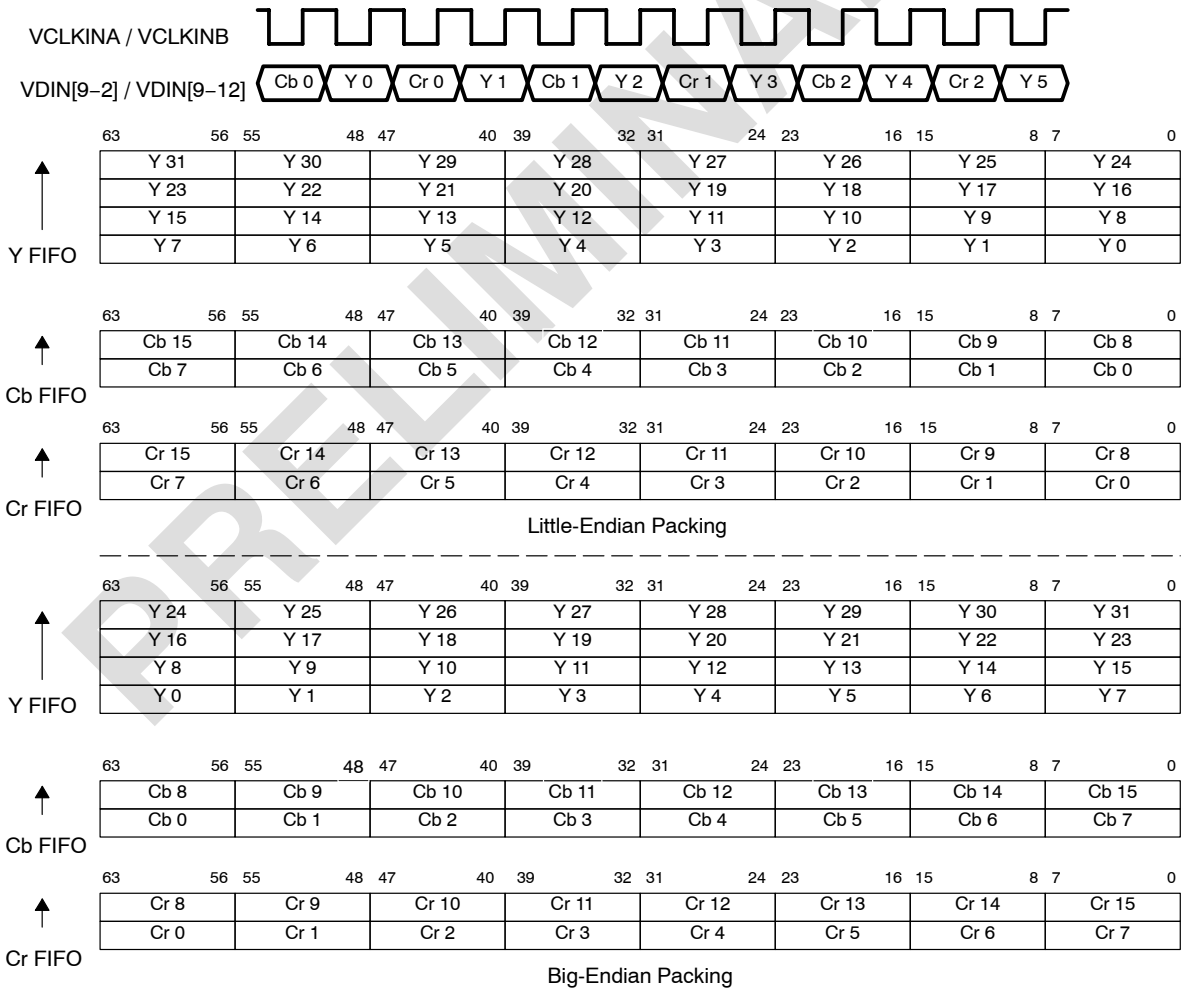
Incoming data (including timing codes) are sampled and the HCOUNT counter advanced only on clock cycles for which the CAPEN input is active. Inputs when CAPEN is inactive are ignored. The timing reference codes are recognized only when three sequential samples with CAPEN valid are the FFh, 00h, 00h sequence. A non-00h sample after the FFh or after the first 00h causes the timing reference recognition logic to be reset and to look for FFh again. (Unsampled data; those with CAPEN inactive; in the middle of a timing reference do not cause the recognition logic to be reset since these are not considered to be valid inputs.)

3.2.5 BT.656 FIFO Packing

Captured data is always packed into 64-bits before being written into the capture FIFO(s). The packing and byte ordering is dependant upon the capture data size and the device endian mode. For little-endian operation (default), data is packed into the FIFO from right to left; for big-endian operation, data is packed from left to right.

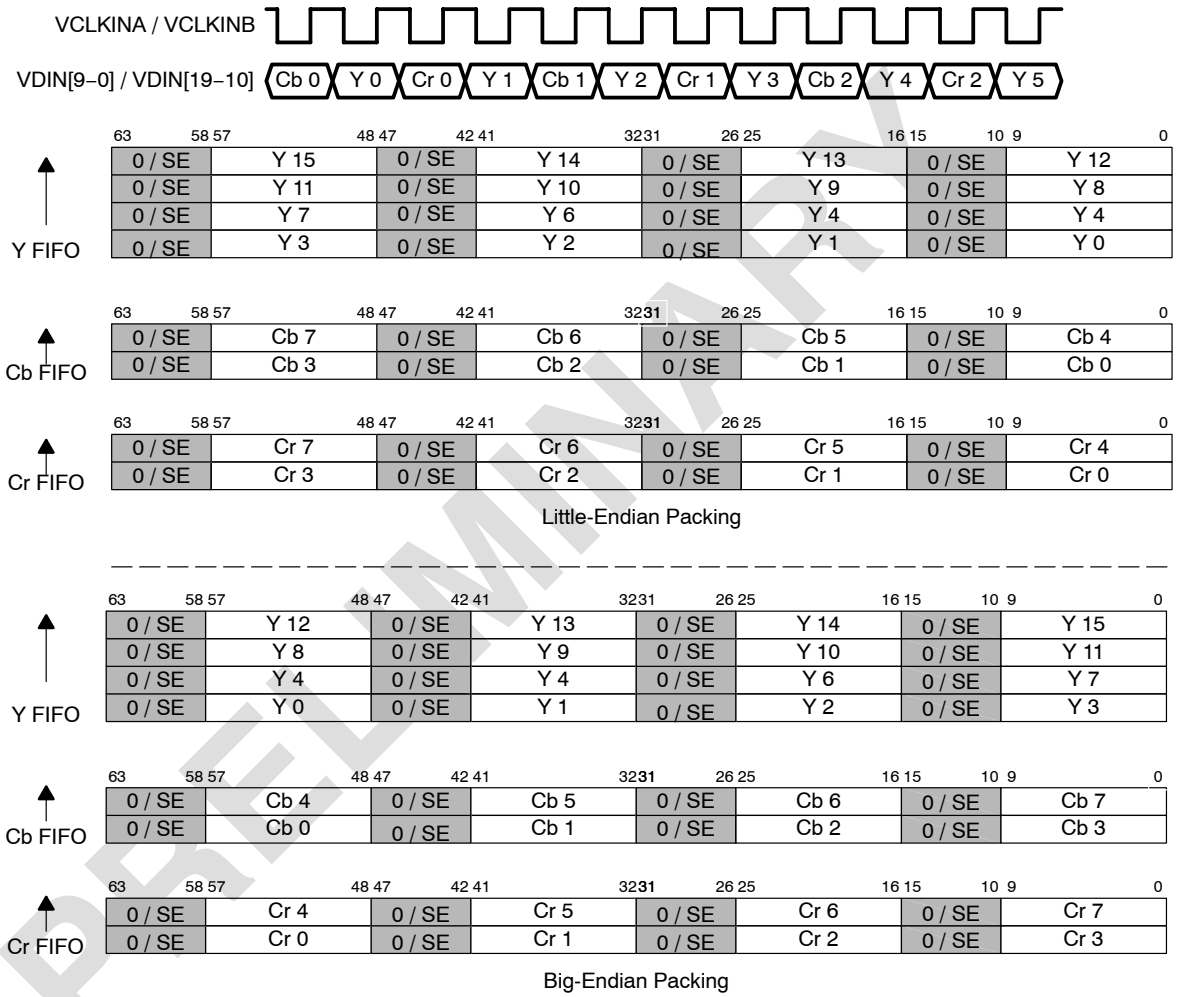
The 8-bit BT.656 mode uses three FIFOs for color separation. Four samples are packed into each word as shown in Figure 3–2.

Figure 3–2. 8-Bit BT.656 FIFO Packing



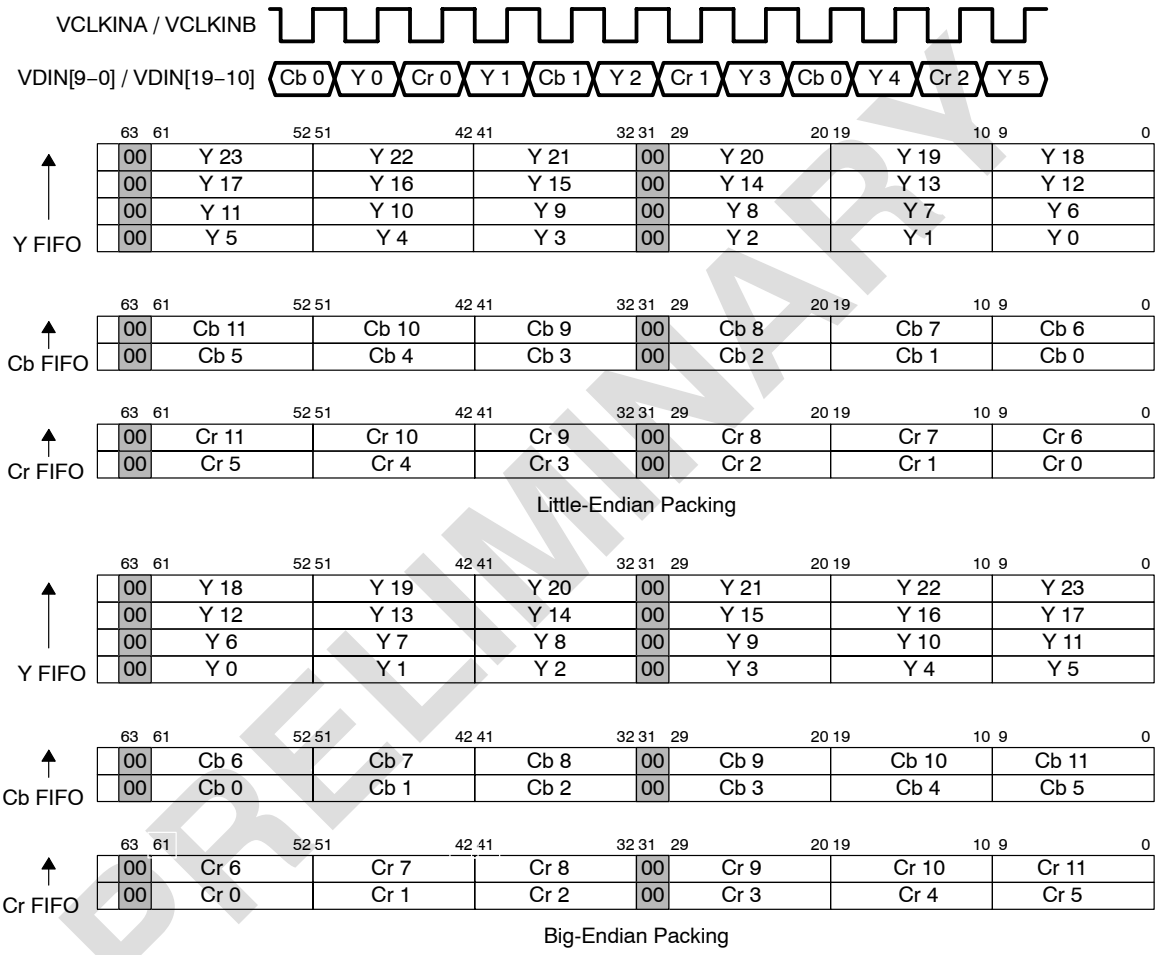
The 10-bit BT.656 mode uses three FIFOs for color separation. Two samples are packed into each word with zero or sign extension as shown in Figure 3–3.

Figure 3–3. 10-Bit BT.656 FIFO Packing



The 10-bit BT.656 dense mode uses three FIFOs for color separation. Three samples are packed into each word with zero extension to provide increased DMA bandwidth as shown in Figure 3–4.

Figure 3–4. 10-Bit BT.656 Dense FIFO Packing



3.3 Y/C Video Capture Mode

The Y/C capture mode is similar to the BT.656 capture mode but captures 8 or 10-bit 4:2:2 data on separate luma and chroma data streams. One data stream contains Y samples and the other stream contains multiplexed Cb and Cr samples co-sited with every other Y sample. The Y samples are written into a Y FIFO and the chroma samples are demultiplexed and written into separate Cb and Cr FIFOs for transfer into Y, Cb, and Cr buffers in DSP memory. The packing and order of the samples is determined by the sample size (8-bit or 10-bit) and the device endian mode.

The Y/C capture mode supports HDTV standards such as SMPTE260, SMPTE296, and BT.1120 with embedded EAV and SAV codes. It also supports SDTV YCbCr modes that use separate control signals (sometimes called CCIR601 mode)

As with the BT.656 capture mode, data bytes where the 8 most-significant bits are all set to 1 (FF.0h, FF.4h, FF.8h, FF.Ch) or are all cleared to 0 (00.0h, 00.4h, 00.8h, 00.Ch) are reserved for data identification purposes and consequentially only 254 of the possible 256 8-bit words (or 1016 of 1024 10-bit words) may be used to express signal value.

3.3.1 Y/C Capture Channels

Because Y/C mode requires the entire VDATA bus, only single channel operation is supported. If the DCHDIS bit in VPCTL is set, then Y/C mode cannot be selected. Y/C capture takes place on channel A only. Both embedded timing references and external control inputs are supported.

3.3.2 Y/C Timing Reference Codes

Many high-resolution Y/C interface standards provide for embedded timing reference codes. These codes are identical to those used in the BT.656 standard except that they appear on both the luma (Y) and chroma (CbCr) data streams in parallel.

3.3.3 Y/C Image Window and Capture

The SDTV Y/C format (CCIR601) is an interlaced format consisting of two fields just like BT.656. HDTV Y/C formats may be interlaced or progressive scan. For interlaced capture, the capture windows are programmed identically to BT.656 mode. For progressive scan formats, only field1 is used.

In Y/C mode, HCOUNT increments on every luma sample period (every VCLKINA rising edge) for which capture is enabled. Once YCOUNT = YSTART, line capture begins when HCOUNT = XSTART. It continues until HCOUNT = XSTOP. A field's capture is complete when HCOUNT = VCXSTOP and VCOUNT = VCYSTOP.

For the Y/C video capture mode, the FIFO buffer is divided into three sections (three buffers). One section is 2560 bytes deep and is dedicated for storage of Y data samples. The other two sections are dedicated for storage of Cb and Cr data samples, respectively. The buffers for Cb and Cr samples are each 1280 bytes deep. The incoming video data stream is separated into Y, Cb, and Cr data streams, scaled (if selected) and the Y, Cb, and Cr buffers are filled. Each of the three buffers has a memory-mapped location associated with it; YSRC, CBSRC, and CRSRC. The YSRC, CBSRC, and CRSRC locations are read only and are used by DMAs to access video data samples stored in the FIFOs. Reads must always be 64 bits.

If video capture is enabled, pixels in the capture window are captured in the Y, Cb, and Cr buffers. The video capture module uses the YEVT, CbEVT, and CrEVT events to notify the DMA controller to copy data from the capture buffers to the DSP memory. The number of pixels required to generate the events is set by the VCTHRLD*n* bits in VC*x*CTL (the VCTHRLD*n* value must be an even number for Y/C mode). The capture module generates the events after VCTHRLD new pixels have been received. On every YEVT, the DMA should move data from the Y buffer to DSP memory using the YSRC register as the source address. On every CbEVT, the DMA should move data from the Cb buffer to DSP memory using the CBSRC register as the source address. On every CrEVT, the DMA should move data from the Cr buffer to DSP memory using the CRSRC register as the source address. Note that transfer size from the Cb and Cr buffers is half of the transfer size from the Y buffer since for every four Y samples, there are two Cb and two Cr samples.

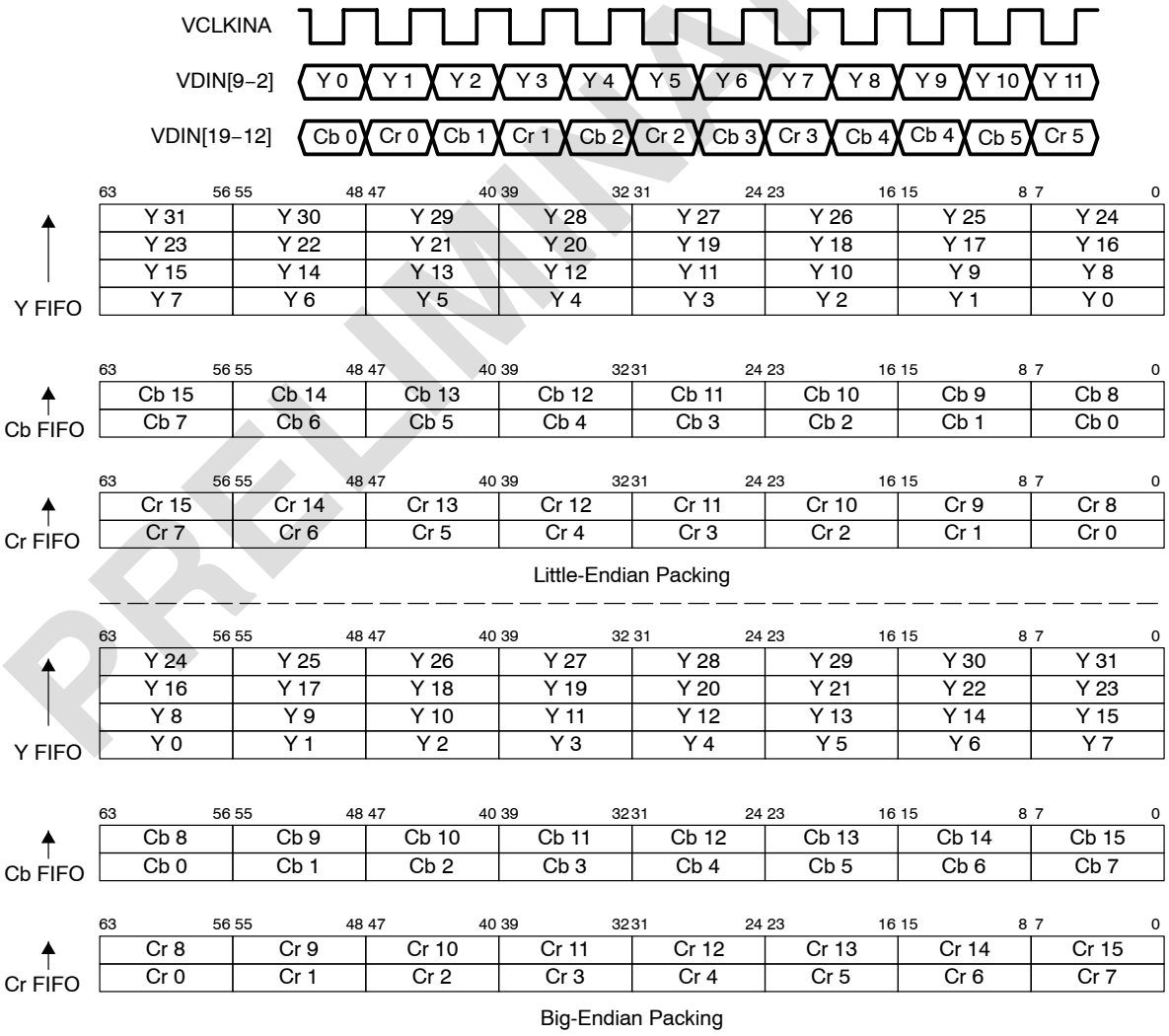
The three DMA events are generated simultaneously when VCTHRLD is reached. Each event is reenabled when the first read of the respective FIFO by the requested DMA begins.

3.3.4 Y/C FIFO Packing

Captured data is always packed into 64 bits before being written into the capture FIFO(s). The packing and byte ordering is dependant upon the capture data size and the device endian mode. For little-endian operation (default), data is packed into the FIFO from right to left; for big-endian operation, data is packed from left to right.

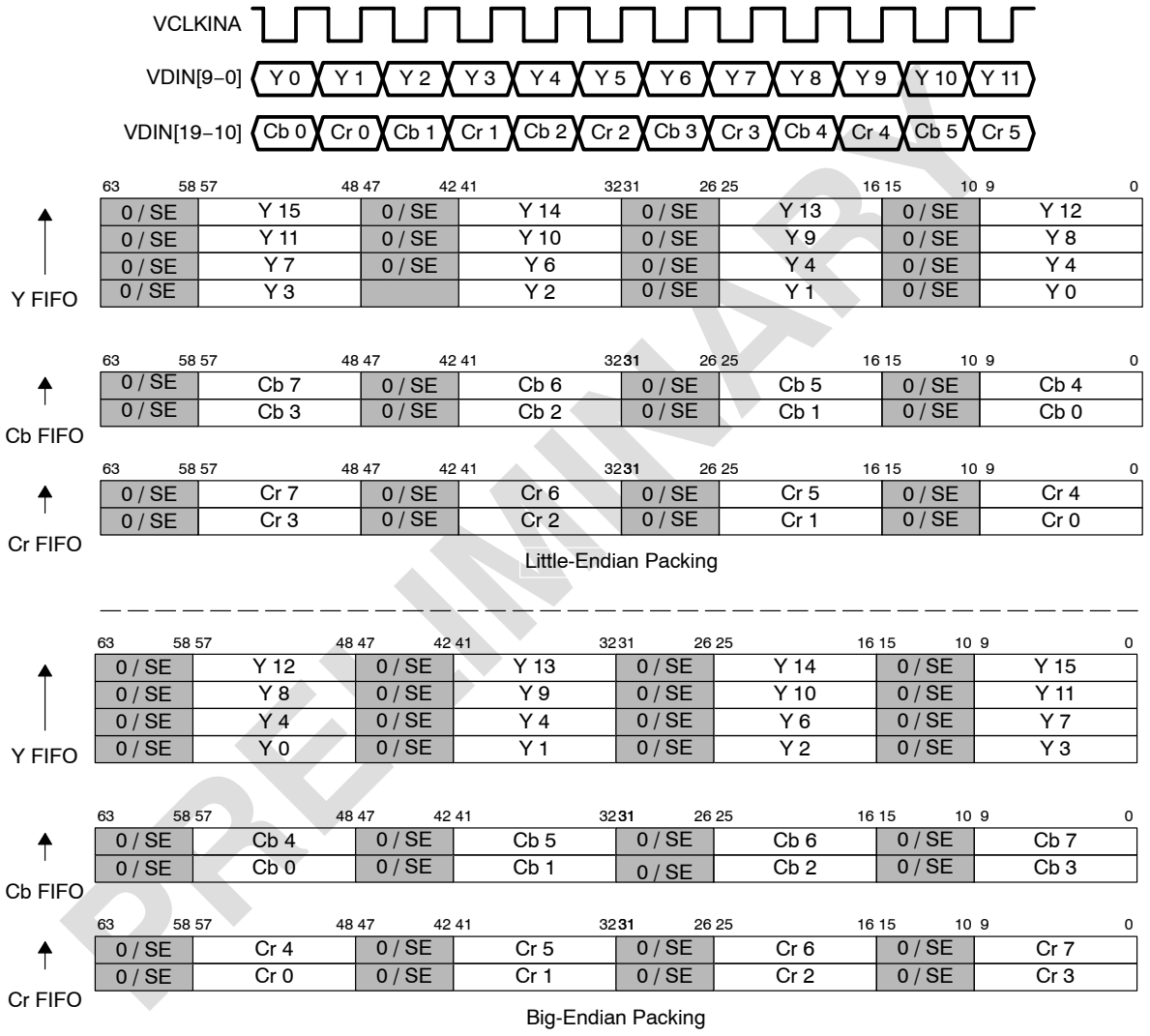
The 8-bit Y/C mode uses three FIFOs for color separation. Four samples are packed into each word as shown in Figure 3–5.

Figure 3–5. 8-Bit Y/C FIFO Packing



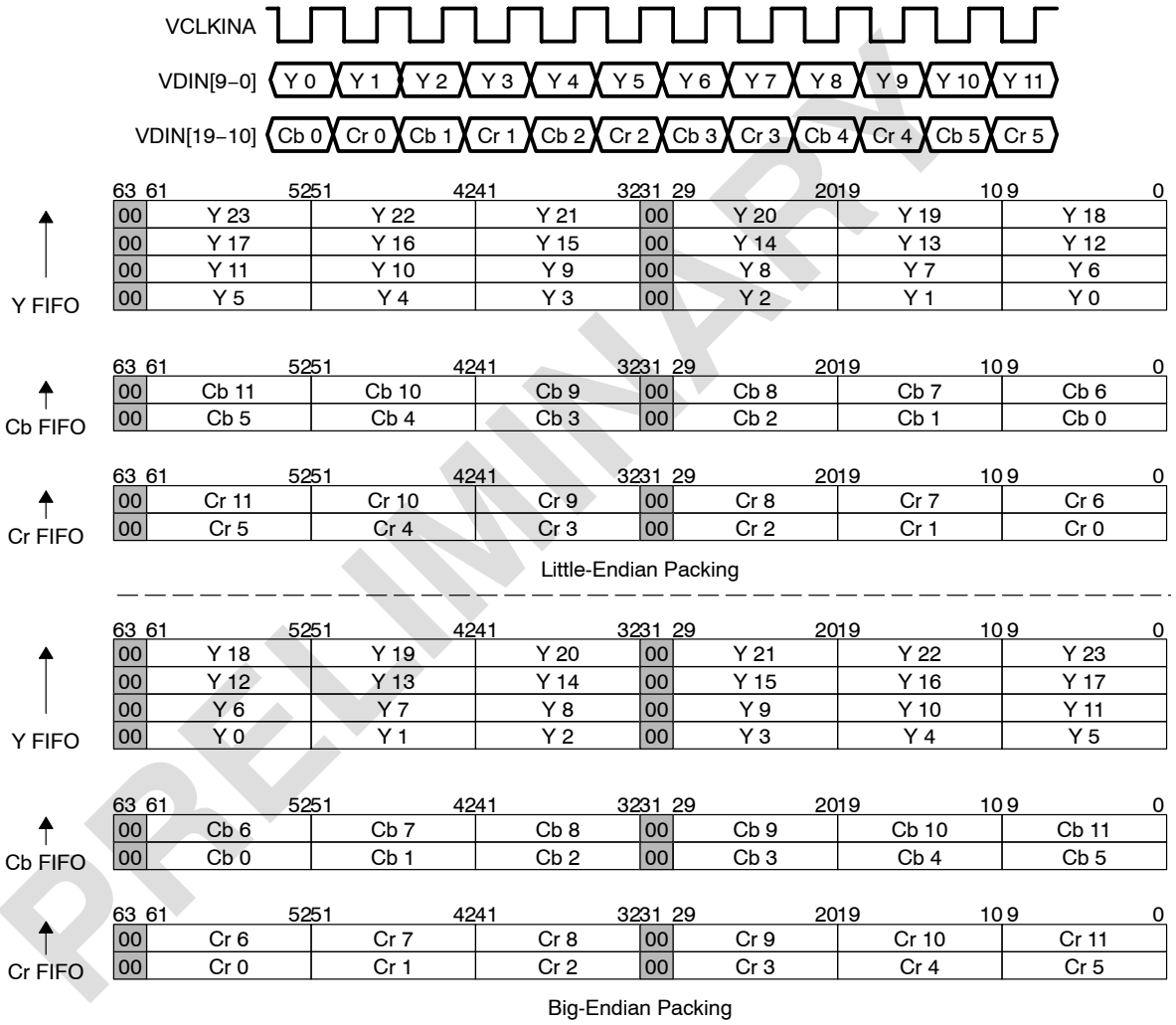
The 10-bit Y/C mode uses three FIFOs for color separation. Two samples are packed into each word with zero or sign extension as shown in Figure 3–6.

Figure 3–6. 10-Bit Y/C FIFO Packing



The 10-bit Y/C dense mode uses three FIFOs for color separation. Three samples are packed into each word with zero extension to provide increased DMA bandwidth as shown in Figure 3–7.

Figure 3–7. 10-Bit Y/C Dense FIFO Packing



3.4 BT.656 and Y/C Mode Field and Frame Operation

Because DMAs are used to transfer data from the capture FIFOs to memory, there is a large amount of flexibility in the way that capture fields and frames are transferred and stored in memory. In some cases, for example a DMA structure can be created to provide a set of ping-pong or round-robin memory buffers to which a continuous stream of fields are stored without DSP intervention. In other cases, the DSP may need to modify DMA pointer addresses after each field or frame is captured. In some applications, only one field may be captured and the other ignored completely, or a frame may need to be ignored in order to have time to process a previous frame. The video port addresses these issues by providing programmable control over different aspects of the capture process.

3.4.1 Capture Determination and Notification

Noncontinuous capture and single frame capture are currently not supported at this time.

The video port treats the capture of every field as a separate operation. In order to accommodate various capture scenarios, DMA structures, and processing flows, the video port employs a flexible capture and DSP notification method. This is programmed using the CON, FRAME, CF1, and CF2 bits in VC*x*CTL.

The CON bit controls the capture of multiple fields or frames. When CON = 1, continuous capture is enabled, the video port captures incoming fields (assuming the VCEN bit is set) without the need for DSP interaction. It relies on a DMA structure with circular buffering capability to service the capture FIFOs. When CON = 0, continuous capture is disabled, the video port sets a field or frame capture complete bit (F1C, F2C, or FRMC) in VC*x*STAT upon the capture of each field as determined by the state of the other capture control bits (FRAME, CF1, and CF2). Once the capture complete bit is set, at most, one more field or frame can be received before capture operation is halted. This prevents subsequent data from overwriting previous fields until the DSP has a chance to update DMA pointers or process those fields. When a capture halt occurs, the video port stops capturing data (for the halted field). It then checks the appropriate capture complete bit at the start of each subsequent field and resumes capture if the bit has been cleared.

The CON, FRAME, CF1, and CF2 bits encode the capture operations as listed in Table 3–6.

Table 3–6. BT.656 and Y/C Mode Capture Operation

VCxCTL Bit				
CON	**FRAME**	**CF2**	**CF1**	**Operation**
0	0	0	0	Reserved
0	0	0	1	†Noncontinuous field 1 capture. Capture only field 1. F1C is set after field 1 capture and causes CCMPx to be set. The F1C bit must be cleared by the DSP before capture can continue. (The DSP has the entire field 2 time to clear F1C before next field 1 begins.) Can also be used for single progressive frame capture. (The DSP has vertical blanking time to clear F1C before next frame begins.)
0	0	1	0	†Noncontinuous field 2 capture. Capture only field 2. F2C is set after field 2 capture and causes CCMPx to be set. The F2C bit must be cleared by the DSP before capture can continue. (The DSP has the entire field 1 time to clear F2C before next field 2 begins.)
0	0	1	1	†Noncontinuous field 1 and field 2 capture. Capture both fields. F1C is set after field 1 capture and causes CCMPx to be set. The F1C bit must be cleared by the DSP before another field 1 capture can occur. (The DSP has the entire field 2 time to clear F1C before next field 1 begins.) F2C is set after field 2 capture and causes CCMPx to be set. The F2C bit must be cleared by the DSP before another field 2 capture can occur. (The DSP has the entire field 1 time to clear F2C before next field 2 begins.)
0	1	0	0	†Noncontinuous frame capture. Capture both fields. FRMC is set after field 2 capture and causes CCMPx to be set. Capture halts upon completion of the next frame unless the FRMC bit is cleared. (The DSP has the entire next frame time to clear FRMC.)
0	1	0	1	†Noncontinuous progressive frame capture. Capture field 1. FRMC is set after field 1 capture and causes CCMPx to be set. Capture halts upon completion of the next frame unless the FRMC bit is cleared. (The DSP has the entire next frame time to clear FRMC.)
0	1	1	0	Reserved
0	1	1	1	†Single frame capture. Capture both fields. FRMC is set after field 2 capture and causes CCMPx to be set. Capture halts until the FRMC bit is cleared. (The DSP has the field 2 to field 1 vertical blanking time to clear FRMC.)
1	0	0	0	Reserved

† Noncontinuous capture and single frame capture are currently not supported at this time.

Table 3−6. BT.656 and Y/C Mode Capture Operation (Continued)

VC*x*CTL Bit				
CON	**FRAME**	**CF2**	**CF1**	**Operation**
1	0	0	1	Continuous field 1 capture. Capture only field 1. F1C is set after field 1 capture and causes CCMPx to be set (CCMPx interrupt can be disabled). The video port continues capturing field 1 fields, regardless of the state of F1C.
1	0	1	0	Continuous field 2 capture. Capture only field 2. F2C is set after field 2 capture and causes CCMPx to be set (CCMPx interrupt can be disabled). The video port continues capturing field 2 fields, regardless of the state of F2C.
1	0	1	1	Reserved
1	1	0	0	Continuous frame capture. Capture both fields. FRMC is set after field 2 capture and causes CCMPx to be set (CCMPx interrupt can be disabled). The video port continues capturing frames, regardless of the state of FRMC.
1	1	0	1	Continuous progressive frame capture. Capture field 1. FRMC is set after field 1 capture and causes CCMPx to be set (CCMPx interrupt can be disabled). The video port continues capturing frames, regardless of the state of FRMC. (Functions identically to continuous field 1 capture mode except the FRMC bit is used instead of the F1C bit.)
1	1	1	0	Reserved
1	1	1	1	Reserved

† Noncontinuous capture and single frame capture are currently not supported at this time.

3.4.2 Vertical Synchronization

The video port uses a capture window to determine which incoming data samples to capture in each field. The capture module uses a vertical line counter (VCOUNT) to track which video line is currently being received. The line counter is compared to the appropriate capture window start (VCYSTART1 or VCYSTART2) and stop (VCYSTOP1 or VCYSTOP2) values for the current field to determine if the current line is within the capture window. In order to correctly align the capture window within the field, the capture module must know which line should correspond to the first line of the field, that is, when to reset the line counter. This point may vary depending on the type of capture being performed and the signals available for vertical synchronization. The video port allows the vertical counter reset trigger to be determined by programming the EXC and VRST bits in VC*x*CTL. The encoding of these bits is shown in Table 3−7. Note that VModes 2 and 3 are only available for single channel operation (channel A).

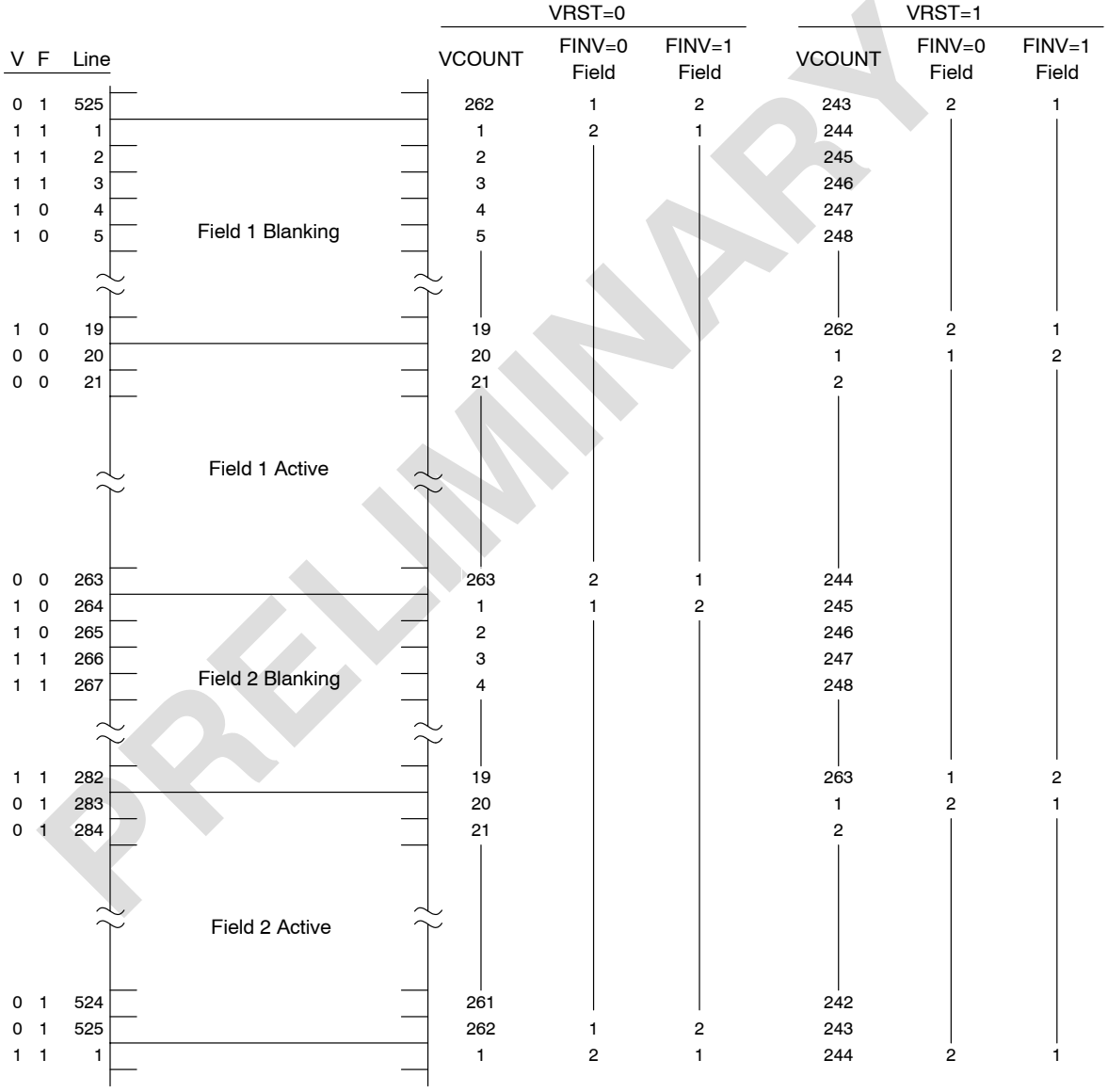
Table 3−7. Vertical Synchronization Programming

	VC*x*CTL Bit		
VMode	**EXC**	**VRST**	**Vertical Counter Reset Point**
0	0	0	First EAV with V=1 after EAV with V=0 − beginning of vertical blanking period. VCOUNT increments on each EAV.
1	0	1	First EAV with V=0 after EAV with V=1 − first active line. VCOUNT increments on each EAV.
2	1	0	On HCOUNT reset after VCTL1 input active edge − beginning of vertical blanking or vertical sync period. (VCTL1 must be configured as vertical control signal). VCOUNT increments when HCOUNT is reset.
3	1	1	On HCOUNT reset after VCTL1 input inactive edge − end of vertical sync or first active scan line. (VCTL1 must be configured as vertical control signal). VCOUNT increments when HCOUNT is reset.

VMode 0 is used for BT.656 or Y/C capture (with embedded control) and corresponds to most digital video standards that number lines beginning with the start of vertical blanking. VMode 1 can also be used for BT.656 or Y/C capture but counts from the first active video line. This makes field detection more straightforward in some instances (see section 3.4.4) and allows the VCYSTART*n* bit to be set to 1, but also has the effect of associating vertical blanking periods with the end of the previous field rather than the beginning of the current field. (This could be an issue when capturing VBI data.) VCOUNT operation for VMode 0 and VMode 1 is shown in Figure 3−8.

VMode 2 and VMode 3 are used for BT.656 or Y/C capture without embedded EAV/SAV codes and allow alignment with either the active or inactive edge of the vertical control signal on VCTL1. This can be a VBLNK or VSYNC signal from the video decoder.

Figure 3−8. VCOUNT Operation Example (EXC = 0)



3.4.3 Horizontal Synchronization

Horizontal synchronization determines when the horizontal pixel/sample counter is reset. The EXC and HRST bits in VC*x*CTL allow you to program the event that triggers the start of a line. The encoding of these bits is shown in Table 3–8.

Table 3–8. Horizontal Synchronization Programming

	VCxCTL Bit		
HMode	**EXC**	**HRST**	**Horizontal Counter Reset Point**
0	0	0	EAV code (H=1) – beginning of horizontal blanking.
1	0	1	SAV code (H=0) – Start of active video.
2	1	0	VCTL0 input active edge – beginning of horizontal blanking or horizontal sync period. (VCTL0 must be configured as a horizontal control signal.)
3	1	1	VCTL0 input inactive edge – first active pixel on line or end of horizontal sync. (VCTL0 must be configured as a horizontal control signal.)

HMode 0 is used for BT.656 or Y/C capture (with embedded control) and corresponds to the idea that each line begins with the horizontal blanking period. It does not align with most standards that start counting with the first active pixel; therefore, is only useful if capturing of HANC data before the SAV code is desired. HMode 1 is the default mode and corresponds to most digital video standards by making the first active pixel pixel0. It has the effect of associating horizontal blanking periods with the end of the previous line rather than the beginning of the line, but this is only an issue if you try to capture HANC data. In either mode, HCOUNT increments on every VCLKIN edge for Y/C operation and on every other VCLKIN edge for BT.656 operation but only when CAPEN is active. HCOUNT operation for HMode 1 and HMode 2 is shown in Figure 3–9.

HMode 2 and HMode 3 are used for BT.656 or Y/C capture without embedded EAV/SAV code and allow alignment with either the beginning of the horizontal blanking period or the first active pixel, or the beginning or end of horizontal sync depending on the VCTL0 input. When VCTL0 is configured as a horizontal control input, no external CAPEN signal is available so the CAPEN signal is considered to always be active. HCOUNT operation for HMode 3 and HMode 4 is shown in Figure 3–10 for VCTL operating as either HSYNC or AVID.

Figure 3−9. HCOUNT Operation Example (EXC = 0)

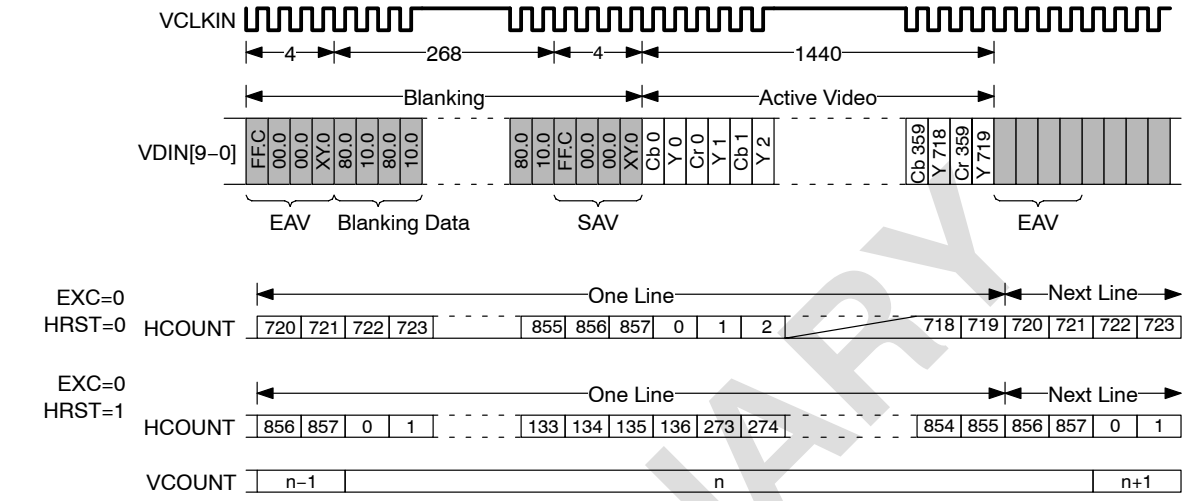
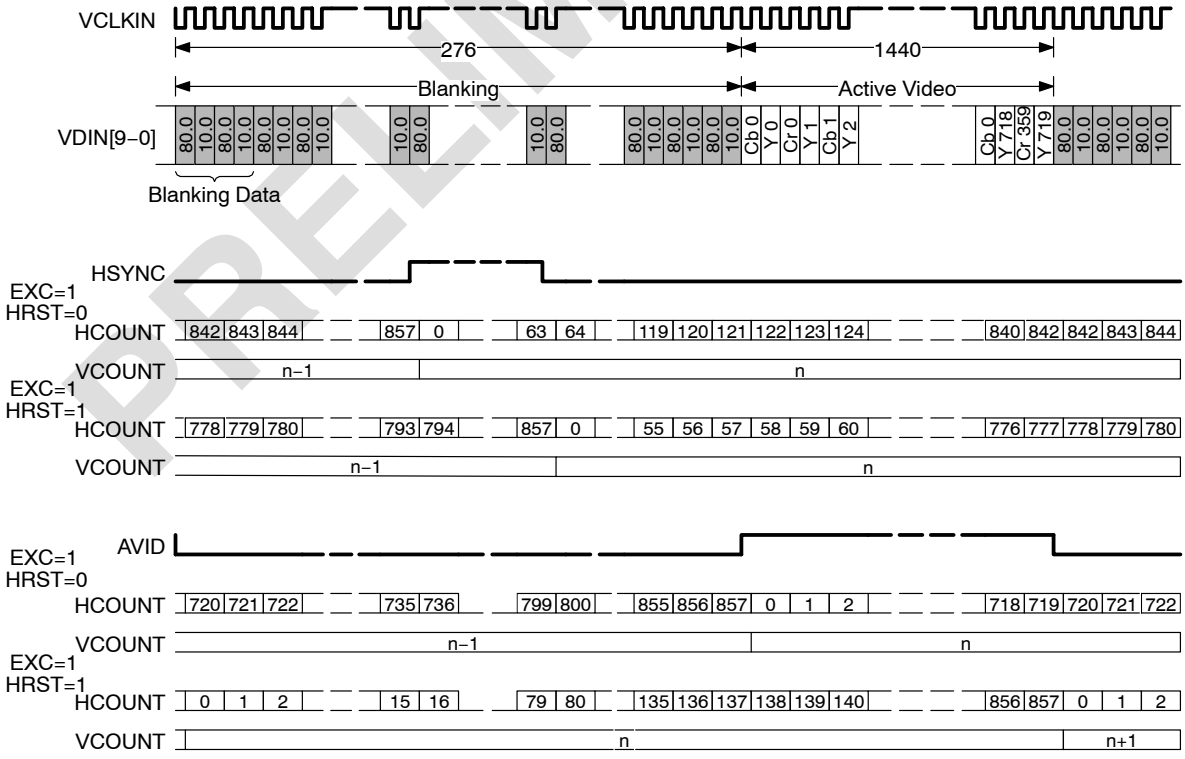


Figure 3−10. HCOUNT Operation Example (EXC = 1)



3.4.4 Field Identification

In order to properly synchronize to the source data stream and capture the correct fields, field identification needs to be performed. Field identification is made using one of three methods: EAV, field indicator input, or field detect logic. The field identification method is determined by the EXC, FLDD, and FINV bits in VC*x*CTL.

Table 3–9. Field Identification Programming

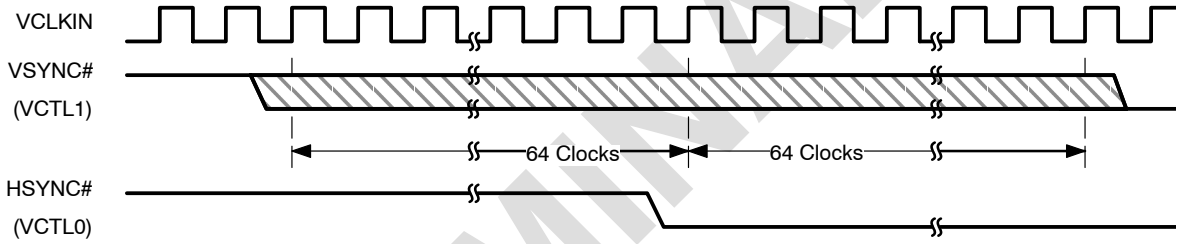
VC*x*CTL Bit		
EXC	**FLDD**	**Field Detect Method**
0	0	EAV code
0	1	EAV code
1	0	Use FID input
1	1	Use field detect (from HSYNC and VSYNC inputs)

In the BT.656 standard and in many Y/C standards, a field identification (F) bit is contained in EAV and SAV codes embedded in the data stream. In the EAV field detect method, the F bit in the EAV of the first line of every field is checked. If F = 0, then the current field is defined as field 1. If F = 1, then the current field is defined as field 2. Depending on how the first line of a field is defined (as determined by the VRST bit in VC*x*CTL) and the video stream being captured, the F value at the start of a field may not reflect the actual field being supplied. The FINV bit in VC*x*CTL allows the detected field value to be inverted. (For example, in BT.656 525/60 operation, the F bit changes to 0 to indicate field 1 on the fourth line of the field. If the VRST bit is set so the line counter begins counting at line 1 of the field (the first EAV where V is 1), then the F bit still indicates field 2 (F = 1) and needs to be inverted. If the VRST bit is set to start counting lines beginning with the first active line (the first EAV where V is 0), the F value will have already changed to indicate field 1 (F = 0) and no inversion is necessary.)

The field indicator method uses the FID input directly to determine the current field. This is useful for Y/C data streams that do not have embedded EAV and SAV codes. The FID input is sampled at the start of each field. If FID = 0, then field 1 is starting; if FID = 1, then field 2 is starting. The start of each field is defined by the VRST bit in VC*x*CTL and is either the start or end of vertical blanking as determined by the VBLNK input. The FINV bit may be used in this method in systems where the FID input has the opposite polarity or where the field identification change lags the start of the field.

The field detect method uses HYSNC and VSYNC based field detect logic. This is used for BT.656 or Y/C systems that provide only HSYNC and VSYNC. The field detect logic samples the state of the HSYNC input on the VSYNC active edge. If HSYNC is active on the active VSYNC edge, then field 1 is detected; if HSYNC is inactive on the active VSYNC edge, then field 2 is detected. Because of slight timing variations, the VSYNC transition may not coincide exactly with the HSYNC transition. The detection logic should implement a ±64 clock detection window around HSYNC. If both HSYNC and VSYNC leading edges occur within 64 cycles of each other, then field 1 is detected; otherwise, field 2 is assumed. This is shown in Figure 3–11 for active-low sync signals.

Figure 3–11. Field 1 Detection Timing



3.4.5 Short and Long Field Detect

The short and long field detect logic is used to notify the DSP when a captured field shorter or longer than expected. Detection is enabled by the SFDE and LFDE bits in VC*x*CTL. The SFD and LFD bits in VPIS indicate when a short or long field occurred and trigger an interrupt to the DSP if enabled.

If a vertical blanking period is detected before the end of the capture field, a short field is detected . If EAV is used for vertical sync (EXC = 0), then a short field is detected when an EAV with V = 1 occurs on or before VCOUNT = VCYSTOP*n*. If the VCTL1 input is used for vertical sync (EXC = 1), then a short field is detected if a VCTL1 active edge occurs before VCOUNT = (VCYSTOP*n*).

If a vertical blanking period occurs more than 1 line past the end of the capture field, a long field is detected. A long field is detected when VCOUNT = VCYSTOP*n* + 1. (A long field is only detected when the VRST bit in VC*x*CTL is cleared to 0; when VRST = 1, a long field is always detected.) Long field detection cannot be used if the capture window is a vertical subset of the field that crops lines at the bottom. Such a window would always result in a long field detection. If VCTL1 is used for vertical sync, then the VCTL1 signal must represent VBLNK (vertical blank) for proper long field detect. If

VCTL1 is a VSYNC (vertical sync) input, then a long field is always detected. (Even if VCYSTOP*n* is set to the last active line, VCOUNT usually increments past VCYSTOP*n* + 1 while it counts the vertical front porch lines that occur prior to VSYNC active.)

3.5 Video Input Filtering

The video input filter performs simple hardware scaling and resampling on incoming 8-bit BT.656 or 8-bit Y/C data. Filtering hardware is always disabled during 10-bit or raw data capture modes. For proper filter operation, the channel's EXC bit in VC*x*CTL must be cleared to 0 (embedded timing reference codes used) and the CAPEN input must not go inactive during the active video window.

3.5.1 Input Filter Modes

The input filter has four modes of operation: no-filtering, ½ scaling, chrominance resampling, and ½ scaling with chrominance resampling. Filter operation is determined by the CMODE, SCALE, and RESMPL bits of VC*x*CTL.

Table 3–10 shows the input filter mode selection. When 8-bit BT.656 or Y/C capture operation is selected (CMODE = x00), scaling is selected by setting the SCALE bit and chrominance resampling is selected by setting the RESMPL bit. If 8-bit BT.656 or Y/C capture is not selected (CMODE ≠ x00), filtering is disabled.

Table 3–10. Input Filter Mode Selection

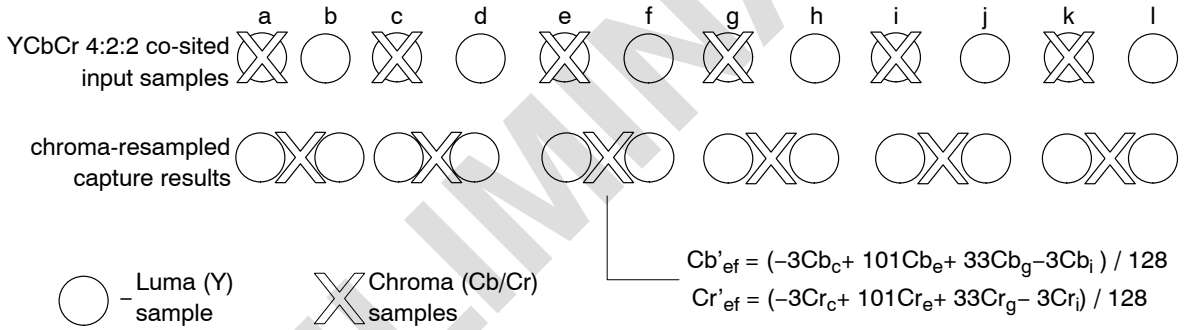
VC*x*CTL Bit			
CMODE	**RESMPL**	**SCALE**	**Filter Operation**
x00	0	0	No filtering
x00	0	1	½ scaling
x00	1	0	Chrominance resampling (full scale)
x00	1	1	½ scaling with chrominance resampling
x01	x	x	No filtering
x10	x	x	No filtering
x11	x	x	No filtering

3.5.2 Chrominance Resampling Operation

Chrominance resampling computes chrominance values at sample points midway between the input luminance samples based on the input co-sited chrominance samples. This filter performs the horizontal portion of a conversion between YCbCr 4:2:2 format and YCbCr 4:2:0 format. The vertical portion of the conversion must be performed in software.

The chrominance resampling filters calculate the implied value of Cb and Cr in between luminance sample points based upon nearby co-sited Cb and Cr samples. The resulting values are clamped to between 01h and FEh and sent to the Cb and Cr capture buffers. Chrominance resampling is shown in Figure 3−12.

Figure 3−12. Chrominance Resampling



3.5.3 Scaling Operation

The ½-scaling mode is used to reduce the horizontal resolution of captured luminance and chrominance data by a factor of two. For applications that require only CIF or lower resolutions, this reduces the video capture buffer memory requirements (and the bandwidth needed to write the buffer) by a factor of two. Vertical scaling must be performed in software. (The bandwidth to load in the buffer is again reduced by 50% over the nonhorizontal scaled case.)

The filtering for the luminance portion of the scaling filter changes depending on if chrominance resampling is also enabled. (By changing the luminance filter, the chrominance filters can remain the same.) The resulting values are clamped to between 01h and FEh and sent to the Y, Cb, and Cr capture buffers. Scaling for co-sited capture is shown in Figure 3−13 and scaling for chrominance resampling is shown in Figure 3−14.

Figure 3−13. 1/2 Scaled Co-Sited Filtering

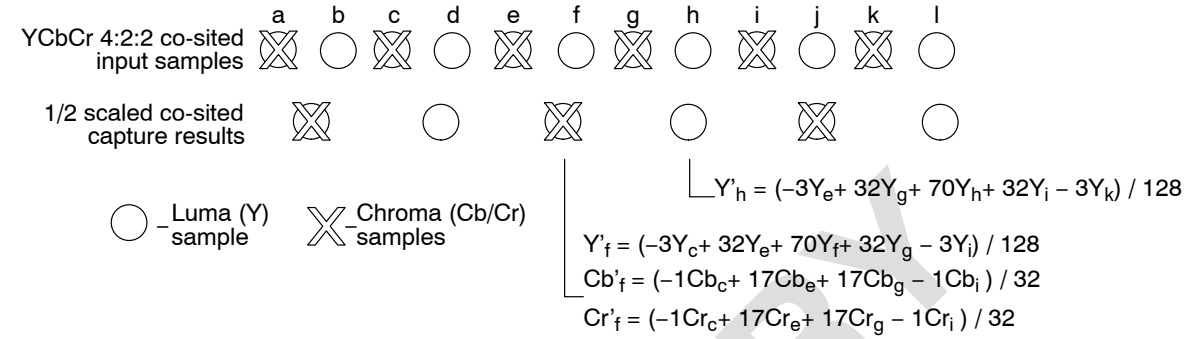
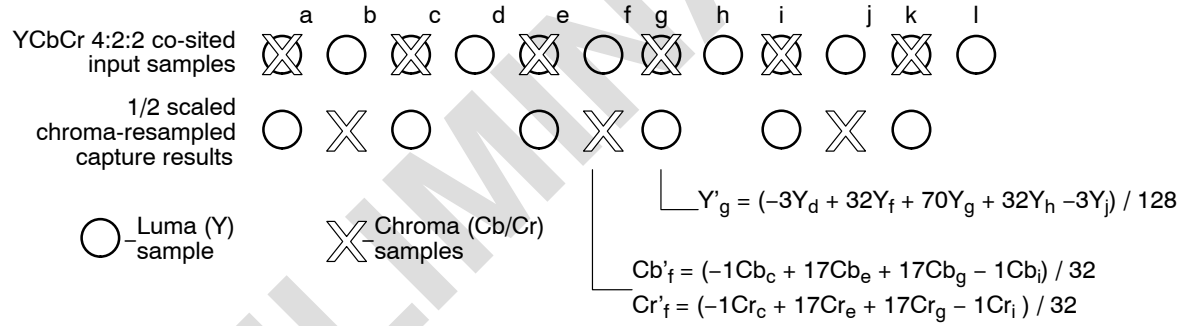


Figure 3−14. 1/2 Scaled Chrominance Resampled Filtering



Note that because input scaling is limited to ½, true CIF horizontal resolution is not achieved if the full BT.656 horizontal line (720 pixels) is captured. A CIF size line can be captured by selecting a 704 pixel-sized window within the BT.656 line. This window size and location on the line are programmed using the VCXSTART*n* and VCXSTOP*n* bits.

Note that when ½ scaling is selected, horizontal timing applies to the incoming data (before scaling). The VCTHRLD value applies to the data written into the FIFO after scaling.

3.5.4 Edge Pixel Replication

Because the filters make use of preceding and trailing samples, filtering artifacts can occur at the beginning of the BT.656 or Y/C active line because no samples exist before the SAV code, and at the end of the BT.656 active line because no samples exist after the EAV code. In order to minimize artifacts, the first *m* samples after sample 0 (where *m* is the maximum number of preceding samples used by any of the filters) are mirrored to the left of sample 0 and the last *m* samples before the last sample are mirrored to the right of the last sample.

Figure 3–15 shows edge pixel replication assuming an *m* value of 3. Sample a is the first sample after the SAV code. Therefore, samples b–d are mirrored to the left of sample a to provide values for the filter calculations on the first few pixels in the line. Likewise, samples n – 1 to n – 3 are mirrored to the right of the last sample n to provide values for the last few pixels on the line.

Note that edge pixel replication only comes into effect when the full BT.656 stream is being captured. If VCXSTART is greater than 0, then only some of the leading edge replicated pixels are used by the filter. If VCXSTART is greater than *m,* then none of the leading edge replicated pixels are used. Similarly, if VCXSTOP is less than the number of samples before EAV, then none or only some of the trailing edge replicated pixels are used by the filters.

Figure 3–15. Edge Pixel Replication

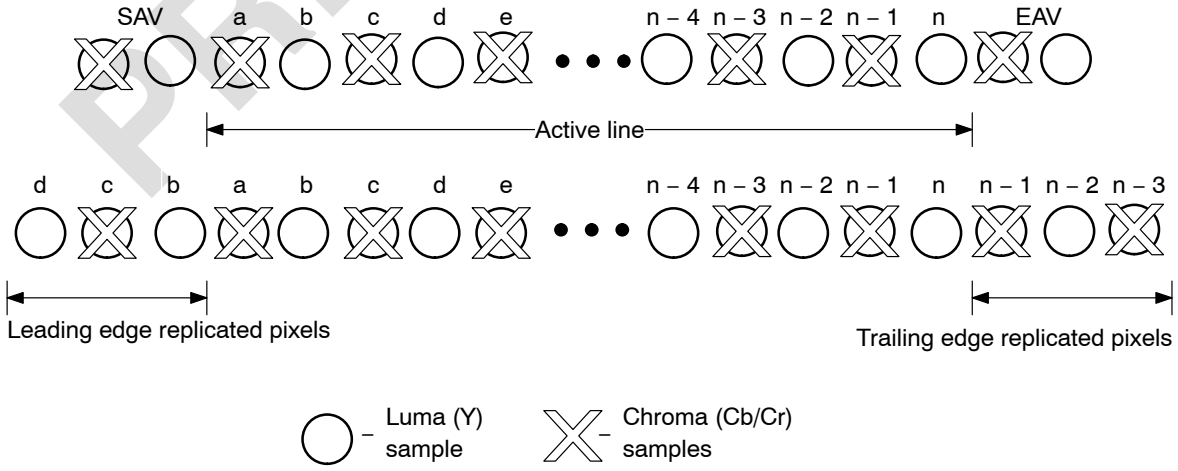
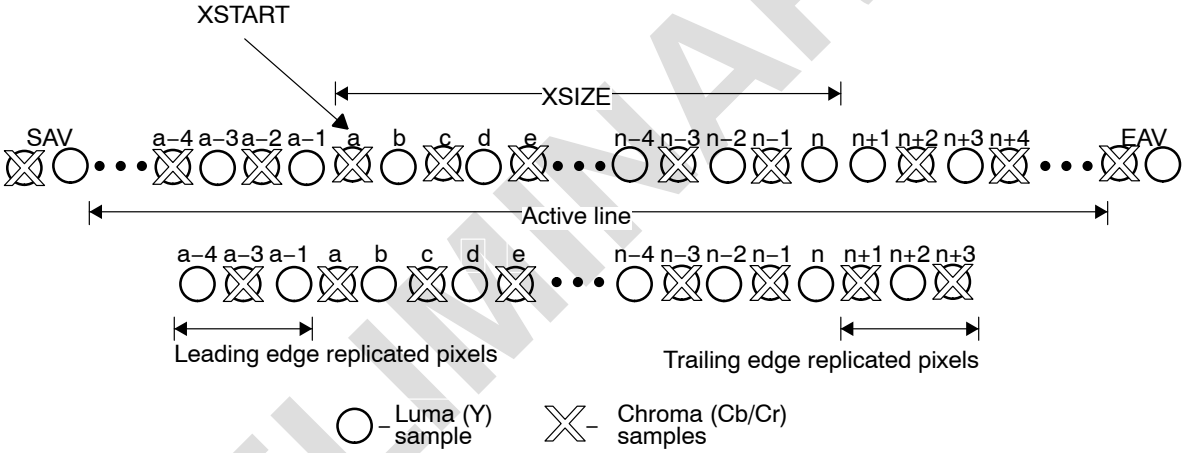


Figure 3−16 shows an example of a capture window that is smaller than the BT.656 active line. Sample a is the first sample in the horizontal capture window and sample n is the last sample. In this case, any filtering done on the first sample location uses the *m* leading edge captured pixels (*m* is 3 in this example), and any filtering done on the last sample location uses the *m* trailing captured pixels. (From an implementation standpoint, the mirroring and filtering can still begin and end with SAV and EAV, but the samples before VCXSTART or after VCXSTOP must not be saved to the YCbCr buffers.)

Figure 3−16. Capture Window Not Requiring Edge Pixel Replication



3.6 Ancillary Data Capture

The BT.656 and some Y/C specifications includes provision for carrying ancillary (nonvideo) data within the horizontal and vertical blanking regions. Horizontal ancillary (HANC) data appears between the EAV code and SAV codes. Vertical ancillary (VANC) data, also called vertical blanking interval (VBI) data, appears during the active horizontal line portion of vertically blanking (for example, after an SAV with V = 1).

3.6.1 Horizontal Ancillary (HANC) Data Capture

No special provisions are made for the capture of HANC data. HANC data may be captured using the normal video capture mechanism by programming VCXSTRT to occur before the SAV (when HCOUNT is reset by the EAV code) or by programming VCXSTOP to occur past the EAV code (when HCOUNT is reset by the SAV code). Note that the EAV code and any subsequent HANC data will still be YCbCr separated. Software must parse the Y, Cb, and Cr memory buffers to determine any HANC data presence and to reconstruct the HANC data. The VCTHRLD value and DMA size must be programmed to comprehend the additional samples. You must disable scaling and chroma re-sampling when including the capture of HANC data to prevent data corruption.

3.6.2 Vertical Ancillary (VANC) Data Capture

VANC (or VBI) data is commonly used for such features as teletext and closed-captioning. No special provisions are made for the capture of VBI data. VBI data may be captured using the normal capture mechanism by programming VCYSTART to occur before the first line of active video on the first line of desired VBI data. (VCOUNT must be reset by an EAV with V = 1). Note that the VBI data will be YCbCr separated. Software must parse the Y, Cb, and Cr memory buffers to determine any VBI data presence and to reconstruct the VBI data. You must disable scaling and chroma resampling when the capture of VBI data is desired or the data will be corrupted by the filters.

3.7 Raw Data Capture Mode

This mode/feature of the video port is currently not supported at this time.

In the raw data capture mode, the data is sampled by the interface only when the CAPEN signal is active. Data is captured at the rate of the sender's clock, without any interpretation or start/stop of capture based on the data values.

To ensure initial capture synchronization to the beginning of a frame, an optional setup synchronization enable (SSE) bit is provided in VC*x*STRT1. If the SSE bit is set, then when the VCEN bit is set to 1, the video port will not start capturing data until after detecting two vertical blanking intervals. If the SSE bit is cleared to 0, capture begins immediately when the VCEN bit is set.

The incoming digital video capture data is stored in the FIFO, which is 2560-bytes (in dual-channel operation) or 5120-bytes deep (in single-channel operation). The memory-mapped location YSRC*x* is associated with the Y buffer. The YSRC*x* location is a read-only register and is used to access video data samples stored in the buffer.

The captured data set size is set by VC*x*STOP*n*. The VCXSTOP and VCYSTOP bits set the 24-bits of data set size (VCXSTOP sets the lower 12 bits and VCYSTOP sets the upper 12 bits). Capture is complete and the appropriate F1C, F2C, or FRMC bit is set when the captured data size reaches the combined VCYSTOP and VCXSTOP value.

The video port generates a YEVT after the specified number of new samples has been captured in the buffer. The number of samples required to generate YEVTx is programmable and is set in the VCTHRLD*n* bits of VC*x*THRLD. On every YEVT, the DMA should move data from the buffer to the DSP memory. When moving data from the buffer to the DSP memory, the DMA should use the YSRC*x* location as a source address.

3.7.1 Raw Data Capture Notification

Raw data mode captures a single data packet of information using only CAPEN for control. Field information is available only for channel A operation using the FID input on VCTL2. If the RDFE bit in VCACTL is set, then the video port samples the FID input at the start of each data block (when DCOUNT = 0 and CAPENA is active) to determine the current field. In this case, the CON, FRAME, CF1, and CF2 bits in VC*x*CTL are used in a manner identical to BT.656 mode (see section 3.4.1).

For channel B operation or when the RDFE bit in VCACTL is not set, no field information is available. Some flexibility in capture and DSP notification is still provided in order to accommodate various DMA structures and processing flows. Each raw data packet is treated similar to a progressive scan video frame. The raw data mode uses the CON and FRAME bits of VC*x*CTL in a slightly different manner, as listed in Table 3−11.

Table 3−11. Raw Data Mode Capture Operation

VC*x*CTL Bit				
CON	**FRAME**	**CF2**	**CF1**	**Operation**
0	0	x	x	Noncontinuous frame capture. FRMC is set after data block capture and causes CCMPx to be set. Capture will halt upon completion of the next frame unless the FRMC bit is cleared. (DSP has the entire next frame time to clear FRMC.)
0	1	x	x	Single frame capture. FRMC is set after data block capture and causes CCMPx to be set. Capture is halted until the FRMC bit is cleared.
1	0	x	x	Continuous frame capture. FRMC is set after data block capture and causes CCMPx to be set (CCMPx interrupt can be disabled). The port will continue capturing frames regardless of the state of FRMC.
1	1	x	x	Reserved

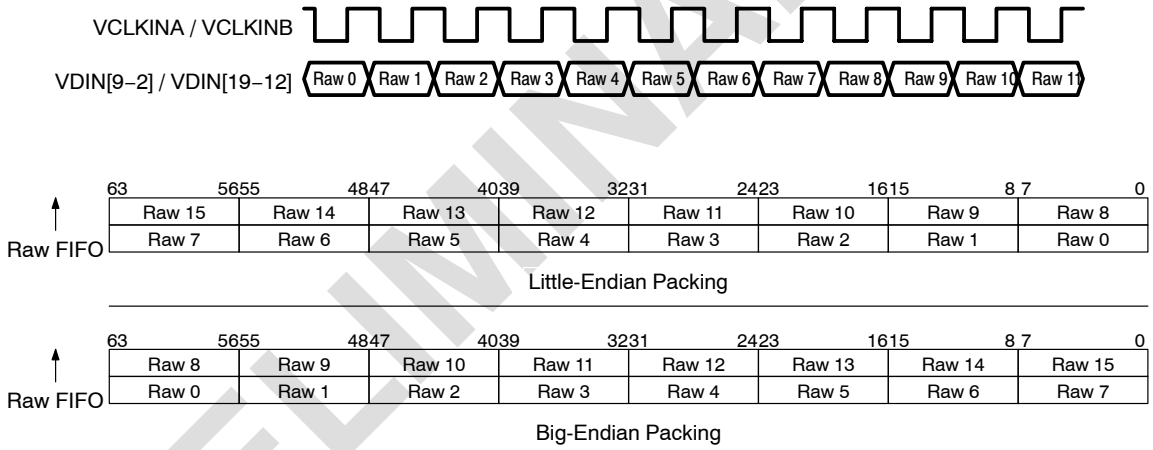
The CON bit controls the capture of multiple frames. When CON = 1, continuous capture is enabled, the video port captures incoming frames (assuming the VCEN bit is set) without the need for DSP interaction. It relies on a DMA structure with circular buffering capability to service the capture FIFO. When CON = 0, continuous capture is disabled, the video port sets the frame capture complete bit (FRMC) in VC*x*STAT upon the capture of each frame. Once the capture complete bit is set, at most, one more frame can be received before capture operation is halted (as determined by the FRAME bit state). This prevents subsequent data from overwriting previous frames until the DSP has a chance to update DMA pointers or process those fields.

3.7.2 Raw Data FIFO Packing

Captured data is always packed into 64-bits before being written into the capture FIFO(s). The packing and byte ordering is dependant upon the capture data size and the device endian mode. For little-endian operation (default), data is packed into the FIFO from right to left; for big-endian operation, data is packed from left to right.

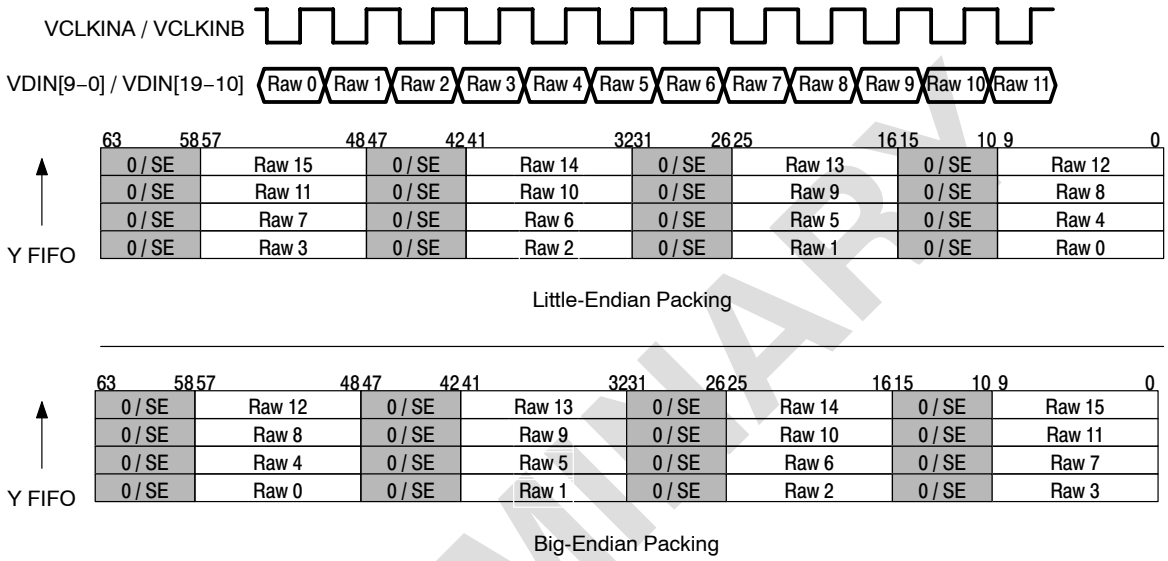
The 8-bit raw-data mode stores all data in a single FIFO. Four samples are packed into each word as shown in Figure 3–17.

Figure 3–17. 8-Bit Raw Data FIFO Packing



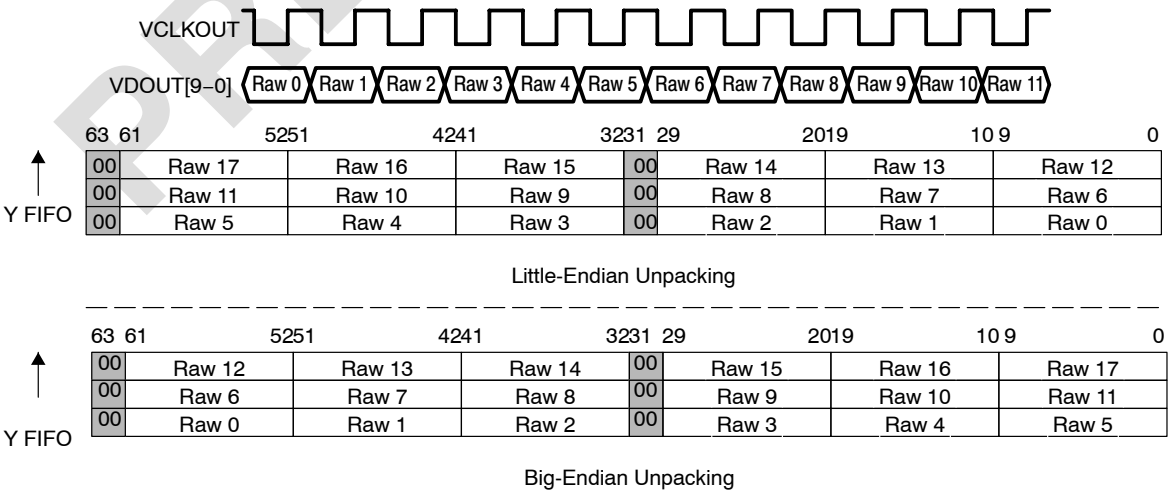
The 10-bit raw data mode stores all data into a single FIFO. Two samples are packed into each word with zero or sign extension as shown in Figure 3–18.

Figure 3–18. 10-Bit Raw Data FIFO Packing



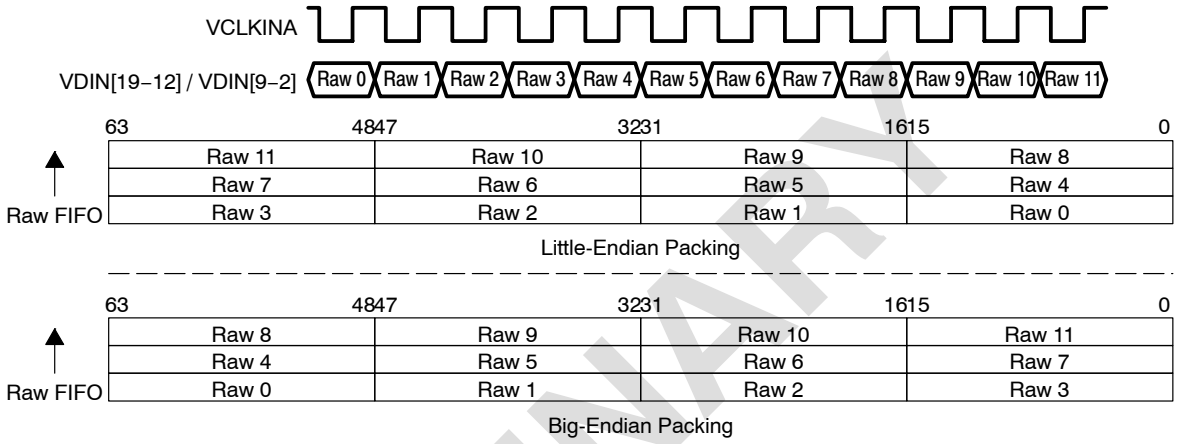
The 10-bit dense raw data mode stores all data into a single FIFO. Three samples are packed into each word with zero extension as shown in Figure 3–19.

Figure 3–19. 10-Bit Dense Raw Data FIFO Packing



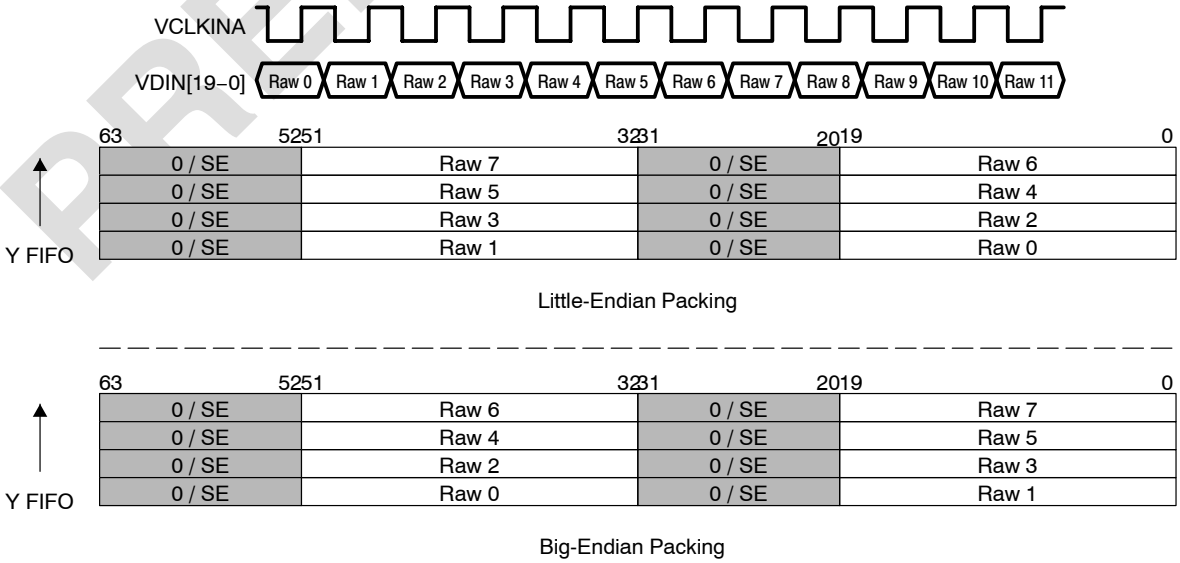
The 16-bit raw data mode stores all data into a single FIFO. Two samples are packed into each word as shown in Figure 3–20.

Figure 3–20. 16-Bit Raw Data FIFO Packing



The 20-bit raw data mode stores all data into a single FIFO. One sample is placed right justified in each word and zero or sign extended as shown in Figure 3–21.

Figure 3–21. 20-Bit Raw Data FIFO Packing



3.8 TSI Capture Mode

This mode/feature of the video port is currerntly not supported at this time.

The transport stream interface (TSI) capture mode captures MPEG-2 transport data.

3.8.1 TSI Capture Features

The video port TSI capture mode supports the following features:

- Supports SYNC detect using the PACSTRT input from a front-end device.
- Data capture at the rising edge of incoming VCLK0.
- Parallel data reception.
- Maximum data rate of 30 Mbytes/second.
- Programmable packet size.
- Hardware counter mechanism to timestamp incoming packet data.
- Programmable filtering of packets with errors.
- Interrupt to the DSP, based on absolute system time or system time clock cycles.

The video port does not perform following functions; these functions should be performed in software:

- PID filtering
- Data parsing
- De-scrambling of data

3.8.2 TSI Data Capture

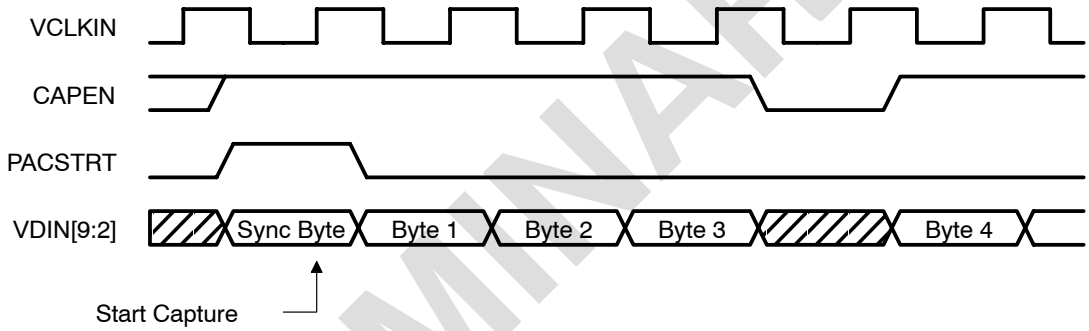
Eight-bit parallel data is received on the input data bus. Data is captured on the rising edge of VCLKIN. The data consists typically of 188-byte packets, with the first byte a SYNC byte (also called a preamble). The capture packet length is determined by the value of VCASTOP.

Data on the data bus is considered valid and captured only when the CAPEN signal is active. TSI data capture begins with a SYNC byte as indicated by PACSTRT (and CAPEN) active. (The SYNC byte may have any value.) Data is captured on each VCLK rising edge when CAPEN is active until the entire

packet has been captured, irrespective of additional PACSTRT transitions. The end-of-packet condition occurs when the 24-bit capture byte counter (as reflected by the VCYPOS and VCXPOS bits of VCASTAT) equals the value in the VCYSTOP and VCXSTOP bits of VCASTOP. The captured data includes both SYNC byte and the data payload as shown in Figure 3−22.

After a packet is captured, the video port waits for the next active PACSTRT to begin capture of another packet. Received packet data is packed into 64 bits before being written to the FIFO.

Figure 3−22. Parallel TSI Capture



3.8.3 TSI Capture Error Detection

The video port checks for two types of errors during TSI capture. The first is a packet error on the incoming packet as indicated by an active PACERR signal. If PACERR is active during any of the first eight bytes of a packet and error packet filtering is enabled (ERRFILT bit in TSICTL is set), then the video port will ignore (not capture) the incoming data until the next PACSTRT is received. If error packet filtering is not enabled or if PACERR becomes active sometime after the first eight bytes of the packet, the entire packet is captured and the PERR bit is set in the timestamp inserted at the end of the packet.

The second error detected is an early PACSTRT error. This occurs when an active PACSTRT is detected before an entire packet (as determined by the packet size programmed in VCASTOP) has been captured. The port will continue to capture the expected packet size but will set the PSTERR bit in the timestamp inserted at the end of the packet. After capture completion, the port will wait for a subsequent PACSTRT before beginning capture of another packet.

3.8.4 Synchronizing the System Clock

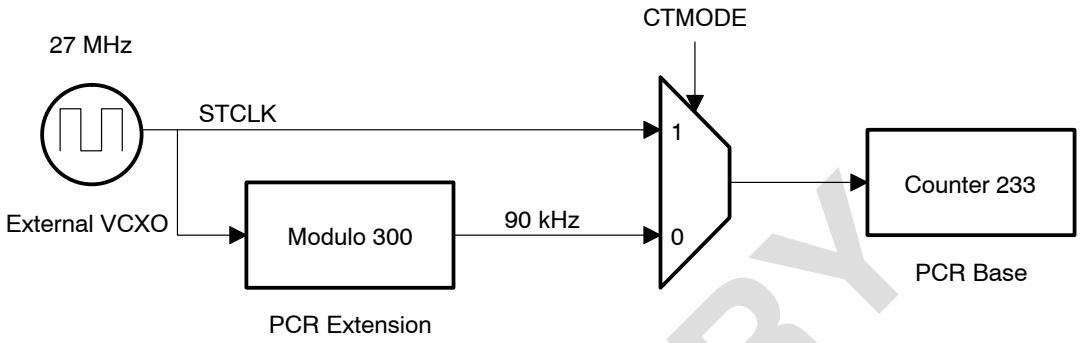
Synchronization is an important aspect of decoding and presenting data in real-time digital data delivery systems. This is addressed in MPEG-2 transport packets by transmitting timing information in the adaptation fields of selected data packets. This value serves as a reference for timing comparison in the receiving system. The program clock reference (PCR) header, shown in Figure 3−23, is a 48-bit field (six bits are reserved). A 42-bit value is transmitted within the 48-bit stream and consists of a 33-bit PCR field that represents a 90-kHz clock sample and a 9-bit PCR extension field that represents a 27-MHz clock sample. The PCR indicates the expected time at the completion of reading the field from the bit stream at the transport decoder. The transport data packets are in-sync with the encoder time clock.

Figure 3−23. Program Clock Reference (PCR) Header Format

The video port, in conjunction with the VCXO interpolated control (VIC), allows a combined hardware and software solution to synchronize the local system time clock (STC) with the encoder time clock reference transmitted in the bit stream.

The video port maintains a hardware counter that counts the system time. The counter is driven by a system time clock (STCLK) input driven by an external VCXO. The counter is split into two fields: a 33-bit field (PCR base) that counts at 90 kHz and a 9-bit field (PCR extension) that counts at 27 MHz. The 9-bit counter counts from 0 to 299 at 27 MHz. Each time the 9-bit counter rolls over to 0, the 33-bit counter is incremented by 1. This is equivalent to the PCR time-stamp transmitted in the bit-stream. The 33-bit field can also be programmed to count at 27 MHz for compatibility with the MPEG-1 32-bit PCR, by setting the CTMODE bit in VCCTL to 1; in which case, the PCR extension portion of the counter is not used. Figure 3−24 shows the system time clock counter operation.

Figure 3–24. System Time Clock Counter Operation



On reception of a packet (during the sync byte), a snapshot of the counter is captured. This snapshot, or timestamp, is inserted in the receiving FIFO at the end of each data packet. Software uses this timestamp, to determine the deviation of the local system time clock from the encoder time clock. Any time a packet with a PCR header is received, the timestamp for that packet is compared with the PCR value by software. A PLL is implemented in software to synchronize the STCLK with the encoder time clock value in the PCR. This algorithm then drives the VIC, which drives the VDAC output to the external VCXO, which supplies STCLK.

The system time clock counter is initialized by software with the PCR of the first packet with a PCR header. After initialization, the counter can be reinitialized by software upon detecting a discontinuity in subsequent packet PCR header values.

The system time is made available to the DSP at any time through the system time clock registers (TSISTCLKL and TSISTCLKM). The DSP can program the video port to interrupt the DSP whenever a specific system time is reached or whenever a specific number of system time clock cycles have elapsed.

3.8.5 TSI Data Capture Notification

Since TSI mode captures only data packets, there is no need for field control. Some flexibility in capture and DSP notification is still provided in order to accommodate various DMA structures and processing flows. Each TSI data packet is treated similar to a progressive scan video frame. The TSI mode uses the CON and FRAME bits of VCACTL in a slightly different manner, as listed in Table 3−12.

The CON bit controls the capture of multiple packets. When CON = 1, continuous capture is enabled, the video port captures incoming data packets (assuming the VCEN bit is set) without the need for DSP interaction. It relies on a DMA structure with circular buffering capability to service the capture FIFO. When CON = 0, continuous capture is disabled, the video port sets the frame capture complete bit (FRMC) in VCASTAT upon the capture of each packet. Once the capture complete bit is set, at most, one more frame can be received before capture operation is halted (as determined by the FRAME bit state). This prevents subsequent data from overwriting previous packets until the DSP has a chance to update DMA pointers or process those packets.

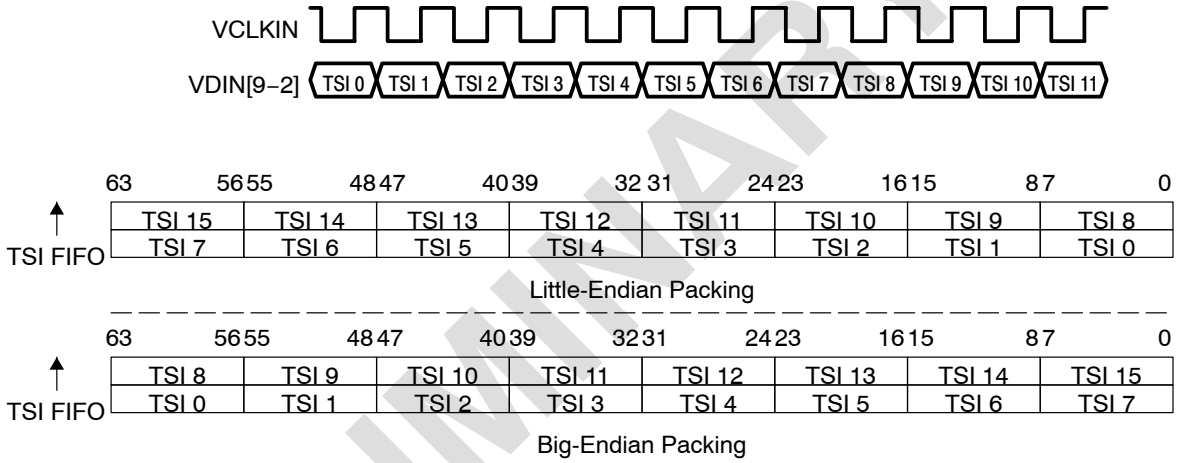
Table 3−12. TSI Capture Mode Operation

VCACTL Bit				
CON	**FRAME**	**CF2**	**CF1**	**Operation**
0	0	x	x	Noncontinuous packet capture. FRMC is set after packet capture and causes CCMPA to be set. Capture will halt upon completion of the next data packet unless the FRMC bit is cleared. (DSP has the entire next data packet time to clear FRMC.)
0	1	x	x	Single packet capture. FRMC is set after packet capture and causes CCMPA to be set. Capture is halted until the FRMC bit is cleared.
1	0	x	x	Continuous packet capture. FRMC is set after packet capture and causes CCMPA to be set (CCMPx interrupt can be disabled). The port will continue capturing packets regardless of the state of FRMC.
1	1	x	x	Reserved

3.8.6 Writing to the FIFO

The captured TSI packet data and the associated timestamps are written into the receive FIFO. The packet data is written first, followed by the timestamp. The FIFO controller controls both data writes and timestamp writes into the FIFO. The FIFO data packing is shown in Figure 3–25.

Figure 3–25. TSI FIFO Packing



The data capture circuitry signals to the synchronizing circuit when to take a timestamp of the hardware counters. The FIFO write controller keeps track of number of bytes received in a packet. It multiplexes the timestamp data and the packet data onto the FIFO write data bus. The timestamp and packet error information are inserted after each packet in the FIFO and must use the correct endian byte ordering. The format for the timestamp is shown in Figure 3–26 and Figure 3–27.

Figure 3–26. TSI Timestamp Format (Little Endian)

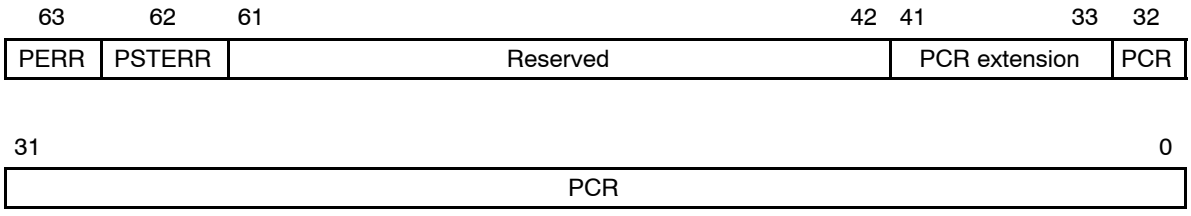
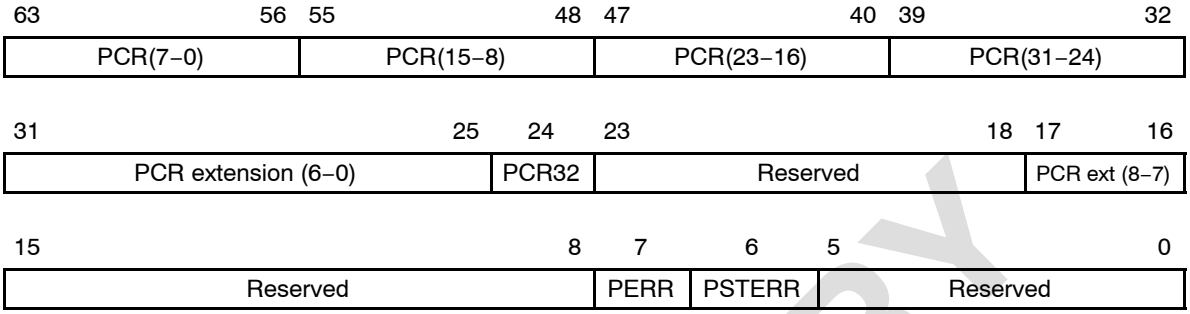


Figure 3–27. TSI Timestamp Format (Big Endian)



3.8.7 Reading from the FIFO

The YSRCA location is associated with the TSI capture buffer. The YSRCA location is a read-only pseudo-register and is used to access the TSI data samples stored in the buffer.

The captured data packet size is set by VCASTOP. The VCXSTOP and VCYSTOP bits set the 24-bits of TSI packet size (VCXSTOP sets the lower 12 bits and VCYSTOP sets the upper 12 bits). Capture is complete and the FRMC bit is set when the data counter equals the combined VCYSTOP and VCXSTOP value.

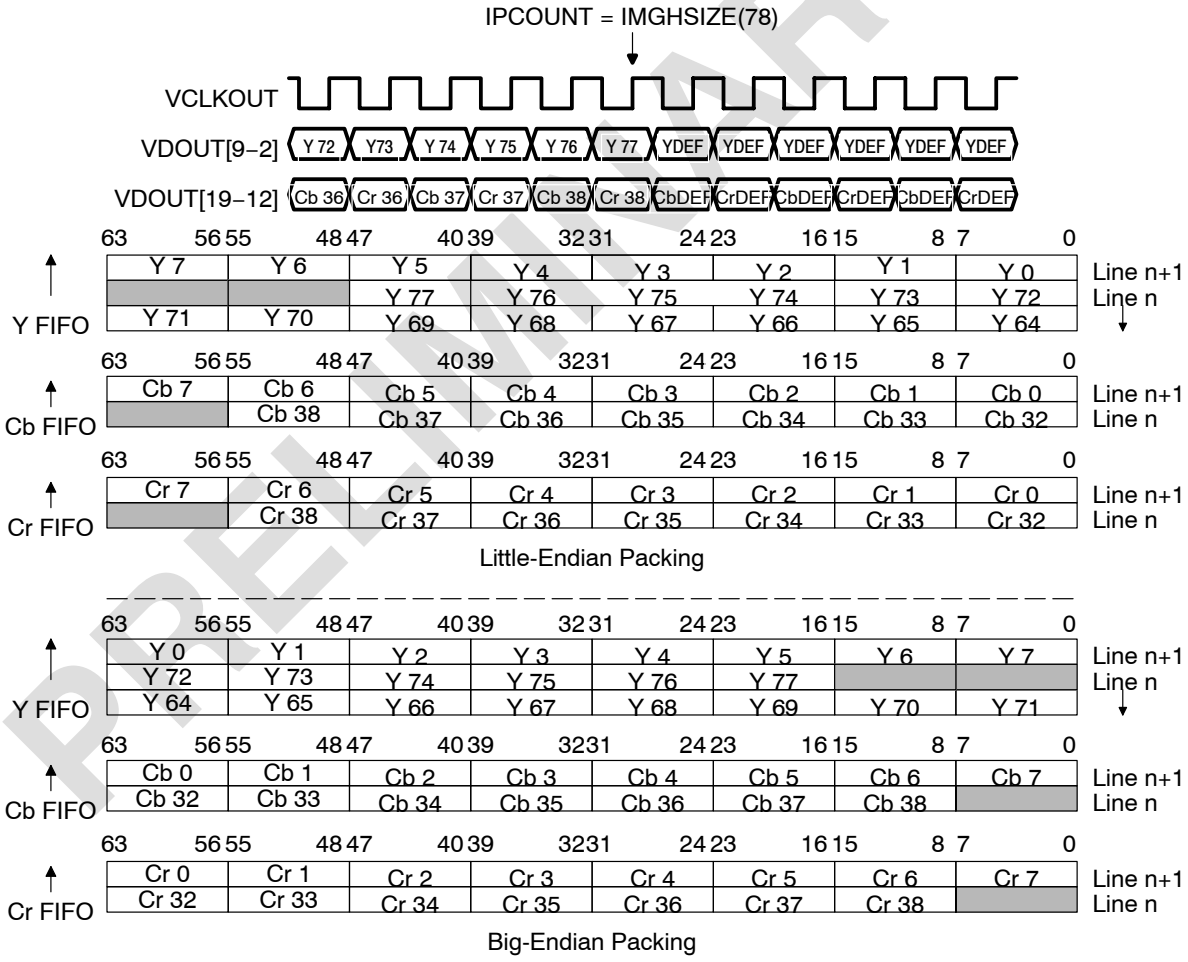
The video port generates a YEVT after the specified number of new samples has been captured in the buffer. The number of samples required to generate YEVT is programmable and is set in the VCTHRLD1 bits of VCATHRLD. VCTHRLD1 should be set to the packet size plus 8 bytes of timestamp. On every YEVT, the DMA should move data from the buffer to the DSP memory. When moving data from the buffer to the DSP memory, the DMA should use the memory address of the YSRCA location as a source address.

3.9 Capture Line Boundary Conditions

In order to simplify DMA transfers, FIFO doublewords must not contain data from more than one capture line. This means that a FIFO write must be performed whenever 8 bytes have been received or when the line complete condition (HCOUNT = VCXSTOP) occurs. Thus, every captured line begins on a doubleword boundary and non-doubleword length lines are padded at the end. An example is shown in Figure 3–28.

In Figure 3–28 (8-bit Y/C mode), the line length is not a doubleword. When the condition HCOUNT = VCXSTOP occurs, the FIFO location is written even though 8 bytes have not been received. The next capture line then begins in the next FIFO location at byte 0. This operation extends to all capture modes. In the case of TSI and raw data modes, there are no lines. In these modes, a final write at the end of the packet must be performed when the packet data count equals the 24-bit combined value of VCXCOUNT and VCYCOUNT.

Figure 3–28. Capture Line Boundary Example



3.10 Capturing Video in BT.656 or Y/C Mode

This mode/feature of the video port is currently not supported at this time.

In order to capture video in the BT.656 or Y/C format, the following steps are needed:

1) Set the last pixel to be captured in VC*x*STOP1 and VC*x*STOP2 (set the VCXSTOP and VCYSTOP bits).

2) Set the first pixel to be captured in VC*x*STRT1 and VC*x*STRT2 (set the VCXSTART and VCYSTART bits).

3) Write to VC*x*THRLD to set the capture threshold. Every time the number of received pixels reaches the number specified by the VCTHRLD1 bits, a YEVT*x*, CbEVT*x*, and CrEVT*x* are generated by the video capture module. The VCTHRLD1 bits value must be an even number.

4) Configure a DMA channel to move data from YSRC*x* to a destination in the DSP memory. The channel transfers should be triggered by the YEVT*x*. The size of the transfers should be set to VCTHRLD1/4 for 8-bit mode, VCTHRLD1/2 for 10-bit mode, or VCTHRLD1/3 for dense 10-bit mode. (This is because 4, 2, or 3 pixels are packed per FIFO word and the DMA is moving 32-bit words from YSRC*x* to the memory). The DMA must start on a doubleword boundary and move an even number of words.

5) Configure a DMA channel to move data from CBSRC*x* to a destination in the DSP memory. The channel transfers should be triggered by the CbEVT*x*. The size of the transfers should be set to VCTHRLD1/8 for 8-bit mode, VCTHRLD1/4 for 10-bit mode, or VCTHRLD1/6 for dense 10-bit mode. (This is because 4, 2, or 3 pixels are packed per FIFO word, the DMA is moving 32-bit words from CBSRC*x* to the memory, and there are half the number of pixels in the Cb FIFO as in the Y FIFO.) The DMA must start on a doubleword boundary and move an even number of words.

6) Configure a DMA channel to move data from CRSRC*x* to a destination in the DSP memory. The channel transfers should be triggered by the CrEVT*x*. The size of the transfers should be set to VCTHRLD1/8 for 8-bit mode, VCTHRLD1/4 for 10-bit mode, or VCTHRLD1/6 for dense 10-bit mode. (This is because 4, 2, or 3 pixels are packed per FIFO word, the

DMA is moving 32-bit words from CRSRC*x* to the memory, and there are half the number of pixels in the Cr FIFO as in the Y FIFO.) The DMA must start on a double-word boundary and move an even number of words.

7) Write to the video port interrupt enable register (VPIE) to enable overrun (COVR*x*) and capture complete (CCMP*x*) interrupts, if desired.

8) Write to VC*x*CTL to:
   - ❏ Set capture mode (CMODE = 00x for BT.656 input, 10x for Y/C input).
   - ❏ Set desired field/frame operation (CON, FRAME, CF2, CF1 bits).
   - ❏ Set sync and field ID control (VRST, HRST, FDD, FINV, VCTL0 bits).
   - ❏ Set 10-bit pack mode (10BPK bits), if 10-bit operation is selected.
   - ❏ Enable scaling (SCALE and RESMPL bits), if desired and using 8-bit data.
   - ❏ Set VCEN bit to enable capture.

9) Capture is enabled at the start of the first frame after VCEN = 1 and begins at the start of the first selected field. DMA events are generated as triggered by VC*x*THRLD1. When a selected field has been captured (VCXPOS = VCXSTOP and VCYPOS = VCYSTOP), the F1C, F2C, or FRMC bits in VC*x*STAT are set and cause the CCMPx bit in VPIS to be set. This generates a DSP interrupt, if the CCMPx bit is enabled in VPIE.

10) If continuous capture is enabled, the video port begins capturing again at the start of the next selected field or frame. If noncontinuous field 1 and field 2 or frame capture is enabled, the next field or frame is captured, during which the DSP must clear the appropriate completion status bit or further capture is disabled. If single frame capture is enabled, capture is disabled until the DSP clears the FRMC bit.

3.10.1 Handling FIFO Overrun in BT.656 or Y/C Mode

In case of a FIFO overrun, the COVR*x* bit is set in VPIS. This condition initiates an interrupt to the DSP, if the overrun interrupt is enabled (setting the COVR bit in VPIE enables overrun interrupt).

The overrun interrupt routine should set the BLKCAP bit in VC*x*CTL and it should reconfigure DMA channel settings. The DMA channel must be reconfigured for capture of the next frame since the current frame transfer failed. Setting the BLKCAP bit flushes the capture FIFO and blocks DMA events for the channel. As long as the BLKCAP bit is set, the video capture channel ignores the incoming data with exception of SAV and EAV codes but the internal counters continue counting.

The BLKCAP bit should be cleared to 0 in order to continue capture. Clearing the BLKCAP bit takes effect in the subsequent video field (DMA events are still going to be blocked in the video field in which the BLKCAP bit is cleared.)

3.11 Capturing Video in Raw Data Mode

This mode/feature of the video port is currently not supported at this time.

In order to capture video in the raw data mode, the following steps are needed:

1) Set VC*x*STOP1 to specify size of an image to be captured (VCXSTOP sets the lower 12 bits and VCYSTOP sets the upper 12 bits of the captured image size in pixels).

2) Write to VC*x*THRLD to set the capture threshold. Every time the number of received pixels reaches the number specified by the VCTHRLD1 bits, a YEVT*x* is generated by the video capture module.

3) Configure a DMA channel to move data from YSRC*x* to a destination in the DSP memory. The channel transfers should be triggered by the YEVT*x*. The size of the transfers should be set to VCTHRLD1/4 for 8-bit mode, VCTHRLD1/3 for dense 10-bit mode, VCTHRLD1/2 for 10-bit or 16-bit mode, or VCTHRLD1 for 20-bit mode. The DMA must start on a doubleword boundary and move an even number of words.

4) Write to the video port interrupt enable register (VPIE) to enable overrun (COVR*x*) and capture complete (CCMP*x*) interrupts, if desired.

5) Write to VC*x*CTL to:

   - ❏ Set capture mode (CMODE = x1x for raw data mode).
   - ❏ Choose capture operation (CON, FRAME bits).
   - ❏ Set 10-bit pack mode (10BPK bits), if 10-bit operation is selected.
   - ❏ Enable raw data sync (RDS), if desired.
   - ❏ Set VCEN bit to enable capture.

6) Capture starts when the ICAPEN signal is asserted and VCEN = 1. Data is captured on every VCLKINx rising edge when CAPENx is active. DMA events (YEVTx) are generated as triggered by VCxTHRLD1. When a complete data block has been captured (DCOUNT = VCYSTOP and VCXSTOP combined value), the FRMC bit in VC*x*STAT is set causing the CCMP*x* bit in VPIS to be set. This generates a DSP interrupt, if CCMP*x* is enabled in VPIE.

7) If continuous capture is enabled, the video port begins capturing again on the next VCLKIN rising edge when CAPEN is valid. If noncontinuous capture is enabled, the next data block is captured during which the DSP must clear the FRMC bit or further capture is disabled. If single frame capture is enabled, capture is disabled until the DSP clears the FRMC bit (at which point, raw data sync must again be performed if enabled).

3.11.1 Handling FIFO Overrun Condition in Raw Data Mode

In case of a FIFO overrun, the COVR*x* bit is set in VPIS. This condition initiates an interrupt to the DSP, if the overrun interrupt is enabled (setting the COVR*x* bit in VPIE enables overrun interrupt).

The overrun interrupt routine should set the BLKCAP bit in VC*x*CTL and it should reconfigure DMA channel settings. The DMA channel must be reconfigured for capture of the next frame since the current frame transfer failed. Setting the BLKCAP bit flushes the capture FIFO and blocks DMA events for the channel. As long as the BLKCAP bit is set, the video capture channel ignores the incoming data but the internal data counter continues counting.

The BLKCAP bit should be cleared to 0 in order to continue capture. Clearing the BLKCAP bit takes effect in the subsequent frame after a raw data sync period is detected on CAPEN*x*. (DMA events are still going to be blocked in the frame in which the BLKCAP bit is cleared.)

3.12 Capturing Data in TSI Capture Mode

This mode/feature of the video port is currently not supported at this time.

In order to capture data in TSI capture mode, the following steps are needed:

1) Set VCASTOP1 to specify size of a data packet to be captured (VCXSTOP sets the lower 12 bits and VCYSTOP sets the upper 12 bits of the data packet).

2) Write to VC*x*THRLD to set the capture threshold to the data packet size. Every time the number of received bytes reaches the number specified by the VCTHRLD1 bits, a YEVT*x* is generated by the video capture module.

3) Configure a DMA channel to move data from YSRCA to a destination in the DSP memory. The channel transfers should be triggered by the VIDEVTA. The size of the transfers should be set to the data packet size + 8 bytes of timestamp information. The DMA must start on a double-word boundary and move an even number of words.

4) Write to TSICTL to:

   - ❑ Set TSI capture mode (TCMODE = 0 for parallel data, 1 for serial data)
   - ❑ Select counter mode (TCMODE)
   - ❑ Enable error packet filtering (ERRFILT) if desired

5) In Sigma-Delta peripheral:

   - ❑ Write to the SDCTL register to set the precision for the Sigma Delta module.
   - ❑ Write to the SDDIV register to set the divider value Sigma Delta interpolation frequency.

6) Write to TSISTCMPL, TSISTCMPM, TSISTMSKL, and TSISTMSKM if needed to initiate an interrupt, based on STC absolute time.

7) Write to TSITICKS if an interrupt is desired every x cycles of STC.

8) Write to VPCTL to select TSI capture operation (TSI = 1).

9) Write to VPIE to enable overrun (COVRA) and capture complete (CCMPA) interrupts, if desired.

10) Write to VCACTL to set capture mode (CMODE = 010).

11) Set VCEN bit in VCACTL to enable capture.

12) Capture begins on the first VCLKINA rising edge when CAPENA and PACSTRT are valid. A DMA event is generated as triggered by VCATHRLD1. When the entire packet has been captured (DCOUNT = VCYSTOP and VCXSTOP combined value), the FRMC bit in VCASTAT is set causing the CCMP*x* bit in VPIS to be set. This generates a DSP interrupt, if CCMP*x* is enabled in VPIE.

13) If continuous capture is enabled, the video port begins capturing again on the next VCLKIN rising edge when CAPEN and PACSTRT are valid. If noncontinuous capture is enabled, the next data packet is captured during which the DSP must clear the FRMC bit or further capture is disabled. If single frame capture is enabled, capture is disabled until the DSP clears the FRMC bit.

3.12.1 Handling FIFO Overrun Condition in TSI Capture Mode

In case of a FIFO overrun, the COVR*x* bit is set in VPIS. This condition initiates an interrupt to the DSP, if the overrun interrupt is enabled (setting the COVR*x* bit in VPIE enables overrun interrupt).

The overrun interrupt routine should set the BLKCAP bit in VC*x*CTL and it should reconfigure DMA channel settings. The DMA channel must be reconfigured for capture of the next frame since the current frame transfer failed. Setting the BLKCAP bit flushes the capture FIFO and blocks DMA events for the channel. As long as the BLKCAP bit is set, the video capture channel ignores the incoming data but the internal data counter continues counting.

The BLKCAP bit should be cleared to 0 in order to continue capture. Clearing the BLKCAP bit takes effect on the next PACSTRT. (DMA events are still going to be blocked in the TSI packet in which the BLKCAP bit is cleared.)

3.13 Video Capture Registers

The registers for controlling the video capture mode of operation are listed in Table 3–13. See the device-specific datasheet for the memory address of these registers.

Table 3–13. Video Capture Control Registers

Offset Address†	Acronym	Register Name	Section
100h	VCASTAT	Video Capture Channel A Status Register	3.13.1
104h	VCACTL	Video Capture Channel A Control Register	3.13.2
108h	VCASTRT1	Video Capture Channel A Field 1 Start Register	3.13.3
10Ch	VCASTOP1	Video Capture Channel A Field 1 Stop Register	3.13.4
110h	VCASTRT2	Video Capture Channel A Field 2 Start Register	3.13.5
114h	VCASTOP2	Video Capture Channel A Field 2 Stop Register	3.13.6
118h	VCAVINT	Video Capture Channel A Vertical Interrupt Register	3.13.7
11Ch	VCATHRLD	Video Capture Channel A Threshold Register	3.13.8
120h	VCAEVTCT	Video Capture Channel A Event Count Register	3.13.9
140h	VCBSTAT	Video Capture Channel B Status Register	3.13.1
144h	VCBCTL	Video Capture Channel B Control Register	3.13.10
148h	VCBSTRT1	Video Capture Channel B Field 1 Start Register	3.13.3
14Ch	VCBSTOP1	Video Capture Channel B Field 1 Stop Register	3.13.4
150h	VCBSTRT2	Video Capture Channel B Field 2 Start Register	3.13.5
154h	VCBSTOP2	Video Capture Channel B Field 2 Stop Register	3.13.6
158h	VCBVINT	Video Capture Channel B Vertical Interrupt Register	3.13.7
15Ch	VCBTHRLD	Video Capture Channel B Threshold Register	3.13.8
160h	VCBEVTCT	Video Capture Channel B Event Count Register	3.13.9
180h	TSICTL	TSI Capture Control Register	3.13.11
184h	TSICLKINITL	TSI Clock Initialization LSB Register	3.13.12
188h	TSICLKINITM	TSI Clock Initialization MSB Register	3.13.13

† The absolute address of the registers is device/port specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

Table 3−13. Video Capture Control Registers (Continued)

Offset Address†	Acronym	Register Name	Section
18Ch	TSISTCLKL	TSI System Time Clock LSB Register	3.13.14
190h	TSISTCLKM	TSI System Time Clock MSB Register	3.13.15
194h	TSISTCMPL	TSI System Time Clock Compare LSB Register	3.13.16
198h	TSISTCMPM	TSI System Time Clock Compare MSB Register	3.13.17
19Ch	TSISTMSKL	TSI System Time Clock Compare Mask LSB Register	3.13.18
1A0h	TSISTMSKM	TSI System Time Clock Compare Mask MSB Register	3.13.19
1A4h	TSITICKS	TSI System Time Clock Ticks Interrupt Register	3.13.20

† The absolute address of the registers is device/port specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

3.13.1 Video Capture Channel *x* Status Register (VCASTAT, VCBSTAT)

The video capture channel *x* status register (VCASTAT, VCBSTAT) indicates the current status of the video capture channel. The VC*x*STAT is shown in Figure 3−29 and described in Table 3−14.

In BT.656 capture mode, the VCXPOS and VCYPOS bits indicate the HCOUNT and VCOUNT values, respectively, to track the coordinates of the most recently received pixel. The F1C, F2C, and FRMC bits indicate completion of fields or frames and may need to be cleared by the DSP for capture to continue, depending on the selected frame capture operation (see section 3.4.1).

In raw data and TSI modes, the VCXPOS and VCYPOS bits reflect the lower and upper 12 bits, respectively, of the 24-bit data counter that tracks the number of received data samples. The FRMC bit indicates when an entire data packet has been received and may need to be cleared by the DSP for capture to continue, depending on the selected frame operation (see section 3.7.1 and section 3.8.5).

Figure 3−29. Video Capture Channel x Status Register (VCASTAT, VCBSTAT)

31	30	29	28	27	16
FSYNC	FRMC	F2C	F1C	VCYPOS	
R-0	R/WC-0	R/WC-0	R/WC-0	R-0	

15	13	12	11	0
Reserved		VCFLD	VCXPOS	
R-0		R-0	R-0	

Legend: R = Read only; WC = Write 1 to clear, write of 0 has no effect; *-n* = value after reset

Table 3−14. Video Capture Channel x Status Register (VCxSTAT) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31	FSYNC			Current frame sync bit.		
		CLEARD	0	VCOUNT = VINT1 or VINT2, as selected by the FSCL2 bit in VC*x*VINT.	Not used.	Not used.
		SET	1	VCOUNT = 1 in field 1.	Not used.	Not used.
30	FRMC			Frame (data) captured bit. Write 1 to clear the bit, a write of 0 has no effect.		
		NONE	0	Complete frame has not been captured.	Complete data block has not been captured.	Entire data packet has not been captured.
		CAPTURED CLEAR	1	Complete frame has been captured.	Complete data block has been captured.	Entire data packet has been captured.
29	F2C			Field 2 captured bit. Write 1 to clear the bit, a write of 0 has no effect.		
		NONE	0	Field 2 has not been captured.	Not used.	Not used.
		CAPTURED CLEAR	1	Field 2 has been captured.	Not used.	Not used.

† For CSL implementation, use the notation VP_VC*x*STAT_*field_symval*

Table 3−14. Video Capture Channel x Status Register (VCxSTAT) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
28	F1C			Field 1 captured bit. Write 1 to clear the bit, a write of 0 has no effect.		
		NONE	0	Field 1 has not been captured.	Not used.	Not used.
		CAPTURED CLEAR	1	Field 1 has been captured.	Not used.	Not used.
27−16	VCYPOS	OF(*value*)	0−FFFh	Current VCOUNT value and the line that is currently being received (within the current field).	Upper 12 bits of the data counter.	Upper 12 bits of the data counter.
15−13	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
12	VCFLD			VCFLD bit indicates which field is currently being captured. The VCFLD bit is updated based on the field detection logic selected by the FLDD bit in VCACTL.		
		NONE	0	Field 1 is active.	Not used.	Not used.
		DETECTED	1	Field 2 is active.	Not used.	Not used.
11−0	VCXPOS	OF(*value*)	0−FFFh	Current HCOUNT value. The pixel index of the last received pixel.	Lower 12 bits of the data counter.	Lower 12 bits of the data counter.

† For CSL implementation, use the notation VP_VC*x*STAT_*field_symval*

3.13.2 Video Capture Channel A Control Register (VCACTL)

Video capture is controlled by the video capture channel A control register (VCACTL) shown in Figure 3−30 and described in Table 3−15.

Figure 3−30. Video Capture Channel A Control Register (VCACTL)

31	30	29	24
RSTCH	BLKCAP	Reserved	
R/WS-0	R/W-1	R-0	

23	22	21	20	19	18	17	16
Reserved		RDFE	FINV	EXC	FLDD	VRST	HRST
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0

15	14	13	12	11	10	9	8
VCEN	PK10B		LFDE	SFDE	RESMPL	Reserved	SCALE
R/W-0	R/W-0		R/W-0	R/W-0	R/W-0	R-0	R/W-0

7	6	5	4	3	2	0
CON	FRAME	CF2	CF1	Reserved	CMODE	
R/W-0	R/W-0	R/W-1	R/W-1	R-0	R/W-0	

Legend: R = Read only; R/W = Read/Write; WS = Write 1 to reset, write of 0 has no effect; *-n* = value after reset

Table 3−15. Video Capture Channel A Control Register (VCACTL) Field Descriptions

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
31	RSTCH			Reset channel bit. Write 1 to reset the bit, a write of 0 has no effect.		
		NONE	0	No effect.		
		RESET	1	Resets the channel by blocking further DMA event generation and flushing the FIFO upon completion of any pending DMAs. Also clears the VCEN bit. All channel registers are set to their initial values. RSTCH is autocleared after channel reset is complete.		

† For CSL implementation, use the notation VP_VCACTL_*field_symval*
‡ For complete encoding of these bits, see Table 3−6, Table 3−11, and Table 3−12.

Table 3−15. Video Capture Channel A Control Register (VCACTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
30	BLKCAP			Block capture events bit. BLKCAP functions as a capture FIFO reset without affecting the current programmable register values. The F1C, F2C, and FRMC status bits, in VCASTAT, are not updated. Field or frame complete interrupts and vertical interrupts are also not generated. Clearing BLKCAP does not enable DMA events during the field where the bit is cleared. Whenever BLKCAP is set and then cleared, the software needs to clear the field and frame status bits (F1C, F2C, and FRMC) as part of the BLKCAP clear operation.		
		CLEAR	0	Enables DMA events in the video frame that follows the video frame where the bit is cleared. (The capture logic must sync to the start of the next frame after BLKCAP is cleared.)		
		BLOCK	1	Blocks DMA events and flushes the capture channel FIFOs.		
29−22	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
21	RDFE			Field identification enable bit. (Channel A only)		
		DISABLE	0	Not used.	Field identification is disabled.	Not used.
		ENABLE	1	Not used.	Field identification is enabled.	Not used.
20	FINV			Detected field invert bit.		
		FIELD1	0	Detected 0 is field 1.	Not used.	Not used.
		FIELD2	1	Detected 0 is field 2.	Not used.	Not used.
19	EXC			External control select bit. (Channel A only)		
		EAVSAV	0	Use EAV/SAV codes.	Not used.	Not used.
		EXTERN	1	Use external control signals.	Not used.	Not used.

† For CSL implementation, use the notation VP_VCACTL_*field_symval*
‡ For complete encoding of these bits, see Table 3−6, Table 3−11, and Table 3−12.

Table 3−15. Video Capture Channel A Control Register (VCACTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
18	FLDD			Field detect method bit. (Channel A only)		
		EAVFID	0	1st line EAV or FID input.	Not used.	Not used.
		FDL	1	Field detect logic.	Not used.	Not used.
17	VRST			VCOUNT reset method bit.		
		V1EAV	0	Start of vertical blank (1st V = 1 EAV or VCTL1 active edge)	Not used.	Not used.
		V0EAV	1	End of vertical blank (1st V = 0 EAV or VCTL1 inactive edge)	Not used.	Not used.
16	HRST			HCOUNT reset method bit.		
		EAV	0	EAV or VCTL0 active edge.	Not used.	Not used.
		SAV	1	SAV or VCTL0 inactive edge.	Not used.	Not used.
15	VCEN			Video capture enable bit. Other bits in VCACTL (except RSTCH and BLKCAP bits) may only be changed when VCEN = 0.		
		DISABLE	0	Video capture is disabled.		
		ENABLE	1	Video capture is enabled.		
14−13	PK10B			10-bit packing format select bit.		
		ZERO	0	Zero extend	Zero extend	Not used.
		SIGN	1h	Sign extend	Sign extend	Not used.
		DENSEPK	2h	Dense pack (zero extend)	Dense pack (zero extend)	Not used.
		−	3h	Reserved	Reserved	Not used.

† For CSL implementation, use the notation VP_VCACTL_*field_symval*
‡ For complete encoding of these bits, see Table 3−6, Table 3−11, and Table 3−12.

Table 3−15. Video Capture Channel A Control Register (VCACTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
12	LFDE			Long field detect enable bit.		
		DISABLE	0	Long field detect is disabled.	Not used.	Not used.
		ENABLE	1	Long field detect is enabled.	Not used.	Not used.
11	SFDE			Short field detect enable bit.		
		DISABLE	0	Short field detect is disabled.	Not used.	Not used.
		ENABLE	1	Short field detect is enabled.	Not used.	Not used.
10	RESMPL			Chroma resampling enable bit.		
		DISABLE	0	Chroma resampling is disabled.	Not used.	Not used.
		ENABLE	1	Chroma is horizontally resampled from 4:2:2 co-sited to 4:2:0 interspersed before saving to chroma buffers.	Not used.	Not used.
9	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
8	SCALE			Scaling select bit.		
		NONE	0	No scaling	Not used.	Not used.
		HALF	1	½ scaling	Not used.	Not used.
7	CON‡			Continuous capture enable bit.		
		DISABLE	0	Continuous capture is disabled.		
		ENABLE	1	Continuous capture is enabled.		

† For CSL implementation, use the notation VP_VCACTL_*field_symval*
‡ For complete encoding of these bits, see Table 3−6, Table 3−11, and Table 3−12.

Table 3−15. Video Capture Channel A Control Register (VCACTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
6	FRAME‡			Capture frame (data) bit.		
		NONE	0	Do not capture frame.	Do not capture single data block.	Do not capture single packet.
		FRMCAP	1	Capture frame.	Capture single data block.	Capture single packet.
5	CF2‡			Capture field 2 bit.		
		NONE	0	Do not capture field 2.	Do not capture field 2.	Not used.
		FLDCAP	1	Capture field 2.	Capture field 2.	Not used.
4	CF1‡			Capture field 1 bit.		
		NONE	0	Do not capture field 1.	Do not capture field 1.	Not used.
		FLDCAP	1	Capture field 1.	Capture field 1.	Not used.
3	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
2−0	CMODE			Capture mode select bit.		
		BT656B	0	Enables 8-bit BT.656 mode.		Not used.
		BT656D	1h	Enables 10-bit BT.656 mode.		Not used.
		RAWB	2h	Enables 8-bit raw data mode.		8-bit TSI mode.
		RAWD	3h	Enables 10-bit raw data mode.		Not used.
		YCB	4h	Enables 16-bit Y/C mode.		Not used.
		YCD	5h	Enables 20-bit Y/C mode.		Not used.
		RAW16	6h	Enables 16-bit raw mode.		Not used.
		RAW20	7h	Enables 20-bit raw mode.		Not used.

† For CSL implementation, use the notation VP_VCACTL_*field_symval*
‡ For complete encoding of these bits, see Table 3−6, Table 3−11, and Table 3−12.

3.13.3 Video Capture Channel *x* Field 1 Start Register (VCASTRT1, VCBSTRT1)

The captured image is a subset of the incoming image. The video capture channel *x* field 1 start register (VCASTRT1, VCBSTRT1) defines the start of the field 1 captured image. Note that the size is defined relative to incoming data (before scaling). VC*x*STRT1 is shown in Figure 3−31 and described in Table 3−16.

In BT.656 or Y/C modes, the horizontal (pixel) counter is reset (to 0) by the horizontal event (as selected by the HRST bit in VC*x*CTL) and the vertical (line) counter is reset (to 1) by the vertical event (as selected by the VRST bit in VC*x*CTL). Field 1 capture starts when HCOUNT = VCXSTART, VCOUNT = VCYSTART, and field 1 capture is enabled.

In raw capture mode, the VCVBLNKP bits defines the minimum vertical blanking period. If CAPEN stays deasserted longer than VCVBLNKP clocks, then a vertical blanking interval is considered to have occurred. If the SSE bit is set when the capture first begins (the VCEN bit is set in VC*x*CTL), the capture does not start until two intervals are counted. This allows the video port to synchronize its capture to the top of a frame when first started.

In TSI capture mode, the capture starts when the CAPEN signal is asserted, the FRMC bit (in VC*x*STAT) is cleared, and a SYNC byte is detected.

Figure 3−31. Video Capture Channel x Field 1 Start Register (VCASTRT1, VCBSTRT1)

31	28	27	16
Reserved		VCYSTART	
R-0		R/W-0	

15	14	12	11	0
SSE	Reserved		VCXSTART/VCVBLNKP	
R/W-1	R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−16. Video Capture Channel x Field 1 Start Register (VCxSTRT1) Field Descriptions

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
27−16	VCYSTART	OF(*value*)	0−FFFh	Starting line number.	Not used.	Not used.
15	SSE			Startup synchronization enable bit.		
		DISABLE	0	Not used.	Startup synchronization is disabled.	Not used.
		ENABLE	1	Not used.	Startup synchronization is enabled.	Not used.
14−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
11−0	VCXSTART VCVBLNKP	OF(*value*)	0−FFFh	VCXSTART bits define the starting pixel number. Must be an even number (LSB is treated as 0).	VCVBLNKP bits define the minimum CAPEN inactive time to be interpreted as a vertical blanking period.	Not used.

† For CSL implementation, use the notation VP_VC*x*STRT1_*field_symval*

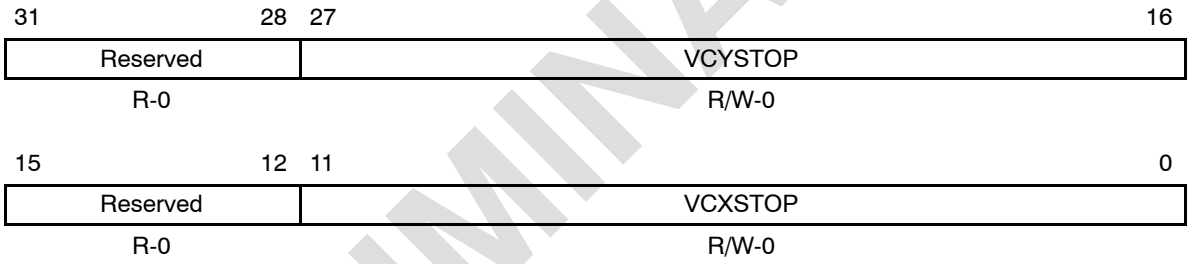
3.13.4 Video Capture Channel *x* Field 1 Stop Register (VCASTOP1, VCBSTOP1)

The video capture channel *x* field 1 stop register (VCASTOP1, VCBSTOP1) defines the end of the field 1-captured image or the end of the raw data or TSI packet. VC*x*STOP1 is shown in Figure 3−32 and described in Table 3−17.

In raw capture mode, the horizontal and vertical counters are combined into a single counter that keeps track of the total number of samples received.

In TSI capture mode, the horizontal and vertical counters are combined into a single data counter that keeps track of the total number of bytes received. The capture starts when a SYNC byte is detected. The data counter counts bytes as they are received. The FRMC bit (in VC*x*STAT) gets set each time a packet has been received.

Figure 3−32. Video Capture Channel x Field 1 Stop Register (VCASTOP1, VCBSTOP1)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−17. Video Capture Channel x Field 1 Stop Register (VCxSTOP1) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31−28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
27−16	VCYSTOP	OF(*value*)	0−FFFh	Last captured line.	Upper 12 bits of the data size (in data samples).	Upper 12 bits of the data size (in data samples).
15−12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
11−0	VCXSTOP	OF(*value*)	0−FFFh	Last captured pixel (VCXSTOP − 1). Must be an even value (the LSB is treated as 0).	Lower 12 bits of the data size (in data samples).	Lower 12 bits of the data size (in data samples).

† For CSL implementation, use the notation VP_VC*x*STOP1_*field_symval*

3.13.5 Video Capture Channel *x* Field 2 Start Register (VCASTRT2, VCBSTRT2)

The captured image is a subset of the incoming image. The video capture channel *x* field 2 start register (VCASTRT2, VCBSTRT2) defines the start of the field 2 captured image. (This allows different window alignment or size for each field.) Note that the size is defined relative to incoming data (before scaling). VC*x*STRT2 is shown in Figure 3−33 and described in Table 3−18.

In BT.656 or Y/C modes, the horizontal (pixel) counter is reset by the horizontal event (as selected by the HRST bit in VC*x*CTL) and the vertical (line) counter is reset by the vertical event (as selected by the VRST bit in VC*x*CTL). Field 2 capture starts when HCOUNT = VCXSTART, VCOUNT = VCYSTART, and field 2 capture is enabled.

These registers are not used in raw data mode or TSI mode because their capture sizes are completely defined by the field 1 start and stop registers.

Figure 3−33. Video Capture Channel x Field 2 Start Register (VCASTRT2, VCBSTRT2)

31	28	27	16
Reserved		VCYSTART	
R-0		R/W-0	

15	12	11	0
Reserved		VCXSTART	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−18. Video Capture Channel x Field 2 Start Register (VCxSTRT2) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
27−16	VCYSTART	OF(*value*)	0−FFFh	Starting line number.	Not used.	Not used.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
11−0	VCXSTART	OF(*value*)	0−FFFh	Starting pixel number. Must be an even number (LSB is treated as 0).	Not used.	Not used.

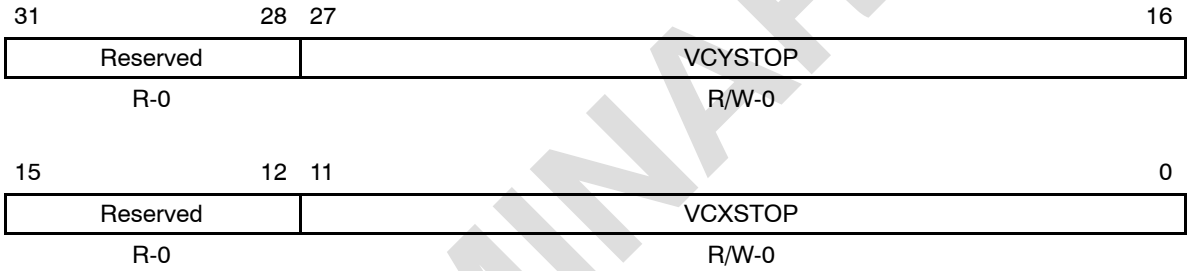
† For CSL implementation, use the notation VP_VC*x*STRT2_*field_symval*

3.13.6 Video Capture Channel *x* Field 2 Stop Register (VCASTOP2, VCBSTOP2)

The video capture channel *x* field 2 stop register (VCASTOP2, VCBSTOP2) defines the end of the field 2-captured image. VC*x*STOP2 is shown in Figure 3−34 and described in Table 3−19.

These registers are not used in raw data mode or TSI mode because their capture sizes are completely defined by the field 1 start and stop registers.

Figure 3−34. Video Capture Channel x Field 2 Stop Register (VCASTOP2, VCBSTOP2)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−19. Video Capture Channel x Field 2 Stop Register (VCxSTOP2) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31−28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
27−16	VCYSTOP	OF(*value*)	0−FFFh	Last captured line.	Not used.	Not used.
15−12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
11−0	VCXSTOP	OF(*value*)	0−FFFh	Last captured pixel (VCXSTOP − 1). Must be an even value (the LSB is treated as 0).	Not used.	Not used.

† For CSL implementation, use the notation VP_VC*x*STOP2_*field_symval*

3.13.7 Video Capture Channel *x* Vertical Interrupt Register (VCAVINT, VCBVINT)

The video capture channel *x* vertical interrupt register (VCAVINT, VCBVINT) controls the generation of vertical interrupts in each field. VCxVINT is shown in Figure 3−35 and described in Table 3−20.

In BT.656 or Y/C mode, an interrupt can be generated upon completion of the specified line in a field (end of line when VCOUNT = VINT*n*). This allows the software to synchronize to the frame or field. The interrupt can be programmed to occur in one or both fields (or not at all) using the VIF1 and VIF2 bits. The VINT*n* bits also determine when the FSYNC bit in VC*x*STAT is cleared. If FSCL2 is 0, then the FSYNC bit is cleared in field 1 when VCOUNT = VINT1; if FSCL2 is 1, then the FSYNC bit is cleared in field 2 when VCOUNT = VINT2.

Figure 3−35. Video Capture Channel x Vertical Interrupt Register (VCAVINT, VCBVINT)

31	30	29	28	27	16
VIF2	FSCL2	Reserved		VINT2	
R/W-0	R/W-0	R-0		R/W-0	

15	14	12	11	0
VIF1	Reserved		VINT1	
R/W-0	R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3–20. Video Capture Channel x Vertical Interrupt Register (VCxVINT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
31	VIF2			Setting of VINT in field 2 enable bit.		
		DISABLE	0	Setting of VINT in field 2 is disabled.	Not used.	Not used.
		ENABLE	1	Setting of VINT in field 2 is enabled.	Not used.	Not used.
30	FSCL2			FSYNC bit cleared in field 2 enable bit.		
		NONE	0	FSYNC bit is not cleared.	Not used.	Not used.
		FIELD2	1	FSYNC bit is cleared in field 2 instead of field 1.	Not used.	Not used.
29–28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
27–16	VINT2	OF(*value*)	0–FFFh	Line that vertical interrupt occurs if VIF2 bit is set.	Not used.	Not used.
15	VIF1			Setting of VINT in field 1 enable bit.		
		DISABLE	0	Setting of VINT in field 1 is disabled.	Not used.	Not used.
		ENABLE	1	Setting of VINT in field 1 is enabled.	Not used.	Not used.
14–12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
11–0	VINT1	OF(*value*)	0–FFFh	Line that vertical interrupt occurs if VIF1 bit is set.	Not used.	Not used.

† For CSL implementation, use the notation VP_VC*x*VINT_*field_symval*

3.13.8 Video Capture Channel *x* Threshold Register (VCATHRLD, VCBTHRLD)

The video capture channel *x* threshold register (VCATHRLD, VCBTHRLD) determines when DMA requests are sent. VC*x*THRLD is shown in Figure 3−36 and described in Table 3−21.

The VCTHRLD1 bits determine when capture DMA events are generated. Once the threshold is reached, generation of further DMA events is disabled until service of the previous event(s) begins (the first FIFO read by the DMA occurs).

In BT.656 and Y/C modes, every two captured pixels represent 2 luma values in the Y FIFO and 2 chroma values (1 each in the Cb and Cr FIFOs). Depending on the data size and packing mode, each value may be a byte (8-bit BT.656 or Y/C), half-word (10-bit BT.656 or Y/C), or subword (dense pack 10-bit BT.656 or Y/C) within the FIFOs. Therefore, the VCTHRLD1 doubleword number represents 8 pixels in 8-bit modes, 4 pixels in 10-bit modes, or 6 pixels in dense pack 10-bit modes. Since the Cb and Cr FIFO thresholds are represented by ½ VCTHRLD1, certain restrictions are placed on what VCTHRLD1 values are valid (see section 2.3.3).

In raw data mode, each data sample may occupy a byte (8-bit raw mode), half-word (10-bit or 16-bit raw mode), subword (dense pack 10-bit raw mode), or word (20-bit raw mode) within the FIFO, depending on the data size and packing mode. Therefore, the VCTHRLD1 doubleword number represents 8 samples, 4, samples, 6 samples, or 2 samples, respectively.

In TSI mode, VCTHRLD1 represents groups of 8 samples with each sample occupying a byte in the FIFO.

The VCTHRLD2 bits behave identically to VCTHRLD1, but are used during field 2 capture. It is only used if the field 2 DMA size needs to be different from the field 1 DMA size for some reason (for example, different captured line lengths in field 1 and field 2). If VT2EN is not set, then the VCTHRLD1 value is used for both fields.

Note that the VCTHRLD*n* applies to data being written into the FIFO. In the case of 8-bit BT.656 or Y/C modes, this means the output of any selected filter.

Figure 3−36. Video Capture Channel x Threshold Register (VCATHRLD, VCBTHRLD)

31	26	25	16
Reserved		VCTHRLD2	
R-0		R/W-0	

15	10	9	0
Reserved		VCTHRLD1	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−21. Video Capture Channel x Threshold Register (VCxTHRLD) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31−26	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
25−16	VCTHRLD2	OF(*value*)	0−3FFh	Number of field 2 doublewords required to generate DMA events.	Not used.	Not used.
15−10	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
9−0	VCTHRLD1	OF(*value*)	0−3FFh	Number of field 1 doublewords required to generate DMA events.	Number of raw data doublewords required to generate a DMA event.	Number of doublewords required to generate a DMA event.

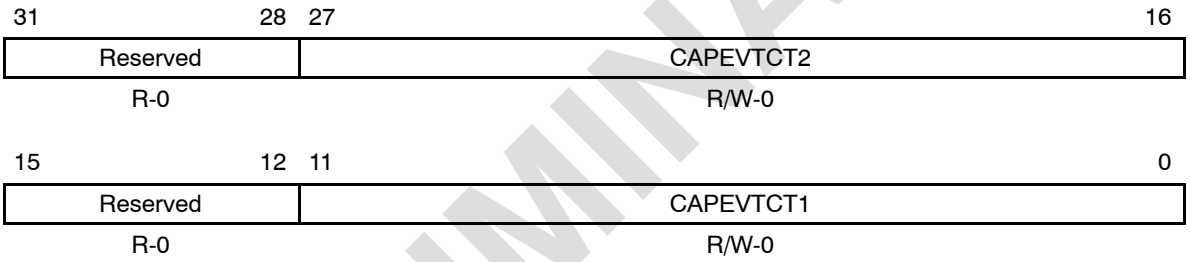
† For CSL implementation, use the notation VP_VC*x*THRLD_VCTHRLD*n_symval*

3.13.9 Video Capture Channel *x* Event Count Register (VCAEVTCT, VCBEVTCT)

The video capture channel *x* event count register (VCAEVTCT, VCBEVTCT) is programmed with the number of DMA events to be generated for each capture field. VC*x*EVTCT is shown in Figure 3−37 and described in Table 3−22.

An event counter tracks how many events have been generated and indicates which threshold value (VCTHRLD1 or VCTHRLD2 in VC*x*THRLD) to use in event generation and in the outgoing data counter. Once the CAPEVTCT*n* number of events have been generated, the DMA logic switches to the other threshold value. See section 2.3.1.

Figure 3−37. Video Capture Channel x Event Count Register (VCAEVTCT, VCBEVTCT)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−22. Video Capture Channel x Event Count Register (VCxEVTCT) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31−28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
27−16	CAPEVTCT2	OF(*value*)	0−FFFh	Number of DMA event sets (YEVT, CbEVT, CrEVT) to be generated for field 2 capture.	Not used.	Not used.
15−12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
11−0	CAPEVTCT1	OF(*value*)	0−FFFh	Number of DMA event sets (YEVT, CbEVT, CrEVT) to be generated for field 1 capture.	Not used.	Not used.

† For CSL implementation, use the notation VP_VC*x*EVTCT_CAPEVTCT*n_symval*

3.13.10 Video Capture Channel B Control Register (VCBCTL)

Video capture is controlled by the video capture channel B control register (VCBCTL) shown in Figure 3−38 and described in Table 3−23.

Figure 3−38. Video Capture Channel B Control Register (VCBCTL)

31	30	29	24
RSTCH	BLKCAP	Reserved	
R/WS-0	R/W-1	R-0	

23	21	20	19	18	17	16
Reserved		FINV	Reserved		VRST	HRST
R-0		R/W-0	R-0		R/W-1	R/W-0

15	14	13	12	11	10	9	8
VCEN	PK10B		LFDE	SFDE	RESMPL	Reserved	SCALE
R/W-0	R/W-0		R/W-0	R/W-0	R/W-0	R-0	R/W-0

7	6	5	4	3	2	1	0
CON	FRAME	CF2	CF1	Reserved		CMODE	
R/W-0	R/W-0	R/W-1	R/W-1	R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; WS = Write 1 to reset, write of 0 has no effect; *-n* = value after reset

Table 3−23. Video Capture Channel B Control Register (VCBCTL) Field Descriptions

				Description		
Bit	***field*†**	***symval*†**	**Value**	**BT.656 or Y/C Mode**	**Raw Data Mode**	**TSI Mode**
31	RSTCH			Reset channel bit. Write 1 to reset the bit, a write of 0 has no effect.		
		NONE	0	No effect.		
		RESET	1	Resets the channel by blocking further DMA event generation and flushing the FIFO upon completion of any pending DMAs. Also clears the VCEN bit. All channel registers are set to their initial values. RSTCH is autocleared after channel reset is complete.		

† For CSL implementation, use the notation VP_VCBCTL_*field_symval*
‡ For complete encoding of these bits, see Table 3−6, Table 3−11, and Table 3−12.

Table 3–23. Video Capture Channel B Control Register (VCBCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
30	BLKCAP			Block capture events bit. BLKCAP functions as a capture FIFO reset without affecting the current programmable register values. The F1C, F2C, and FRMC status bits, in VCBSTAT, are not updated. Field or frame complete interrupts and vertical interrupts are also not generated. Clearing BLKCAP does not enable DMA events during the field where the bit is cleared. Whenever BLKCAP is set and then cleared, the software needs to clear the field and frame status bits (F1C, F2C, and FRMC) as part of the BLKCAP clear operation.		
		CLEAR	0	Enables DMA events in the video frame that follows the video frame where the bit is cleared. (The capture logic must sync to the start of the next frame after BLKCAP is cleared.)		
		BLOCK	1	Blocks DMA events and flushes the capture channel FIFOs.		
29–21	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
20	FINV			Detected field invert bit.		
		FIELD1	0	Detected 0 is field 1.	Not used.	Not used.
		FIELD2	1	Detected 0 is field 2.	Not used.	Not used.
19–18	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
17	VRST			VCOUNT reset method bit.		
		V1EAV	0	Start of vertical blank (1^{st} V = 1 EAV or VCTL1 active edge)	Not used.	Not used.
		V0EAV	1	End of vertical blank (1^{st} V = 0 EAV or VCTL1 inactive edge)	Not used.	Not used.

† For CSL implementation, use the notation VP_VCBCTL_*field_symval*
‡ For complete encoding of these bits, see Table 3–6, Table 3–11, and Table 3–12.

Table 3–23. Video Capture Channel B Control Register (VCBCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
16	HRST			HCOUNT reset method bit.		
		EAV	0	EAV or VCTL0 active edge.	Not used.	Not used.
		SAV	1	SAV or VCTL0 inactive edge.	Not used.	Not used.
15	VCEN			Video capture enable bit. Other bits in VCBCTL (except RSTCH and BLKCAP bits) may only be changed when VCEN = 0.		
		DISABLE	0	Video capture is disabled.		
		ENABLE	1	Video capture is enabled.		
14–13	PK10B			10-bit packing format select bit.		
		ZERO	0	Zero extend	Zero extend	Not used.
		SIGN	1h	Sign extend	Sign extend	Not used.
		DENSEPK	2h	Dense pack (zero extend)	Dense pack (zero extend)	Not used.
		–	3h	Reserved	Reserved	Not used.
12	LFDE			Long field detect enable bit.		
		DISABLE	0	Long field detect is disabled.	Not used.	Not used.
		ENABLE	1	Long field detect is enabled.	Not used.	Not used.
11	SFDE			Short field detect enable bit.		
		DISABLE	0	Short field detect is disabled.	Not used.	Not used.
		ENABLE	1	Short field detect is enabled.	Not used.	Not used.

† For CSL implementation, use the notation VP_VCBCTL_*field_symval*
‡ For complete encoding of these bits, see Table 3–6, Table 3–11, and Table 3–12.

Table 3–23. Video Capture Channel B Control Register (VCBCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
10	RESMPL			Chroma resampling enable bit.		
		DISABLE	0	Chroma resampling is disabled.	Not used.	Not used.
		ENABLE	1	Chroma is horizontally resampled from 4:2:2 co-sited to 4:2:0 interspersed before saving to chroma buffers.	Not used.	Not used.
9	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
8	SCALE			Scaling select bit.		
		NONE	0	No scaling	Not used.	Not used.
		HALF	1	½ scaling	Not used.	Not used.
7	CON‡			Continuous capture enable bit.		
		DISABLE	0	Continuous capture is disabled.		
		ENABLE	1	Continuous capture is enabled.		
6	FRAME‡			Capture frame (data) bit.		
		NONE	0	Do not capture frame.	Do not capture single data block.	Do not capture single packet.
		FRMCAP	1	Capture frame.	Capture single data block.	Capture single packet.
5	CF2‡			Capture field 2 bit.		
		NONE	0	Do not capture field 2.	Not used.	Not used.
		FLDCAP	1	Capture field 2.	Not used.	Not used.

† For CSL implementation, use the notation VP_VCBCTL_*field_symval*
‡ For complete encoding of these bits, see Table 3–6, Table 3–11, and Table 3–12.

Table 3–23. Video Capture Channel B Control Register (VCBCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description		
				BT.656 or Y/C Mode	**Raw Data Mode**	**TSI Mode**
4	CF1‡			Capture field 1 bit.		
		NONE	0	Do not capture field 1.	Not used.	Not used.
		FLDCAP	1	Capture field 1.	Not used.	Not used.
3–2	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.		
1–0	CMODE			Capture mode select bit.		
		BT656B	0	Enables 8-bit BT.656 mode.		Not used.
		BT656D	1h	Enables 10-bit BT.656 mode.		Not used.
		RAWB	2h	Enables 8-bit raw data mode.		Not used.
		RAWD	3h	Enables 10-bit raw data mode.		Not used.

† For CSL implementation, use the notation VP_VCBCTL_*field_symval*
‡ For complete encoding of these bits, see Table 3–6, Table 3–11, and Table 3–12.

3.13.11 TSI Capture Control Register (TSICTL)

The transport stream interface capture control register (TSICTL) controls TSI capture operation. TSICTL is shown in Figure 3–39 and described in Table 3–24.

The ERRFILT, STEN, and TCKEN bits may be written at any time. To ensure stable counter operation, writes to the CTMODE bit are disabled unless the system time counter is halted (ENSTC = 0).

Figure 3–39. TSI Capture Control Register (TSICTL)

31	16
Reserved	
R-0	

15	6	5	4	3	2	1	0
Reserved		ENSTC	TCKEN	STEN	CTMODE	ERRFILT	—
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3–24. TSI Capture Control Register (TSICTL) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656, Y/C Mode, or Raw Data Mode	TSI Mode
31–6	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
5	ENSTC			System time clock enable bit.	
		HALTED	0	Not used.	System time clock input is disabled (to save power). The system time clock counters and tick counter do not increment.
		CLKED	1	Not used.	System time input is enabled. The system time clock counters and tick counters are incremented by STCLK.
4	TCKEN			Tick count interrupt enable bit.	
		DISABLE	0	Not used.	Setting of the TICK bit is disabled.
		SET	1	Not used.	The TICK bit in VPIS is set whenever the tick count is reached.
3	STEN			System time clock interrupt enable bit.	
		DISABLE	0	Not used.	Setting of the STC bit is disabled.
		SET	1	Not used.	A valid STC compare sets the STC bit in VPIS.
2	CTMODE			Counter mode select bit.	
		90KHZ	0	Not used.	The 33-bit PCR portion of the system time counter increments at 90 kHz (when PCRE rolls over from 299 to 0).
		STCLK	1	Not used.	The 33-bit PCR portion of the system time counter increments by the STCLK input.
1	ERRFILT			Error filtering enable bit.	
		ACCEPT	0	Not used.	Packets with errors are received and the PERR bit is set in the timestamp inserted at the end of the packet.
		REJECT	1	Not used.	Packets with errors are filtered out (not received in the FIFO).
0	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	

† For CSL implementation, use the notation VP_TSICTL_*field_symval*

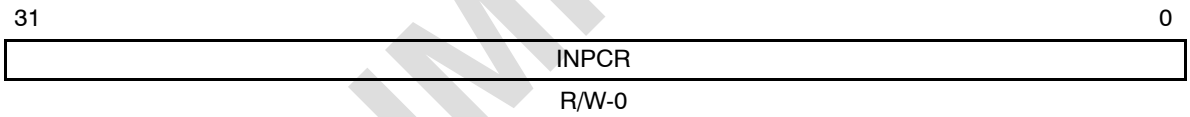
3.13.12 TSI Clock Initialization LSB Register (TSICLKINITL)

The transport stream interface clock initialization LSB register (TSICLKINITL) is used to initialize the hardware counter to synchronize with the system time clock. TSICLKINITL is shown in Figure 3−40 and described in Table 3−25.

On receiving the first packet containing a program clock reference (PCR) and the PCR extension value, the DSP writes the 32 least-significant bits (LSBs) of the PCR into TSICLKINITL. This initializes the counter to the system time clock. TSICLKINITL should also be updated by the DSP whenever a discontinuity in the PCR field is detected.

To ensure synchronization and prevent false compare detection, the software should disable the system time clock interrupt (clear the STEN bit in TSICTL) prior to writing to TSICLKINITL. All bits of the system time counter are initialized whenever either TSICLKINITL or TSICLKINITM are written.

Figure 3−40. TSI Clock Initialization LSB Register (TSICLKINITL)



Legend: R/W = Read/Write; *-n* = value after reset

Table 3−25. TSI Clock Initialization LSB Register (TSICLKINITL) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−0	INPCR	OF(*value*)	0−FFFF FFFFh	Not used.	Initializes the 32 LSBs of the system time clock.

† For CSL implementation, use the notation VP_TSICLKINITL_INPCR_*symval*

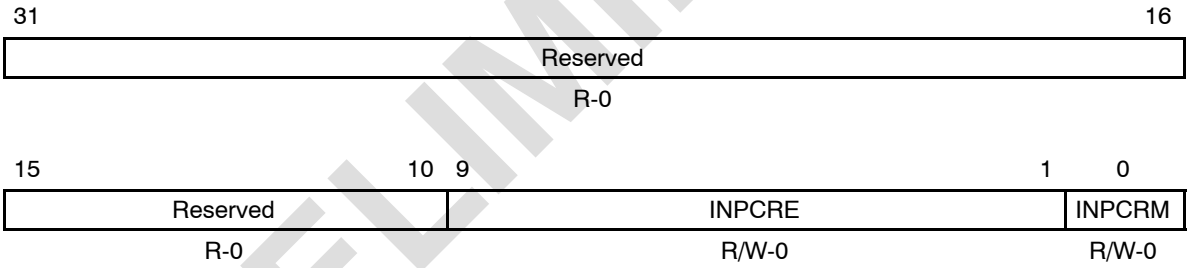
3.13.13 TSI Clock Initialization MSB Register (TSICLKINITM)

The transport stream interface clock initialization MSB register (TSICLKINITM) is used to initialize the hardware counter to synchronize with the system time clock. TSICLKINITM is shown in Figure 3−41 and described in Table 3−26.

On receiving the first packet containing a program clock reference (PCR) header, the DSP writes the most-significant bit (MSB) of the PCR and the 9-bit PCR extension into TSICLKINITM. This initializes the counter to the system time clock. TSICLKINITM should also be updated by the DSP whenever a discontinuity in the PCR field is detected.

To ensure synchronization and prevent false compare detection, the software should disable the system time clock interrupt (clear the STEN bit in TSICTL) prior to writing to TSICLKINITM. All bits of the system time counter are initialized whenever either TSICLKINITL or TSICLKINITM are written.

Figure 3−41. TSI Clock Initialization MSB Register (TSICLKINITM)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−26. TSI Clock Initialization MSB Register (TSICLKINITM) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−10	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
9−1	INPCRE	OF(*value*)	0−1FFh	Not used.	Initializes the extension portion of the system time clock.
0	INPCRM	OF(*value*)	0−1	Not used.	Initializes the MSB of the system time clock.

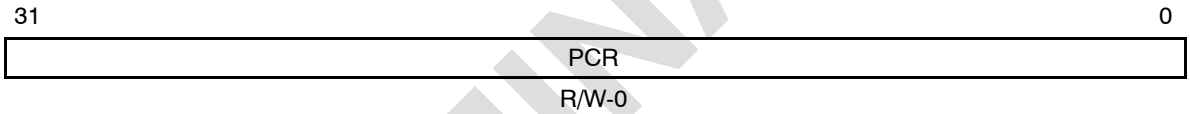
† For CSL implementation, use the notation VP_TSICLKINITM_*field_symval*

3.13.14 TSI System Time Clock LSB Register (TSISTCLKL)

The transport stream interface system time clock LSB register (TSISTCLKL) contains the 32 least-significant bits (LSBs) of the program clock reference (PCR). The system time clock value is obtained by reading TSISTCLKL and TSISTCLKM. TSISTCLKL is shown in Figure 3−42 and described in Table 3−27.

TSISTCLKL represents the current value of the 32 LSBs of the base PCR that normally counts at a 90-kHz rate. Since the system time clock counter continues to count, the DSP may need to read TSISTCLKL twice in a row to ensure an accurate value.

Figure 3−42. TSI System Time Clock LSB Register (TSISTCLKL)



Legend: R/W = Read/Write; -*n* = value after reset

Table 3−27. TSI System Time Clock LSB Register (TSISTCLKL) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−0	PCR	OF(*value*)	0−FFFF FFFFh	Not used.	Contains the 32 LSBs of the program clock reference.

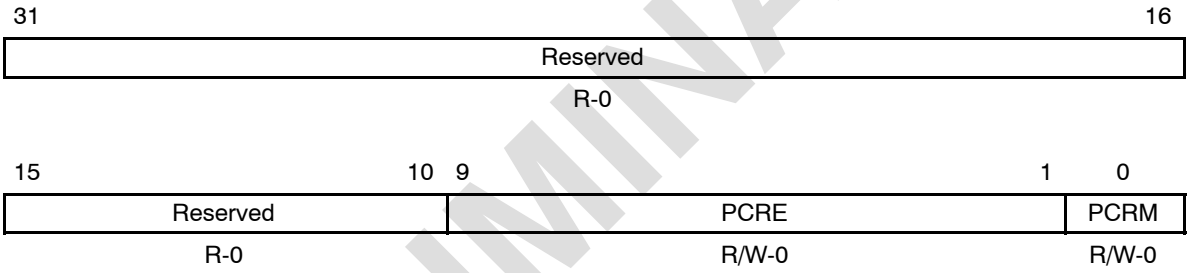
† For CSL implementation, use the notation VP_TSISTCLKL_PCR_*symval*

3.13.15 TSI System Time Clock MSB Register (TSISTCLKM)

The transport stream interface system time clock MSB register (TSISTCLKM) contains the most-significant bit (MSB) of the program clock reference (PCR) and the 9 bits of the PCR extension. The system time clock value is obtained by reading TSISTCLKM and TSISTCLKL. TSISTCLKM is shown in Figure 3−43 and described in Table 3−28.

The PCRE value changes at a 27-MHz rate and is probably not reliably read by the DSP. The PCRM bit normally changes at a 10.5-µHz rate (every 26 hours).

Figure 3−43. TSI System Time Clock MSB Register (TSISTCLKM)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−28. TSI System Time Clock MSB Register (TSISTCLKM) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−10	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
9−1	PCRE	OF(*value*)	0−1FFh	Not used.	Contains the extension portion of the program clock reference.
0	PCRM	OF(*value*)	0−1	Not used.	Contains the MSB of the program clock reference.

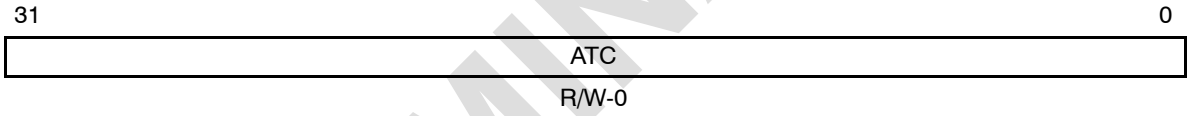
† For CSL implementation, use the notation VP_TSISTCLKM_*field_symval*

3.13.16 TSI System Time Clock Compare LSB Register (TSISTCMPL)

The transport stream interface system time clock compare LSB register (TSISTCMPL) is used to generate an interrupt at some absolute time based on the STC. TSISTCMPL holds the 32 least-significant bits (LSBs) of the absolute time compare (ATC). Whenever the value in TSISTCMPL and TSISTCMPM match the unmasked bits of the time kept by the STC hardware counter and the STEN bit in TSICTL is set, the STC bit in VPIS is set. TSISTCMPL is shown in Figure 3−44 and described in Table 3−29.

To prevent inaccurate comparisons caused by changing register bits, the software should disable the system time clock interrupt (clear the STEN bit in TSICTL) prior to writing to TSISTCMPL.

Figure 3−44. TSI System Time Clock Compare LSB Register (TSISTCMPL)



Legend: R/W = Read/Write; *-n* = value after reset

Table 3−29. TSI System Time Clock Compare LSB Register (TSISTCMPL) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−0	ATC	OF(*value*)	0−FFFF FFFFh	Not used.	Contains the 32 LSBs of the absolute time compare.

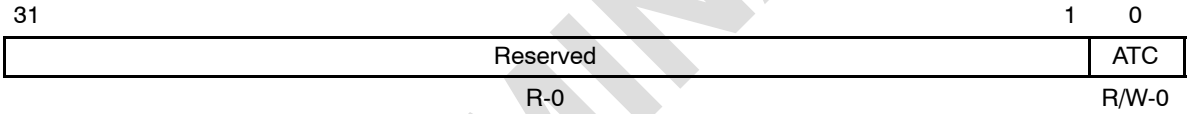
† For CSL implementation, use the notation VP_TSISTCMPL_ATC_*symval*

3.13.17 TSI System Time Clock Compare MSB Register (TSISTCMPM)

The transport stream interface system time clock compare MSB register (TSISTCMPM) is used to generate an interrupt at some absolute time based on the STC. TSISTCMPM holds the most-significant bit (MSB) of the absolute time compare (ATC). Whenever the value in TSISTCMPM and TSISTCMPL match the unmasked bits of the time kept by the STC hardware counter and the STEN bit in TSICTL is set, the STC bit in VPIS is set. TSISTCMPM is shown in Figure 3−45 and described in Table 3−30.

To prevent inaccurate comparisons caused by changing register bits, the software should disable the system time clock interrupt (clear the STEN bit in TSICTL) prior to writing to TSISTCMPM.

Figure 3−45. TSI System Time Clock Compare MSB Register (TSISTCMPM)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−30. TSI System Time Clock Compare MSB Register (TSISTCMPM) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−1	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
0	ATC	OF(*value*)	0−1	Not used.	Contains the MSB of the absolute time compare.

† For CSL implementation, use the notation VP_TSISTCMPM_ATC_*symval*

3.13.18 TSI System Time Clock Compare Mask LSB Register (TSISTMSKL)

The transport stream interface system time clock compare mask LSB register (TSISTMSKL) holds the 32 least-significant bits (LSBs) of the absolute time compare mask (ATCM). This value is used with TSISTMSKM to mask out bits during the comparison of the ATC to the system time clock for absolute time. The bits that are set to one mask the corresponding ATC bits during the compare. TSISTMSKL is shown in Figure 3−46 and described in Table 3−31.

To prevent inaccurate comparisons caused by changing register bits, the software should disable the system time clock interrupt (clear the STEN bit in TSICTL) prior to writing to TSISTMSKL.

Figure 3−46. TSI System Time Clock Compare Mask LSB Register (TSISTMSKL)

31	0
ATCM	
R/W-0	

Legend: R/W = Read/Write; -*n* = value after reset

Table 3−31. TSI System Time Clock Compare Mask LSB Register (TSISTMSKL) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−0	ATCM	OF(*value*)	0−FFFF FFFFh	Not used.	Contains the 32 LSBs of the absolute time compare mask.

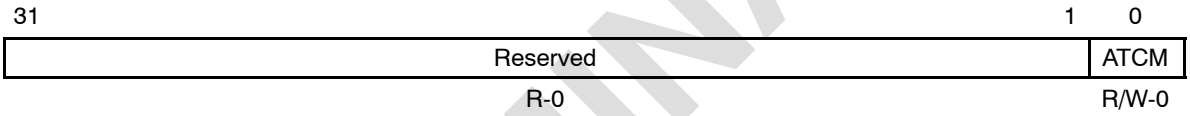
† For CSL implementation, use the notation VP_TSISTMSKL_ATCM_*symval*

3.13.19 TSI System Time Clock Compare Mask MSB Register (TSISTMSKM)

The transport stream interface system time clock compare mask MSB register (TSISTMSKM) holds the most-significant bit (MSB) of the absolute time compare mask (ATCM). This value is used with TSISTMSKL to mask out bits during the comparison of the ATC to the system time clock for absolute time. The bits that are set to one mask the corresponding ATC bits during the compare. TSISTMSKM is shown in Figure 3−47 and described in Table 3−32.

To prevent inaccurate comparisons caused by changing register bits, the software should disable the system time clock interrupt (clear the STEN bit in TSICTL) prior to writing to TSISTMSKM.

Figure 3−47. TSI System Time Clock Compare Mask MSB Register (TSISTMSKM)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 3−32. TSI System Time Clock Compare Mask MSB Register (TSISTMSKM) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31−1	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
0	ATCM	OF(*value*)	0−1	Not used.	Contains the MSB of the absolute time compare mask.

† For CSL implementation, use the notation VP_TSISTMSKM_ATCM_*symval*

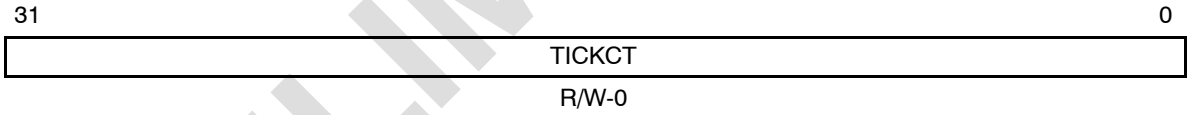
3.13.20 TSI System Time Clock Ticks Interrupt Register (TSITICKS)

The transport stream interface system time clock ticks interrupt register (TSITICKS) is used to generate an interrupt after a certain number of ticks of the 27-MHz system time clock. When the TICKCT value is set to *X* and the TCKEN bit in TSICTL is set, the TICK bit in VPIS is set every *X + 1* STCLK cycles. Note that the tick interrupt counter and comparison logic function are separate from the PCR logic and always count STCLK cycles regardless of the value of the CTMODE bit in TSICTL. TSITICKS is shown in Figure 3–48 and described in Table 3–33.

A write to TSITICKS resets the tick counter 0. Whenever the tick counter reaches the TICKCT value, the TICK bit in VPIS is set and the counter resets to 0.

To prevent inaccurate comparisons caused by changing register bits, the software should disable the tick count interrupt (clear the TCKEN bit in TSICTL) prior to writing to TSITICKS.

Figure 3–48. TSI System Time Clock Ticks Interrupt Register (TSITICKS)



Legend: R/W = Read/Write; -*n* = value after reset

Table 3–33. TSI System Time Clock Ticks Interrupt Register (TSITICKS) Field Descriptions

				Description	
Bit	**Field**	***symval*†**	**Value**	**BT.656, Y/C Mode, or Raw Data Mode**	**TSI Mode**
31–0	TICKCT	OF(*value*)	0–FFFF FFFFh	Not used.	Contains the number of ticks of the 27-MHz system time clock required to generate a tick count interrupt.

† For CSL implementation, use the notation VP_TSITICKS_TICKCT_*symval*

3.14 Video Capture FIFO Registers

The capture FIFO mapping registers are listed in Table 3–34. These registers provide read access to the capture FIFOs. These pseudo-registers should be mapped into DSP memory space rather than configuration register space in order to provide high-speed access. See the device-specific datasheet for the memory address of these registers. The function of the video capture FIFO mapping registers is listed in Table 3–35.

Table 3–34. Video Capture FIFO Registers

Offset Address†	Acronym	Register Name
00h	YSRCA	Y FIFO Source Register A
08h	CBSRCA	Cb FIFO Source Register A
10h	CRSRCA	Cr FIFO Source Register A
00h	YSRCB	Y FIFO Source Register B
08h	CBSRCB	Cb FIFO Source Register B
10h	CRSRCB	Cr FIFO Source Register B

† The absolute address of the registers is device/port specific and is equal to the FIFO base address + offset address. See the device-specific datasheet to verify the register addresses.

Table 3–35. Video Capture FIFO Registers Function

	Capture Mode		
Register	BT.656 or Y/C	Raw Data	TSI
YSRC*x*	Maps Y capture buffer into DSP memory.	Maps data capture buffer into the DSP memory.	Maps data capture buffer into the DSP memory.
CBSRC*x*	Maps Cb capture buffer into DSP memory.	Not used.	Not used.
CRSRC*x*	Maps Cr capture buffer into DSP memory.	Not used.	Not used.

In BT.656 or Y/C capture mode, three DMAs move data from the Y, Cb, and Cr capture FIFOs to the DSP memory by using the memory-mapped YSRC*x*, CBSRC*x*, and CRSRC*x* registers. The DMA transfers are triggered by the YEVT, CbEVT, and CrEVT events, respectively.

In raw capture mode, one DMA channel moves data from the Y capture FIFO to the DSP memory by using the memory-mapped YSRC*x* register. The DMA transfers are triggered by a YEVT event.

The video port packs receive data into 64-bit words in the FIFO and the DMA should always move 64-bit-wide data from YSRC*x*, CBSRC*x*, and CRSRC*x* to the memory.

Chapter 4

Video Display Port

The video port peripheral can operate as a video capture port, video display port, or transport stream interface (TSI) capture port. This chapter discusses the video display port.

Topic **Page**

4.1 Video Display Mode Selection

The following modes/features of the video port are currently not supported at this time. The video port is capable of using these modes/features but no technical assistance will be given at this time.

Modes:

- **Dual-sync display mode**

Features:

- **Noncontinuous display**
- **Single frame display**

The video display module operates in one of three modes as listed in Table 4−1. The DMODE bits are in the video display control register (VDCTL). The Y/C and 16/20-bit raw display modes may only be selected if the DCDIS bit in the video port control register (VPCTL) is cleared to 0.

Table 4−1. Video Display Mode Selection

DMODE Bits	Mode	Description
000	8-Bit ITU-R BT.656 Display	Digital video output is in YCbCr 4:2:2 with 8-bit resolution multiplexed in ITU-R BT.656 format.
001	10-Bit ITU-R BT.656 Display	Digital video output is in YCbCr 4:2:2 with 10-bit resolution multiplexed in ITU-R BT.656 format.
010	8-Bit Raw Display	8-bit data output
011	10-Bit Raw Display	10-bit data output
100	8-Bit Y/C Display	Digital video is output in YCbCr 4:2:2 with 8-bit resolution on parallel Y and Cb/Cr multiplexed channels.
101	10-Bit Y/C Display	Digital video is output in YCbCr 4:2:2 with 10-bit resolution on parallel Y and Cb/Cr multiplexed channels.
110	16-Bit Raw Display	16-bit data output.
111	20-Bit Raw Display	20-bit data output.

4.1.1 Image Timing

Display devices generate interlaced images by controlling the vertical retrace timing. The video display module emits a data stream used to generate a displayed image. An NTSC-compatible interlaced image with field and line information is shown in Figure 4−1. A progressive-scan image (SMPTE 296M compatible) is shown in Figure 4−2.

The active video area represents the pixels visible on the display. The active video area begins after the horizontal and vertical blanking intervals. The image area output by the video display module can be a subset of the active area. The relationship between frame, active video area, and image area is presented in Figure 4−3 for interlaced video and in Figure 4−4 for progressive video. The video display module generates timing for frames, active video areas within frames, and images within the active video area.

Figure 4−1. NTSC Compatible Interlaced Display

Figure 4−2. SMPTE 296M Compatible Progressive Scan Display

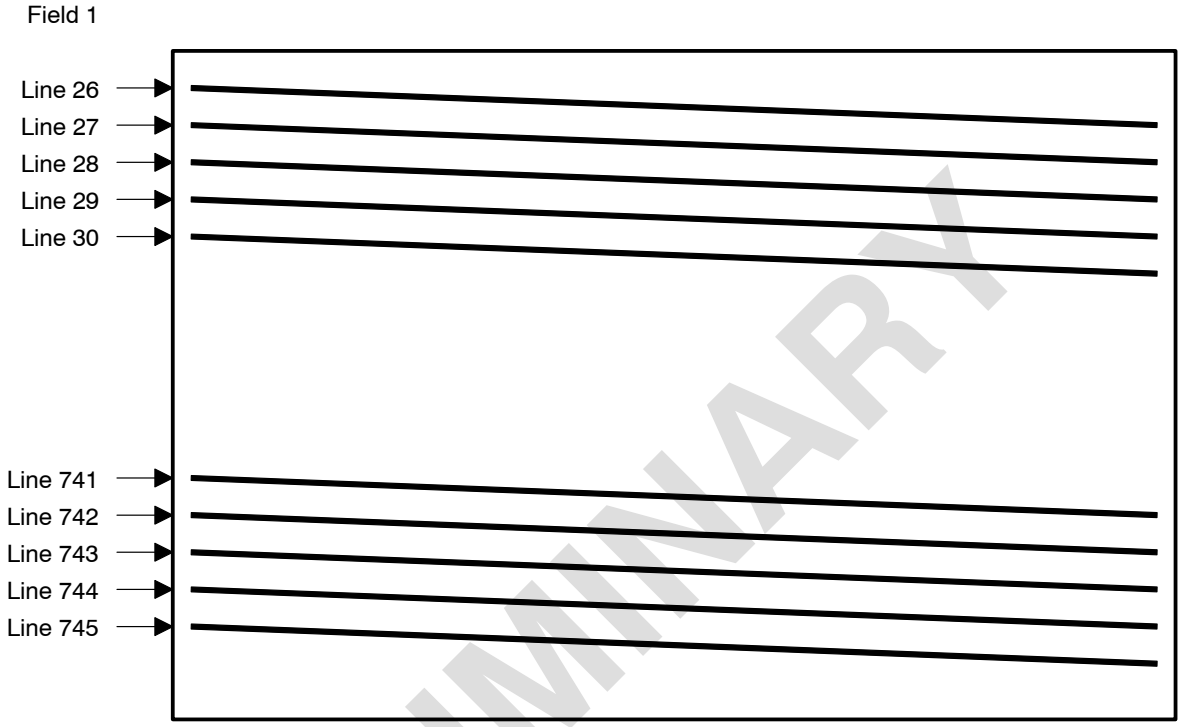


Figure 4–3. Interlaced Blanking Intervals and Video Areas

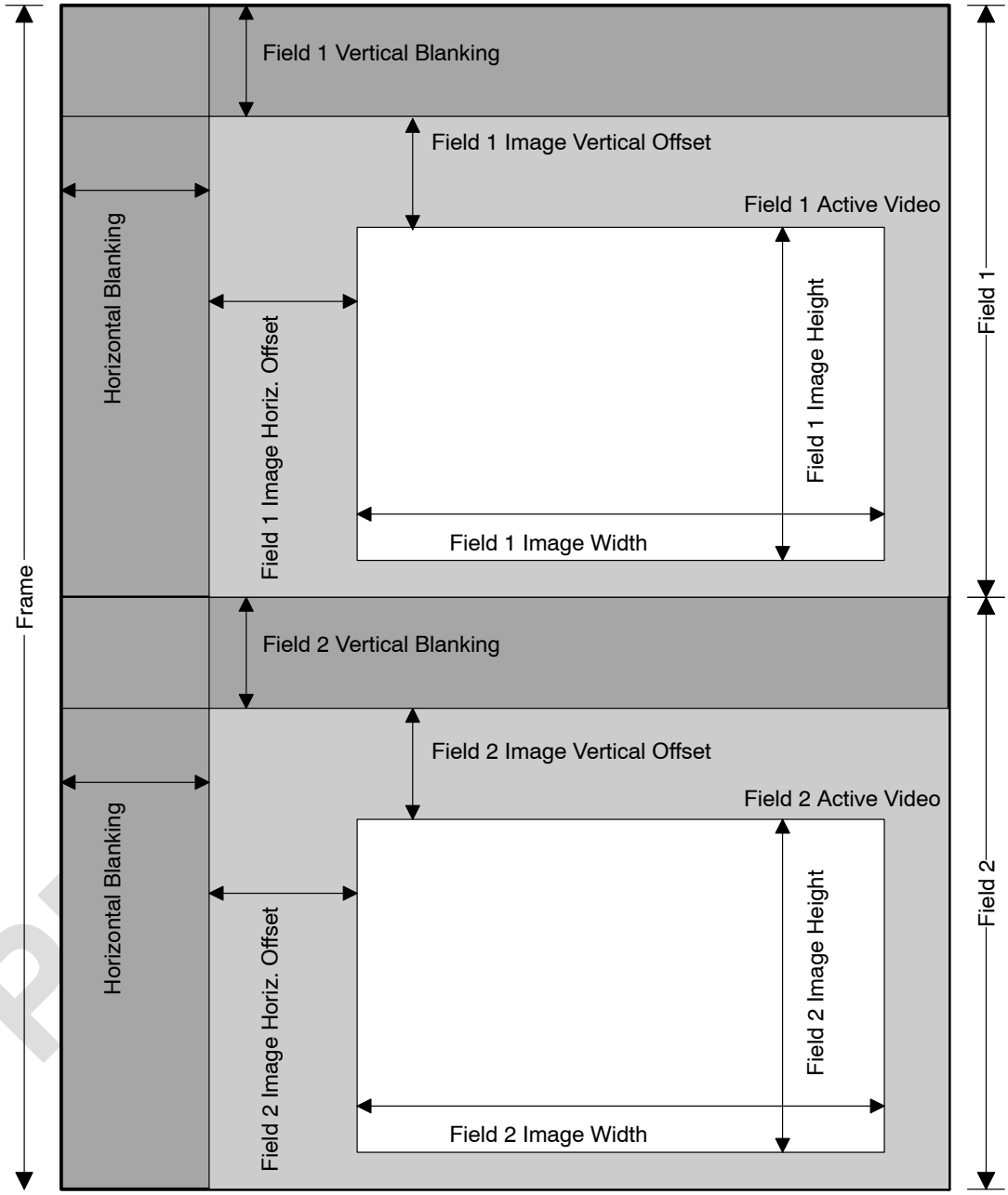
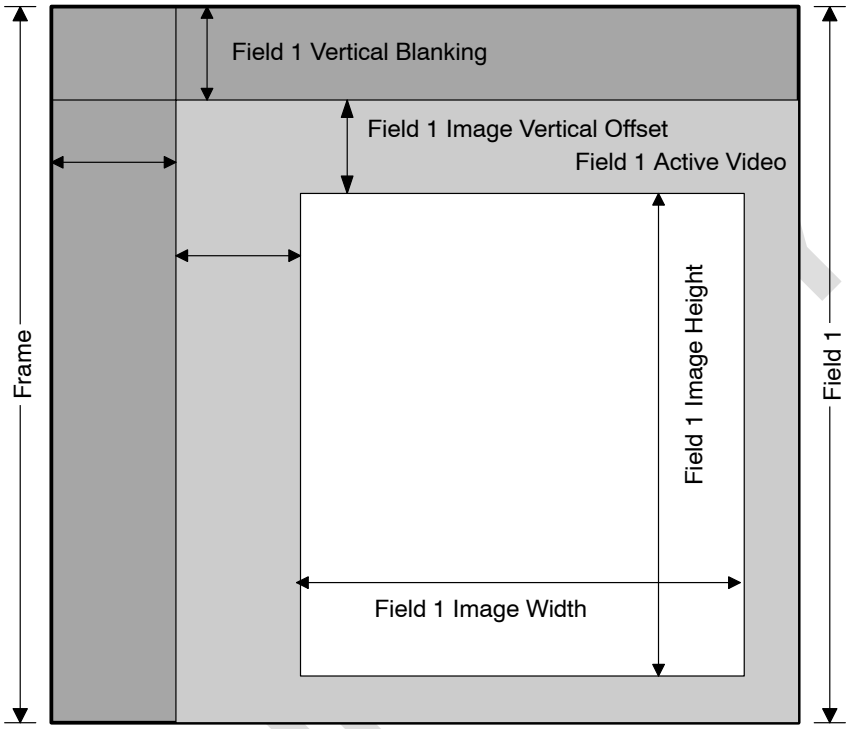


Figure 4−4. Progressive Blanking Intervals and Video Area



4.1.2 Video Display Counters

To generate the image timing, the video display module uses five counters:

- Frame line counter (FLCOUNT)
- Frame pixel counter (FPCOUNT)
- Image line counter (ILCOUNT)
- Image pixel counter (IPCOUNT)
- Video clock counter (VCCOUNT)

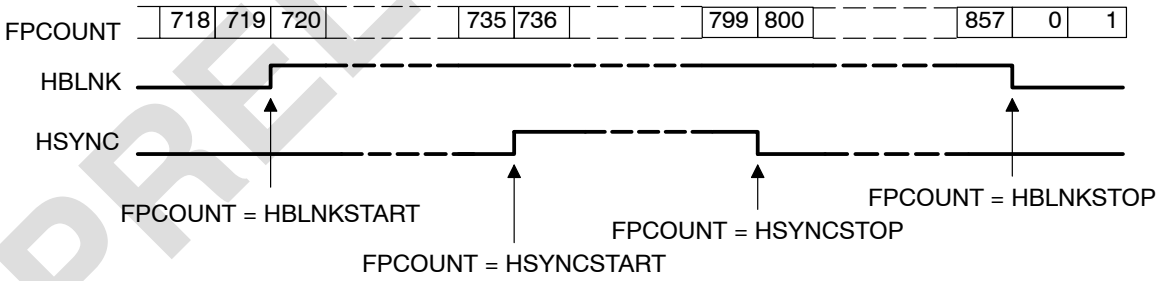
The frame line counter (FLCOUNT) counts the total number of lines per frame including vertical blanking intervals. The frame pixel counter (FPCOUNT) counts the total number of pixels per line including horizontal blanking intervals. FLCOUNT begins counting at the start of the vertical blanking interval of the first field. FPCOUNT begins counting at the end of the horizontal blanking interval of each line. They are reset when they reach their stop values as specified in the video display frame size register (VDFRMSZ).

The image line counter (ILCOUNT) and the image pixel counter (IPCOUNT) track the visible image within the field. ILCOUNT begins counting at the first display image line in each field. IPCOUNT begins counting at the first displayed image pixel on each line. They stop counting when they reach the image height and image width as specified in the video display field *n* image size register (VDIMGSZ*n*).

The video clock counter (VCCOUNT) counts VCLKIN transitions to determine when to increment FPCOUNT and IPCOUNT as determined by the video display mode. In Y/C mode, FPCOUNT and IPCOUNT increment on each VCLKIN rising edge. In BT.656 mode, FPCOUNT and IPCOUNT increment on every other VCLKIN rising edge. In raw mode, FPCOUNT and IPCOUNT increment on every 1 to 16 VCLKIN cycles as programmed by the INCPIX bits in the video display threshold register (VDTHRLD).

FPCOUNT and FLCOUNT are compared to various values to determine when to assert and negate various control signals. The 12-bit FPCOUNT is used to determine where to enable and disable horizontal sync and blanking information along each scan line. The state of FPCOUNT is reflected in the VDXPOS bits of the video display status register (VDSTAT). Figure 4–5 shows how the horizontal blanking and horizontal synchronization signals are triggered. (HBLNK and HSYNC are shown active high).

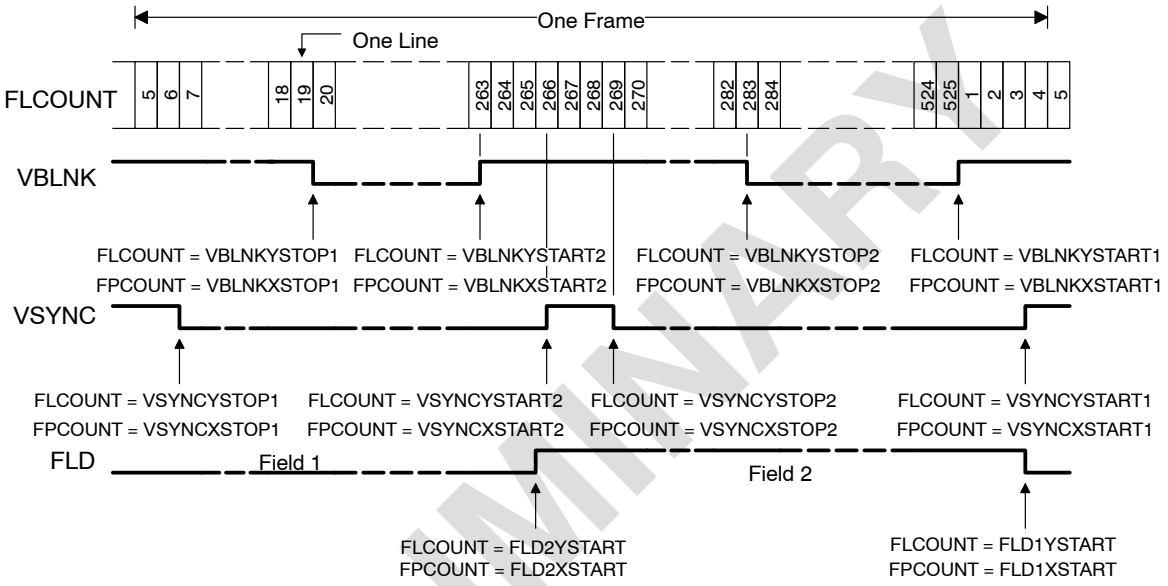
Figure 4–5. Horizontal Blanking and Horizontal Sync Timing



The 12-bit FLCOUNT counts which scan line is being generated. The FLCOUNT is reset to 1 after reaching the count specified in VDFRMSZ. (For BT.656 operation, the FRMHIGHT would be set to 525 (525/60 operation) or 625 (625/50 operation).) The state of FLCOUNT is reflected in the VDYPOS bits of VDSTAT. Figure 4–6 shows how the vertical blanking, vertical synchronization, and field identification signals are triggered. (VBLNK and VSYNC are shown active high.)

Note that the signals can transition at any place along the video line (specified by the XSTART and XSTOP bits of the appropriate registers). In this case, VBLNK starts at horizontal count VBLNKXSTART2 = 429 on scan line VBLNKYSTART2 = 263 (565/60 operation).

Figure 4−6. Vertical Blanking, Sync and Even/Odd Frame Signal Timing



4.1.3 Sync Signal Generation

The video display module must generate a number of control signals for both internal and external use. As seen in section 4.1.2, the HSYNC, HBLNK, VSYNC, VBLNK, and FLD signals are generated directly from the pixel and line counters and comparison registers. Several additional signals are also generated indirectly for use in external control.

A composite blank (CBLNK) signal is generated as the logical-OR of the HBLNK and VBLNK signals. A composite sync (CSYNC) signal is also generated as the logical-OR of the HSYNC and VSYNC signals. (This is not a true analog CSYNC, which must include serration pulses during VSYNC and equalization pulses during vertical front and back porch periods.) Finally, an active video (AVID) signal is generated. AVID is the inverted CBLNK signal indicating when active video data is being output.

Up to three of the eight sync signals may be output on VCTL0, VCTL1, and VCTL2 as selected by the video display control register (VDCTL). Each signal may be output in its noninverted or inverted form, as selected by the VCT*n*P bits in the video port control register (VPCTL).

4.1.4 External Sync Operation

The video display module may be synchronized with an external video source using external sync signals. VCTL0 may be configured as an external horizontal sync input. When the external HSYNC is asserted, FPCOUNT is loaded with the HRLD value and VCCOUNT is loaded with the CRLD value. VCTL1 may be configured as an external vertical sync input. When the external VSYNC is asserted during field 1, FLCOUNT is loaded with the VRLD value. Field determination is made using either VCTL2 as an external FLD input or by field detect logic using the VSYNC and HSYNC inputs.

4.1.5 Port Sync Operation

This mode/feature of the video port is currently not supported at this time.

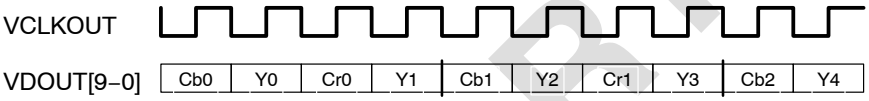
The video display module may be synchronized with the video display module of another video port on the device. This mode is provided to enable the output of 24-bit or 30-bit RGB data. (for example, 8 bits of R and 8 bits of G on video port 0 operating in dual-channel synced 8-bit raw mode, and 8 bits of B on video port 1 operating in 8-bit raw mode with VP1 synced to VP0.) The slave port must have the same VCLKIN and programmed register values as the master port. The master port provides the control signals necessary to reset the slave port counters so that they maintain synchronization. Each video port may only synchronize to the previous video port (the one with a lower number). An example for a three port device is shown in Figure 4–7.

Figure 4–7. Video Display Module Synchronization Chain

4.2 BT.656 Video Display Mode

The BT.656 display mode outputs 8-bit or 10-bit 4:2:2 co-sited luma and chroma data multiplexed into a single data stream. Pixels are output in pairs with each pair consisting of two luma samples and two chroma samples. The chroma samples are associated with the first luma pixel of the pair. Output pixels are valid on the positive edge of VCLKOUT in the sequence CbYCrY as shown in Figure 4−8.

Figure 4−8. BT.656 Output Sequence



4.2.1 Display Timing Reference Codes

The end active video (EAV) code and start active video (SAV) code are issued at the start of each video line. EAV and SAV codes have a fixed format. The format is shown in Table 3−2 (page 3-5). The EAV and SAV codes define the end and start of the horizontal-blanking interval, respectively, and they also indicate the current field number and the vertical blanking interval. The SAV and EAV codes have a 4-bit protection field to ensure valid codes. The video display module generates these protection bits as part of the SAV and EAV codes. Table 3−3 (page 3-6) shows possible combinations of valid SAV and EAV codes with their protection bits. The video display pipeline generates SAV and EAV sync codes and inserts them into the output video stream according to the BT.656 specification.

The BT.656 line timing is shown in Figure 4−9 and Figure 4−10. Each line begins with an EAV code, a blanking interval, an SAV code, followed by the line of active video. The EAV code indicates the end of active video for the previous line, and the SAV code indicates the start of active video for the current line.

Figure 4−9. 525/60 BT.656 Horizontal Blanking Timing

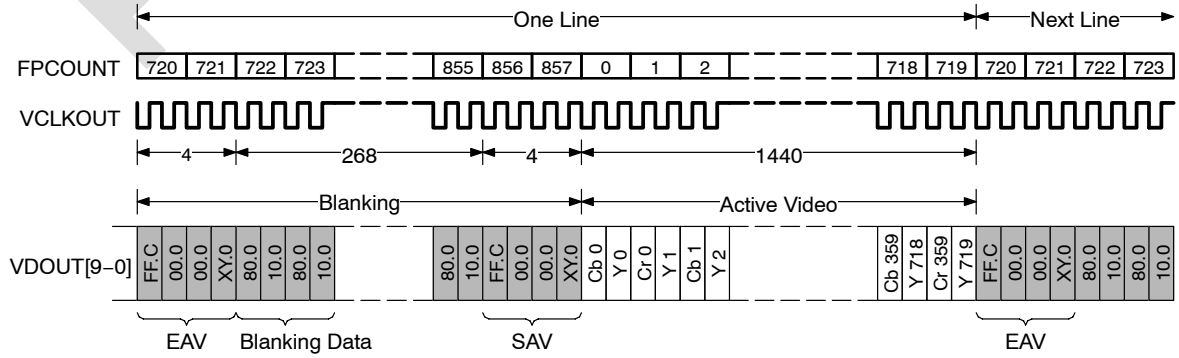
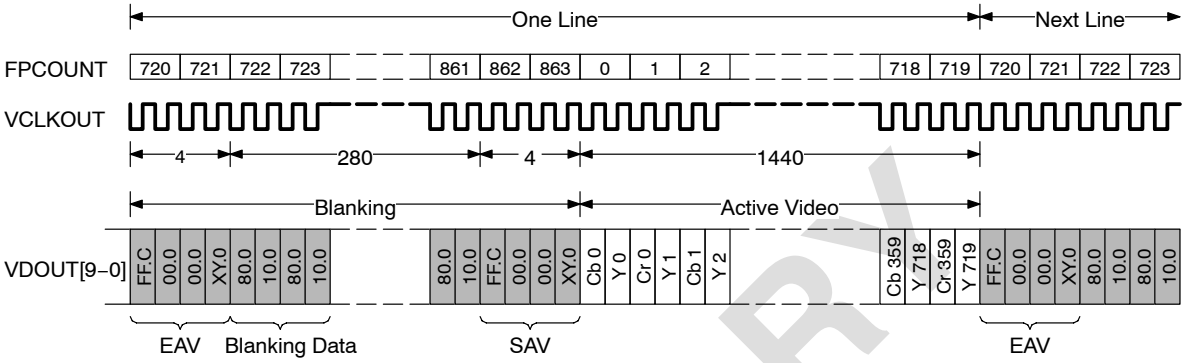


Figure 4−10. 625/50 BT.656 Horizontal Blanking Timing



SAV and EAV codes are identified by a 3-byte preamble of FFh, 00h, and 00h. This combination must be avoided in the video data output by the video port to prevent accidental generation of an invalid sync code. The video display module provides programmable maximum and minimum value clipping on the video data to prevent this possibility.

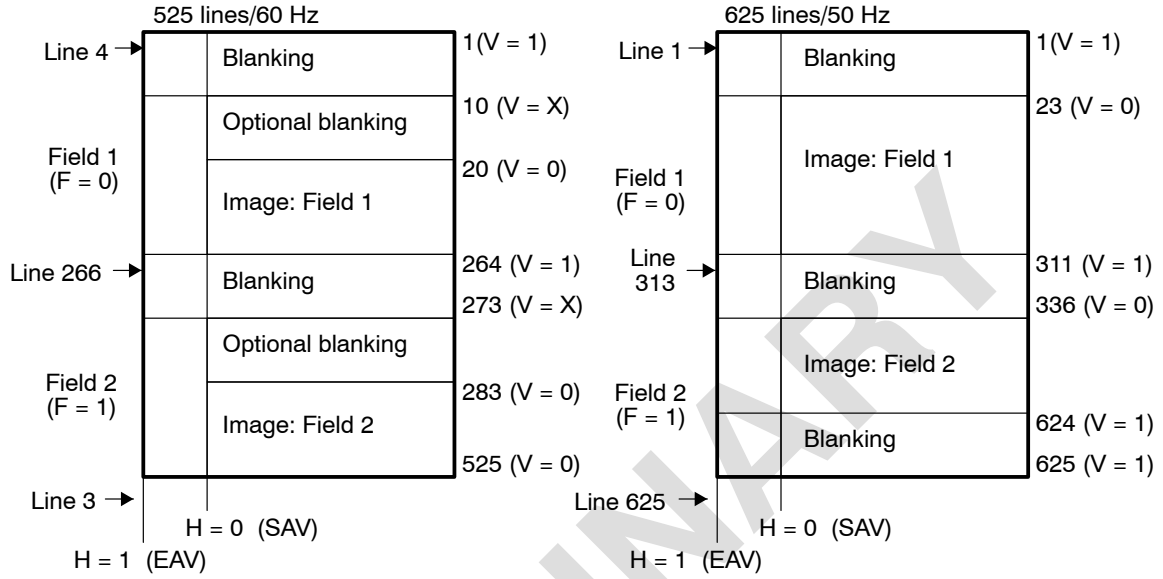
The typical values for H, V, and F on different lines are shown in Table 4−2 and Figure 4−11.

F and V are only allowed to change at EAV sequences. The EAV and SAV sequences must occupy the first four words and the last four words of the digital horizontal-blanking interval, respectively. The EAV code is inserted when FPCOUNT = HBLNKSTART. The SAV code is inserted when FPCOUNT = HBLNKSTOP.

Table 4−2. BT.656 Frame Timing

Line Number				
625/50	**525/60**	**F**	**V**	**Description**
624−625	1−3	1	1	Vertical blanking for field 1, EAV/SAV code still indicates field 2.
1−22	4−19	0	1	Vertical blanking for field 1. Change EAV/SAV code to field 1.
23−310	20−263	0	0	Active video, field 1.
311−312	264−265	0	1	Vertical blanking for field 2, EAV/SAV code still indicates field 1.
313−335	266−282	1	1	Vertical blanking for field 2. Change EAV/SAV code to field 2.
336−623	283−525	1	0	Active video, field 2.

Figure 4−11. Digital Vertical F and V Transitions



LIne Number	F	V	H (EAV)	H (SAV)
1−3	1	1	1	0
4−19	0	1	1	0
20−263	0	0	1	0
264−265	0	1	1	0
266−282	1	1	1	0
283−525	1	0	1	0

Line Number	F	V	H (EAV)	H (SAV)
1−22	0	1	1	0
23−310	0	0	1	0
311−312	0	0	1	0
313−335	1	1	1	0
336−623	1	0	1	0
624−625	1	1	1	0

4.2.2 Blanking Codes

The time between the EAV and SAV code on each line represents the horizontal blanking interval. During this time, the video port outputs digital video blanking values. These values are 10.0h for luma (Y) samples and 80.0h for chroma (Cb/Cr) samples. These values are also output during the active line period of vertical blanking (between SAV and EAV when V = 1). In addition, if the DVEN bit in VDCTL is cleared to 0, the blanking values are output during the portion of active video lines that are not a part of the displayed image.

4.2.3 BT.656 Image Display

For BT.656 display mode, the FIFO buffer is divided into three sections. One FIFO is 2560-bytes deep and is used for the storage of Y output samples; the other two FIFOs are each 1280-bytes deep and are dedicated for storage of Cb and Cr samples. Each FIFO has a memory-mapped location associated with it; YDST, CBDST, and CRDST. The pseudo-registers are write-only and are used by DMAs to fill the FIFOs with output data. The video display module multiplexes the data from the three FIFOs to generate the output CbYCrY data stream.

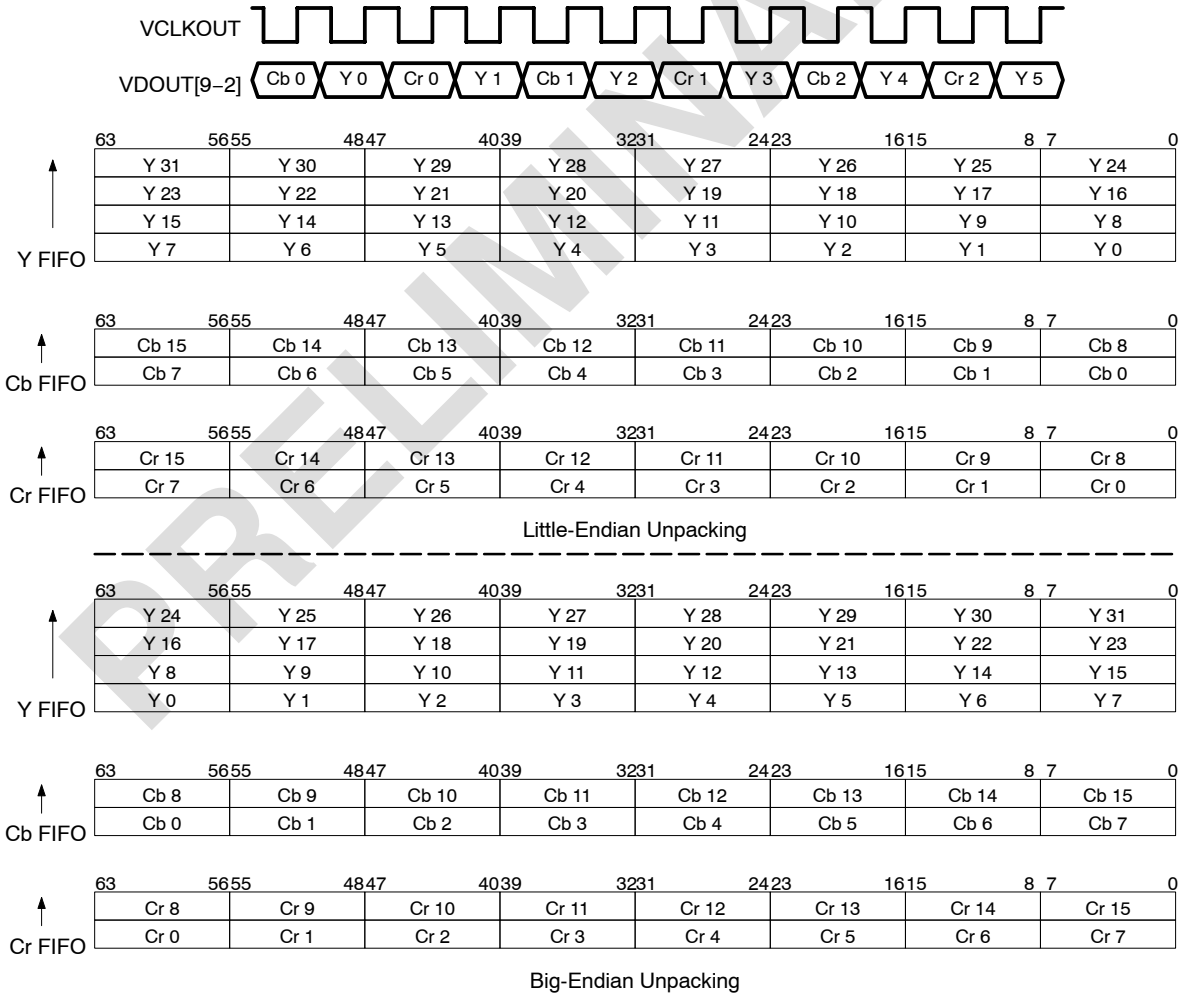
If video display is enabled, the video display module uses the YEVT, CbEVT, and CrEVT events to notify the DMA controller that data needs to be placed into the display FIFOs. The number of pixels required to generate the events is set by the VDTHRLD bits in VDTHRLD (VTHRLD must be an even number). The video display module generates the event signals when the display buffer holds less than the VDTHRLD number of pixels and the DEVTCT counter has not expired. On every YEVT, the DMA should move data from DSP memory to the Y buffer, using the Y FIFO destination register (YDST) content as the destination address. On every CbEVT, the DMA should move data from DSP memory to the Cb buffer, using the Cb FIFO destination register (CBDST) content as the destination address. On every CrEVT, the DMA should move data from DSP memory to the Cr buffer, using the Cr FIFO destination register (CRDST) content as the destination address. The DMA transfer size for the Y buffer is twice the size of the DMA for the Cb or Cr buffers.

4.2.4 BT.656 FIFO Unpacking

Display data is always packed into the FIFOs in 64-bit words and must be unpacked before being sent to the video display data pipeline. The unpacking and byte ordering is dependant upon the display data size and the device endian mode. For little-endian operation (default), data is unpacked from right to left; for big-endian operation, data is unpacked from left to right.

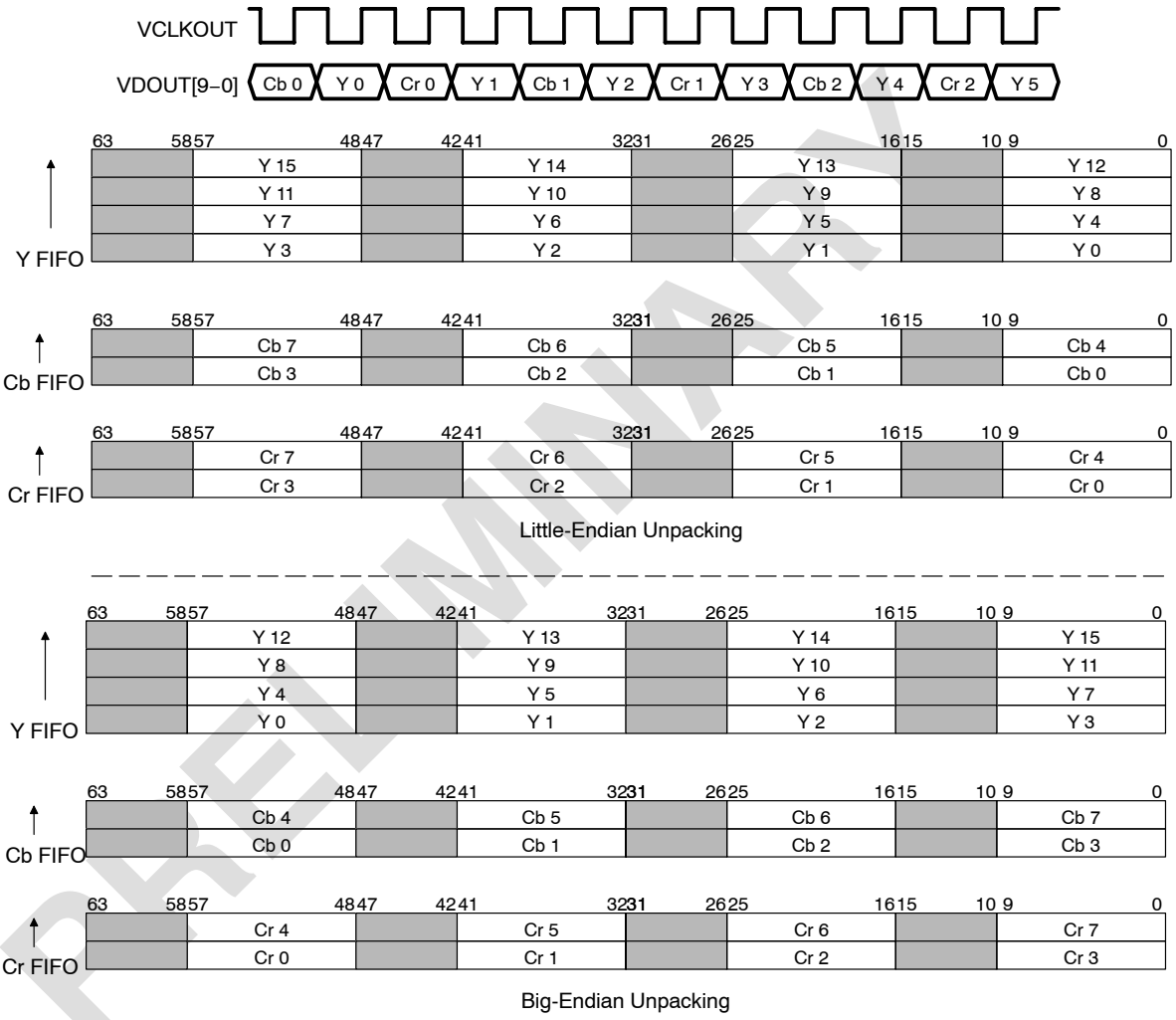
The 8-bit BT.656 mode uses three FIFOs for color separation. Four samples are unpacked from each word as shown in Figure 4−12.

Figure 4−12. 8-Bit BT.656 FIFO Unpacking



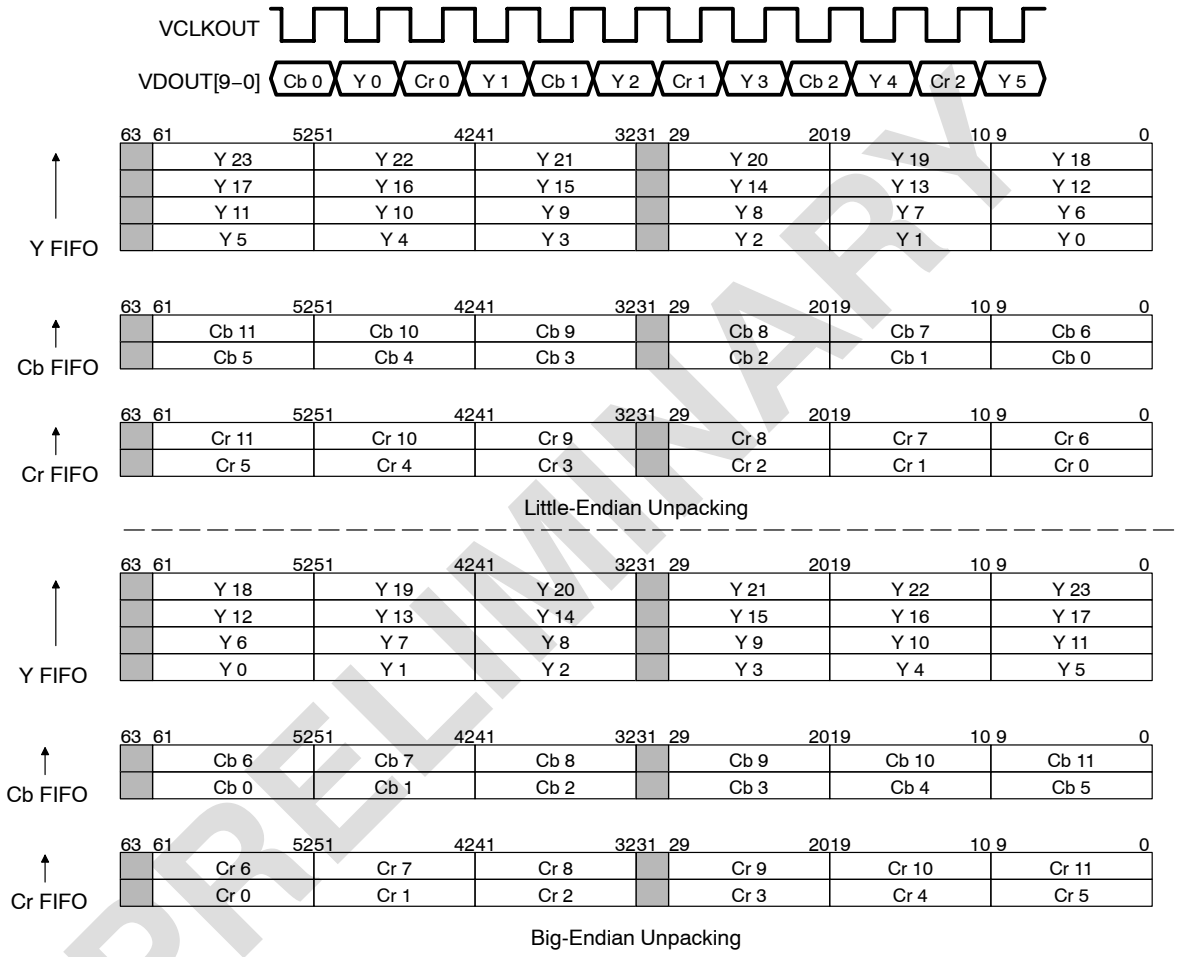
For 10-bit BT.656 operation, two samples are unpacked from each word as shown in Figure 4−13.

Figure 4−13. 10-Bit BT.656 FIFO Unpacking



In 10-bit BT.656 dense-pack mode, three samples are unpacked from each word in the FIFO as seen in Figure 4−14.

Figure 4−14. BT.656 Dense FIFO Unpacking



4.3 Y/C Video Display Mode

The Y/C display mode is similar to the BT.656 display mode but outputs 8 or 10-bit data on separate luma and chroma data streams. One data stream contains Y samples and the other stream contains multiplexed Cb and Cr samples co-sited with every other luminance sample. The Y samples are read from the Y FIFO and the Cb and Cr samples are read from the Cb and Cr FIFOs and combined on the chroma output. The unpacking and order of the samples is determined by the sample size (8-bit or 10-bit) and the device endian mode.

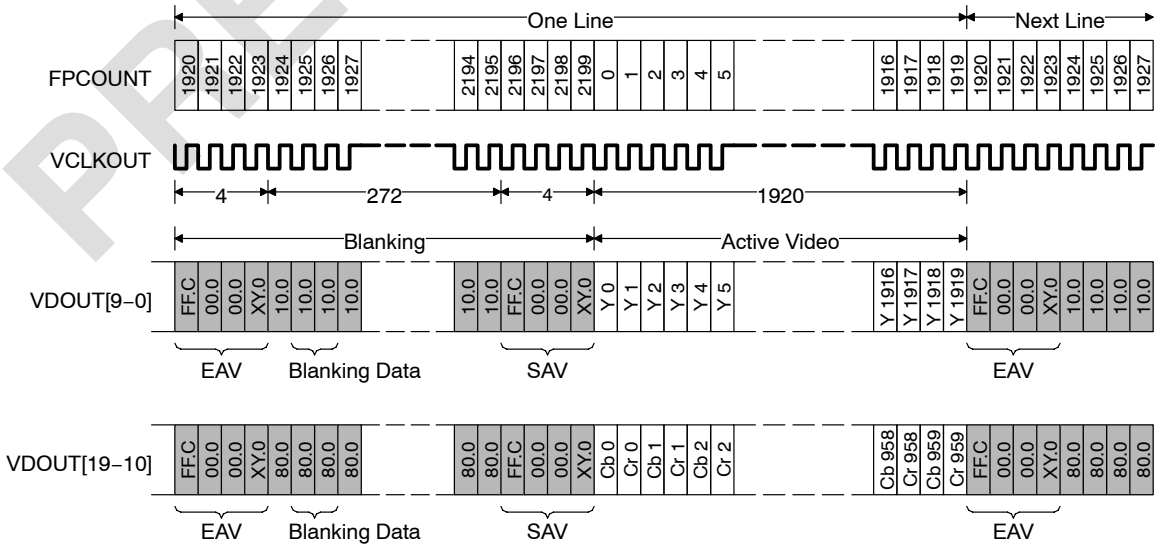
The Y/C display mode can generate HDTV standard output such as BT.1120, SMPTE260, or SMPTE296 with embedded EAV and SAV codes. It can also output separate control signals.

Because 16 or 20 bits are used for data output, the Y/C output mode requires both halves of the video port data bus. If the DCHDIS bit in VPCTL is set, then Y/C mode cannot be selected.

4.3.1 Y/C Display Timing Reference Codes

The EAV and SAV embedded timing codes are identical to those output in BT.656 mode and timing is controlled in the same manner. In Y/C mode, however, the codes must be output on both the Y and C data streams (VDOUT[9–0] and VDOUT[19–10]). An example of BT.1120 line timing is shown in Figure 4–15.

Figure 4–15. Y/C Horizontal Blanking Timing (BT.1120 60I)



4.3.2 Y/C Blanking Codes

The time between the EAV and SAV code on each line represents the horizontal blanking interval. During this time, the video port outputs the digital video blanking values. These values are 10.0h for luma (Y) samples and 80.0h for chroma (Cb/Cr) samples. These values are also output during the active line period of vertical blanking (between SAV and EAV when V = 1), unless replaced by VBI data. In addition, if the DVEN bit in VDCTL is 0, the blanking values are output during the portion of active video lines that are not a part of the displayed image.

4.3.3 Y/C Image Display

Many of the standards supported by the Y/C display mode provide for both interlaced and progressive scan formats. For interlaced display, the display controls are programmed identically to BT.656 mode. For progressive scan formats, the frame size is programmed to the size of a single field and only field 1 is used.

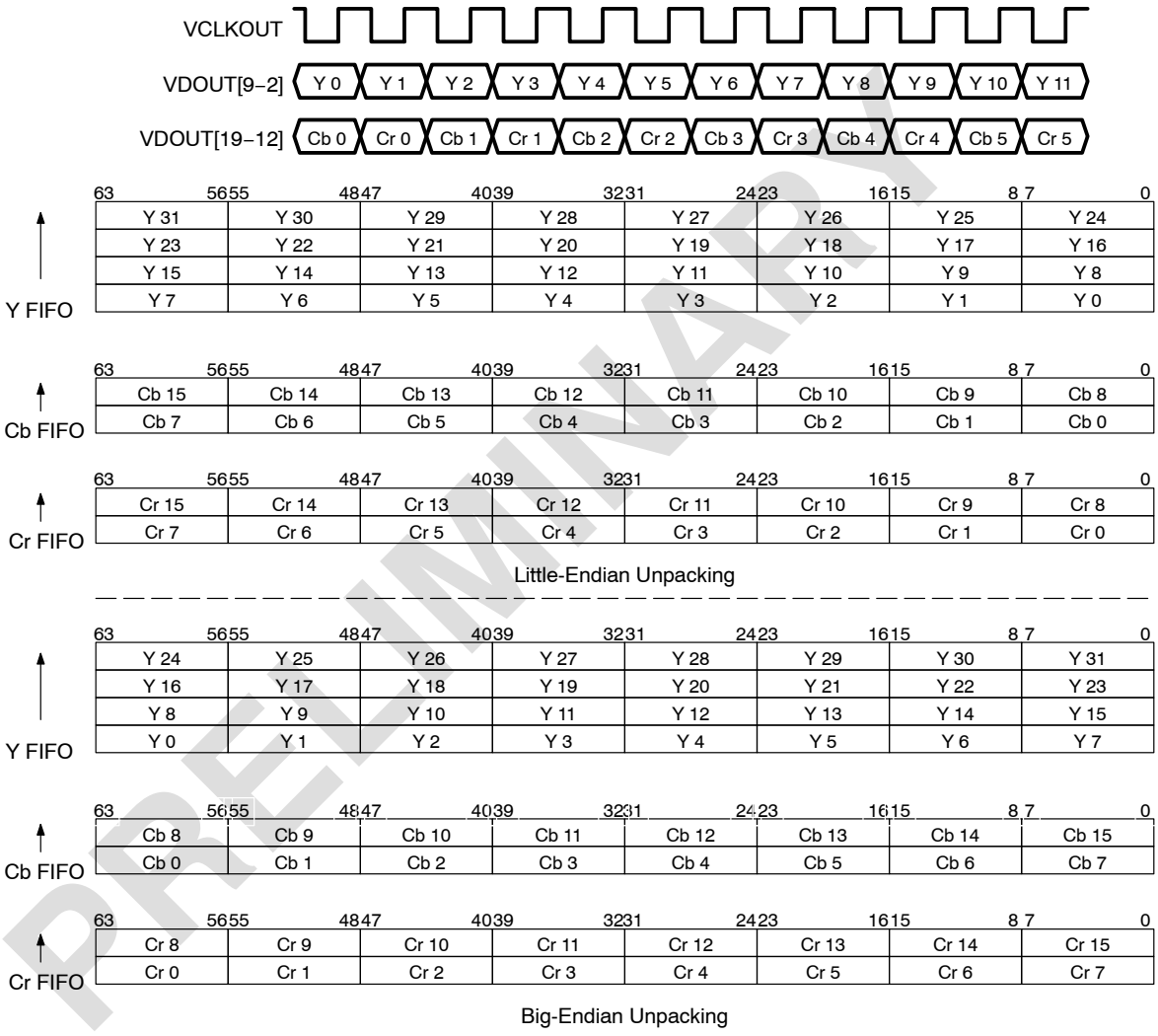
The Y/C display mode uses the same FIFO organization as the BT.656 display mode and generates DMA events in the same manner.

4.3.4 Y/C FIFO Unpacking

Display data is always packed into the FIFOs in 64-bit words and must be unpacked before being sent to the display data pipeline. The unpacking and byte ordering is dependant upon the display data size and the device endian mode. For little-endian operation (default), data is unpacked from right to left; for big-endian operation, data is unpacked from left to right.

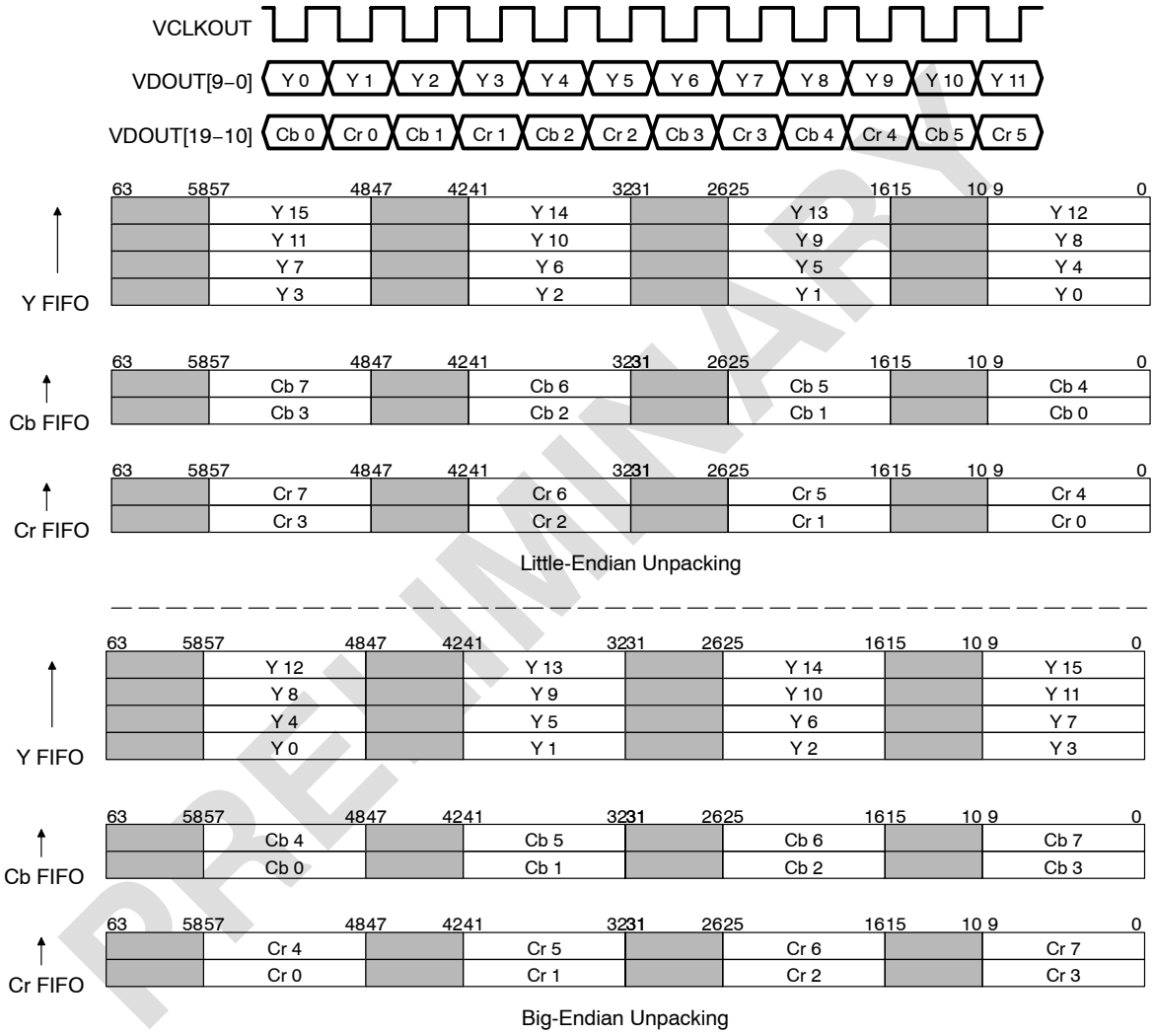
The 8-bit Y/C mode uses three FIFOs for color separation. Four samples are unpacked from each word as shown in Figure 4−16.

Figure 4−16. 8-Bit Y/C FIFO Unpacking



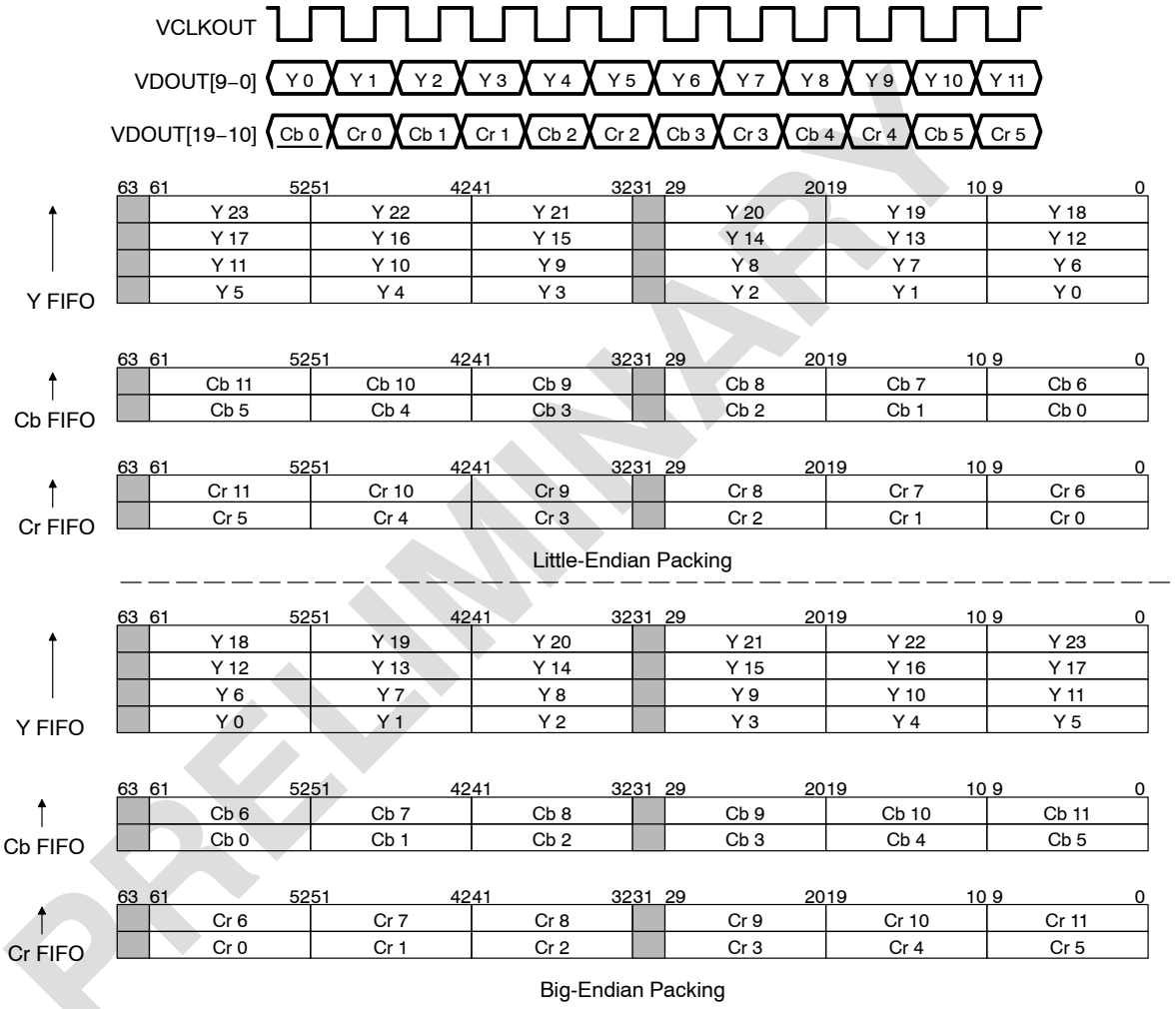
For 10-bit operation, two samples are unpacked from each FIFO word. This is shown in Figure 4−17.

Figure 4−17. 10-Bit Y/C FIFO Unpacking



In 10-bit Y/C dense-pack mode, three samples are unpacked from each word in the FIFO as seen in Figure 4−18.

Figure 4−18. 10-Bit Y/C Dense FIFO Unpacking



4.4 Video Output Filtering

The video output filter performs simple hardware scaling and resampling on outgoing 8-bit BT.656 or 8-bit Y/C data. Filtering hardware is disabled during 10-bit or raw data display modes.

4.4.1 Output Filter Modes

The output filter has four modes of operation: no-filtering, 2× scaling, chrominance resampling, and 2× scaling with chrominance resampling. Filter operation is determined by the DMODE, SCALE, and RESMPL bits of the VDCTL.

Table 4–3 shows the output filter mode selection. When 8-bit BT.656 or Y/C display operation is selected, (DMODE = x00), scaling is selected by setting the SCALE bit and chrominance resampling is selected by setting the RESMPL bit. If 8-bit BT.656 or Y/C display is not selected (DMODE ≠ x00), filtering is disabled.

Table 4–3. Output Filter Mode Selection

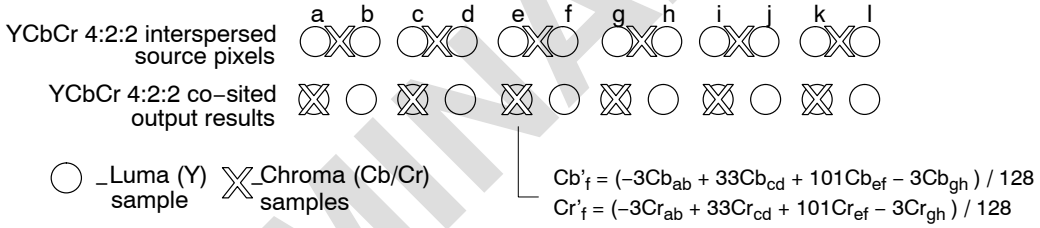
VDCTL Bit			
DMODE	**RESMPL**	**SCALE**	**Filter Operation**
x00	0	0	No filtering
x00	0	1	2× scaling
x00	1	0	Chrominance resampling (full scale)
x00	1	1	2× scaling with chrominance resampling
x01	x	x	No filtering
x10	x	x	No filtering
x11	x	x	No filtering

4.4.2 Chrominance Resampling Operation

Chrominance resampling computes chrominance values at sample points corresponding to output luminance samples based on the input interspersed chrominance samples. This filter performs the conversion between interspersed YCbCr 4:2:2 format and co-sited YCbCr 4:2:2 format. The vertical portion of the conversion from YCbCr 4:2:0 to interspersed YCbCr 4:2:2 must be performed in software.

The chrominance resampling filters calculate the implied value of Cb and Cr co-sited with luminance sample points based upon nearby interspersed Cb and Cr samples. The resulting values are clamped to between 01h and FEh before being output. Chrominance resampling is shown in Figure 4−19.

Figure 4−19. Chrominance Resampling



4.4.3 Scaling Operation

The 2×-scaling mode is used to double the horizontal resolution of output luminance and chrominance data. This allows processed CIF resolution images to be output at full size. Vertical scaling must be performed in software. Scaling for co-sited source is shown in Figure 4−20 and scaling for interspersed source is shown in Figure 4−21.

For a co-sited source, the source luminance pixels are output unchanged for every even pixel (a, b, c, etc., in Figure 4−20). Odd luminance pixels (a', b', c', etc.) are generated from neighboring source (even) pixels using a four tap filter. The chrominance source pixels are output unchanged for every other even pixel (a, c, e, etc.). Other even output pixel (b, d, f, etc.) chrominance values are generated from neighboring source chrominance pixels using a four tap filter.

For an interspersed source, the luminance is output identically to the co-sited case. Chrominance output is generated using a four tap filter with one of two different coefficient sets depending on which source chrominance pixel the output pixel is closest.

Note that because input scaling is limited to 2x, full BT.656 width output is not achieved from CIF source images. The horizontal location of the reduced image can be adjusted using HOFFSET.

Figure 4−20. 2x Co-Sited Scaling

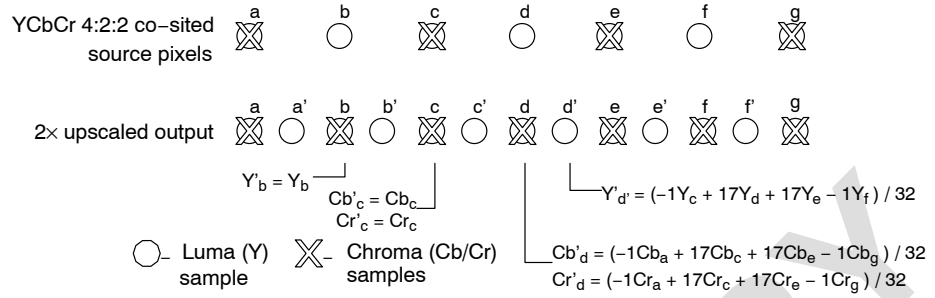
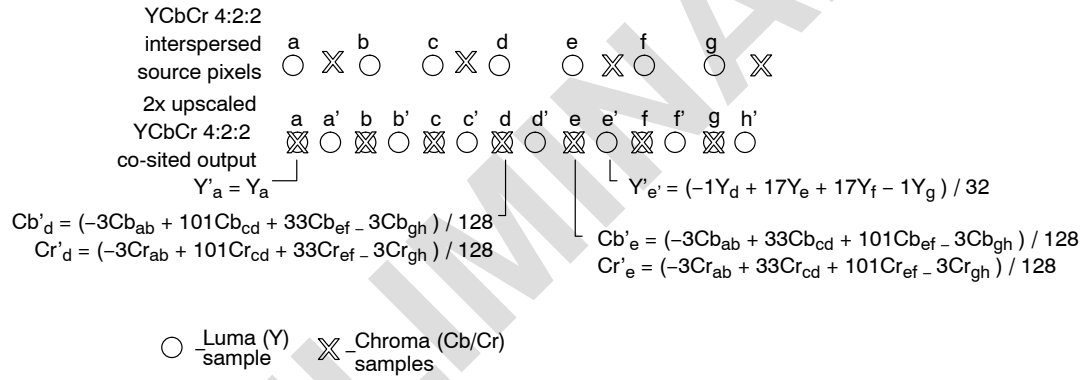


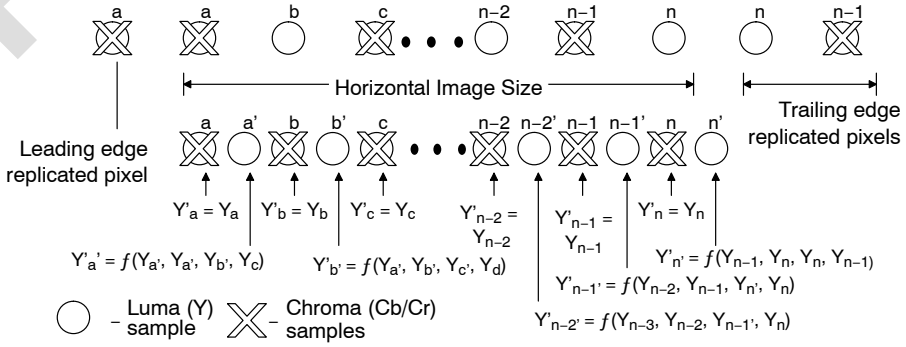
Figure 4−21. 2x Interspersed Scaling



4.4.4 Edge Pixel Replication

Because four tap filters are used on the output, the first and last two pixels on each line must be mirrored. An example of how the filter uses the mirrored pixels for the luminance filter (2× co-sited) is shown in Figure 4−22.

Figure 4−22. Output Edge Pixel Replication



Examples of luma edge and chroma edge replication for 2× interspersed to co-sited output are shown in Figure 4–23 and Figure 4–24, respectively.

Figure 4–23. Luma Edge Replication

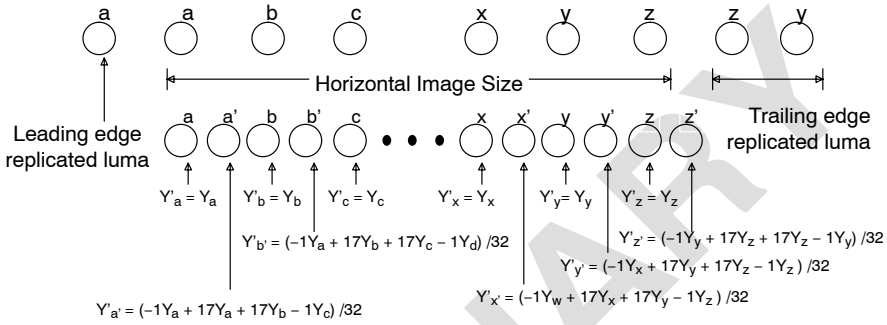
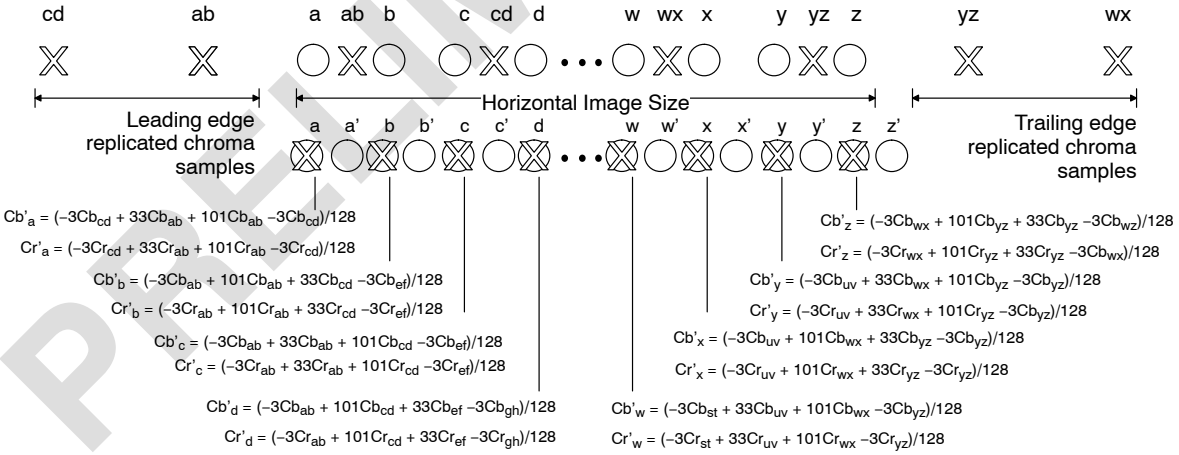


Figure 4–24. Interspersed Chroma Edge Replication



4.5 Ancillary Data Display

The following sections discuss ancillary data display. No special previsions are made for the display of horizontal ancillary (HANC) or vertical ancillary (VANC), also called vertical blanking interval (VBI), data.

4.5.1 Horizontal Ancillary (HANC) Data Display

HANC data can be displayed using the normal video display mechanism by programming IMGHSIZE*n* to occur prior to the SAV code. The HANC data including the ancillary data header must be part of the YCbCr separated data in the FIFOs. The VCTHRLD value and DMA size must be programmed to comprehend the additional samples. You must disable scaling and chroma re-sampling when including the display of HANC data to prevent data corruption.

4.5.2 Vertical Ancillary (VANC) Data Display

VANC (or VBI) data is commonly used for such features as teletext and closed-captioning. No special provisions are made for the display of VBI data. VBI data may be displayed using the normal display mechanism by programming IMGVOFF to occur before the first line of active video on the first line of desired VBI data. Note that the VBI data must be YCbCr separated. You must disable scaling and chroma resampling when the display of VBI data is desired or the data will be corrupted by the filters.

4.6 Raw Data Display Mode

The raw data display modes are intended to output data to a RAMDAC or other D/A-type device. This is typically RGB formatted data. No timing information is inserted into the output data stream; instead, selectable control signals are output to indicate timing. Raw data display includes a synchronized dual channel option. This allows channel B to output a separate data stream using the same clock and control as channel A. This mode is useful when used with a second video port in systems that require 24-bit to 30-bit RGB output.

The raw data mode uses a single FIFO of 5120 bytes for storage of output data. The FIFO is filled by DMAs writing to the Y FIFO destination register A (YDSTA). DMAs are requested using the YEVTA event. In raw sync mode (RSYNC bit is set), the FIFO is split into 2560-byte channel A and B buffers. The channel B FIFO is filled by DMAs using the Y FIFO destination register B (YDSTB) as a destination. Both YEVTA and YEVTB events are generated using the channel A timing control.

4.6.1 Raw Mode RGB Output Support

The raw data display mode has a special pixel count feature that allows the FPCOUNT increment rate to be set. FPCOUNT increments only when INCPIX samples have been sent out. This option allows proper tracking of the display pixels when sending out sequential RGB samples. (INCPIX would be set to three in this case, to indicate that a single pixel is represented by three output samples.)

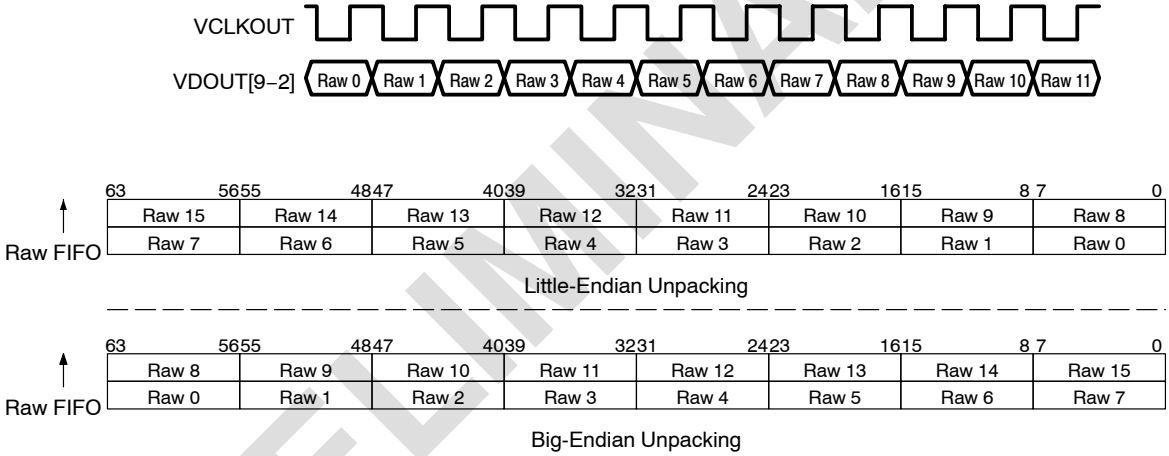
Sequential RGB samples output are also supported through a special FIFO unpacking mode. When the 8-bit raw ¾ unpacking is selected (RGBX bit in VDCTL), three output bytes are selected from each word and the fourth byte is ignored. This allows the video port to correctly output data formatted as 24-bit RGB (or RGBα) words in memory.

4.6.2 Raw Data FIFO Unpacking

Display data is always packed into the FIFOs in 64-bit words and must be unpacked before being sent to the display data pipeline. The unpacking and byte ordering is dependant upon the display data size and the device endian mode. For little-endian operation (default), data is unpacked from right to left; for big-endian operation, data is unpacked from left to right.

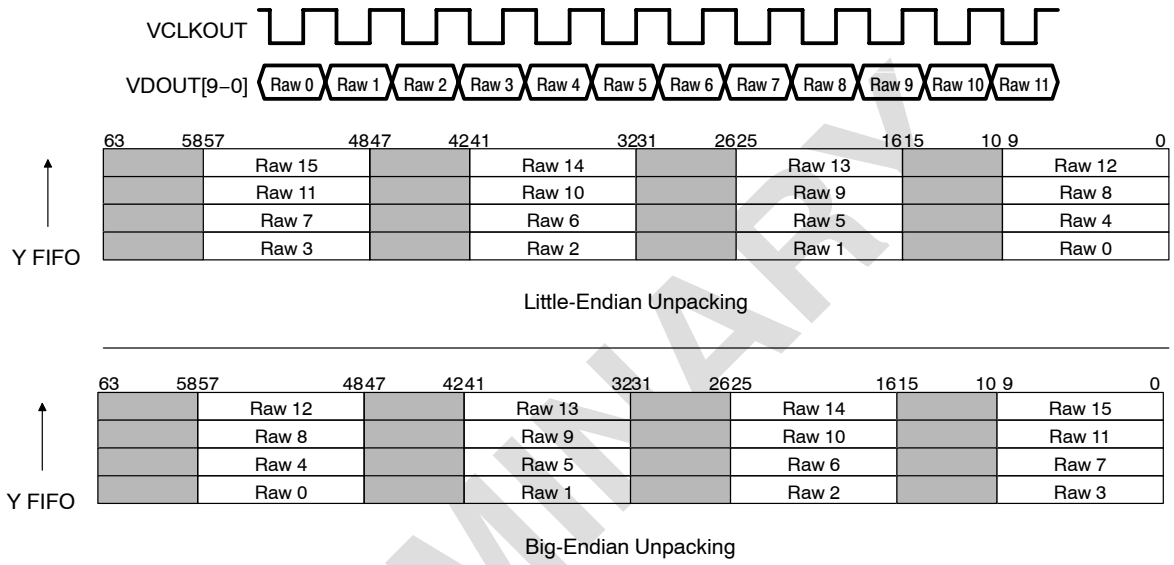
The 8-bit raw mode uses a single data FIFO. Four samples are unpacked from each word as shown in Figure 4−25.

Figure 4−25. 8-Bit Raw FIFO Unpacking



For 10-bit operation, two samples are unpacked from each FIFO word. This is shown in Figure 4–26.

Figure 4–26. 10-Bit Raw FIFO Unpacking



In 10-bit raw dense-pack mode, three samples are unpacked from each word in the FIFO as seen in Figure 4–27.

Figure 4–27. 10-Bit Raw Dense FIFO Unpacking

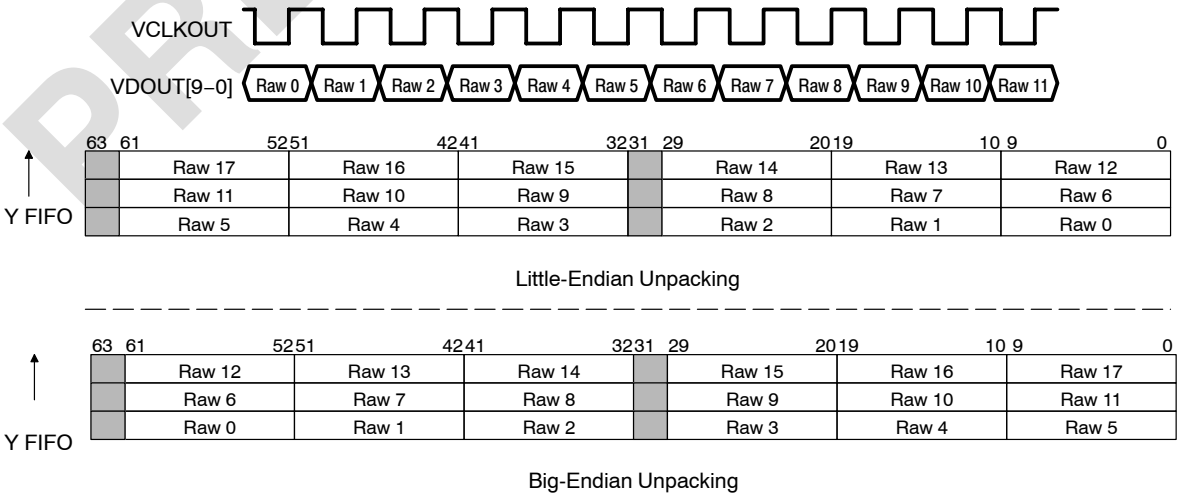
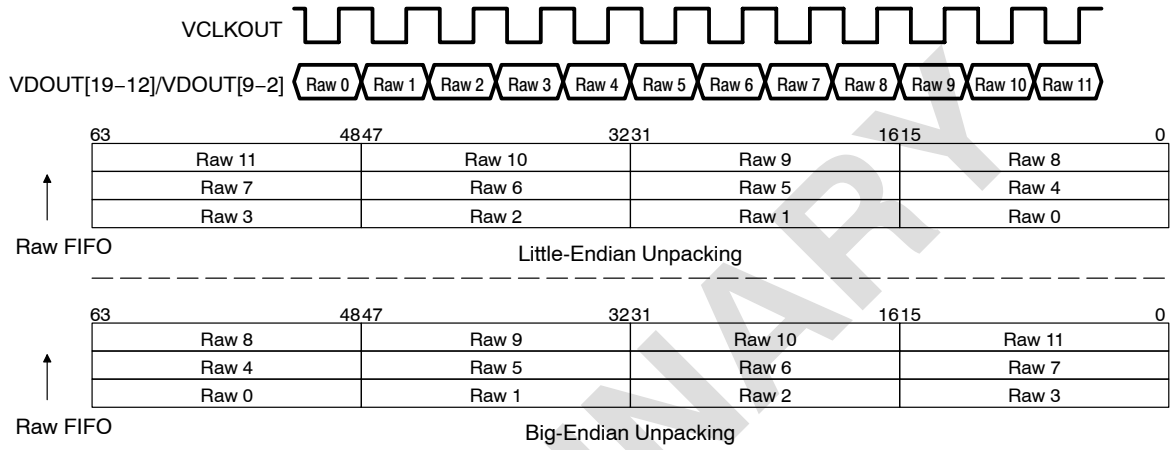


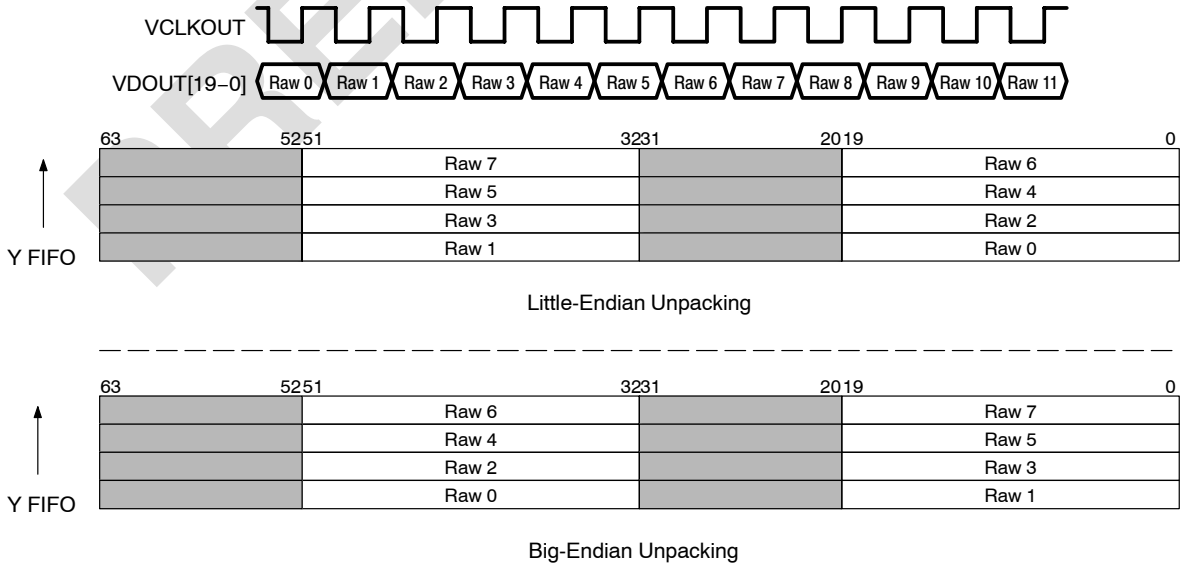
Figure 4–28 shows the 16-bit raw mode. Two samples are unpacked from each word of the FIFO.

Figure 4–28. 16-Bit Raw FIFO Unpacking



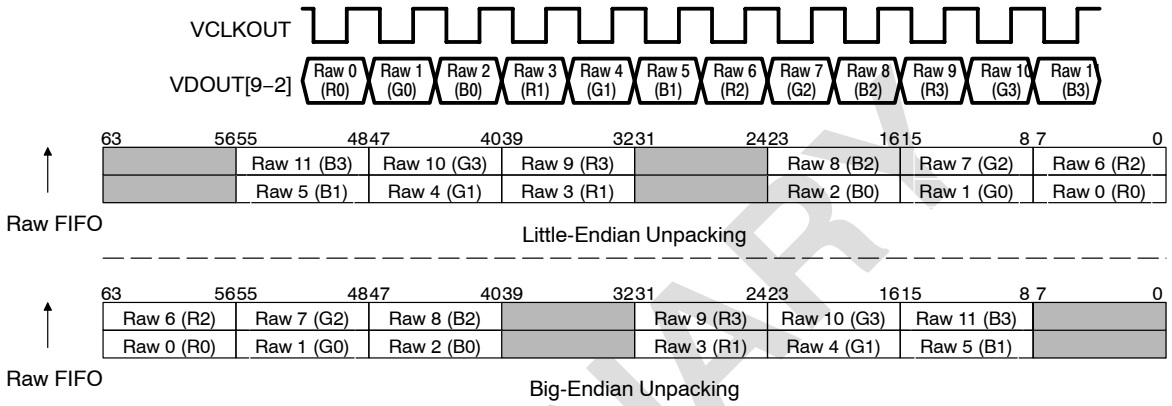
The FIFO unpacking for 20-bit raw format is shown in Figure 4–29. One sample is unpacked from each word of the FIFO.

Figure 4–29. 20-Bit Raw FIFO Unpacking



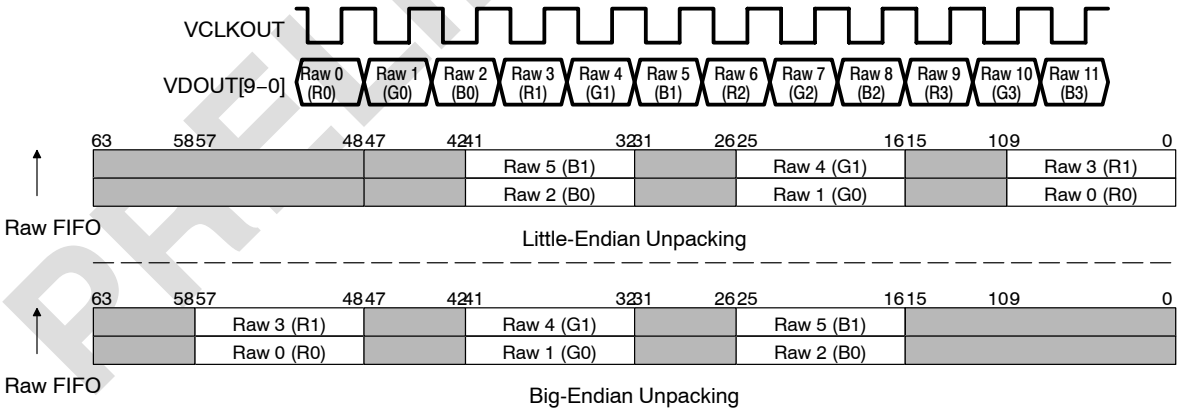
In 8-bit raw ¾ mode, three samples are unpacked from the FIFO and the remaining byte is ignored. This is shown in Figure 4–30.

Figure 4–30. 8-Bit Raw 3/4 FIFO Unpacking



In 10-bit raw ¾ mode, three samples are unpacked from every doubleword of the FIFO and the remaining halfword is ignored. This is shown in Figure 4–31.

Figure 4–31. 10-Bit Raw 3/4 FIFO Unpacking



4.7 Video Display Field and Frame Operation

As a video source, the video port always outputs entire frames of data and transmits continuous video control signals. Depending on the DMA structure, however, the video port may need to interrupt the DSP on a field or frame basis to allow it to update video port registers or DMA parameters. To achieve this, the video port provides programmable control over the display process.

4.7.1 Display Determination and Notification

Noncontinuous display and single frame display are currently not supported at this time.

In order to accommodate various display scenarios, DMA structures, and processing flows, the video port employs a flexible display and DSP notification system. This is programmed using the CON, FRAME, DF1, and DF2 bits in VDCTL.

The CON bit controls the display of multiple fields or frames. When CON = 1, continuous display is enabled, the video port displays outgoing fields (assuming the VDEN bit is set) without the need for DSP interaction. It relies on a single display buffer in memory or on a DMA structure with circular buffering capability to service the display FIFOs. When CON = 0, continuous display is disabled, the video port sets a field or frame display complete bit (F1D, F2D, or FRMD) in VDSTAT upon the display of each field as determined by the state of the other display control bits (FRAME, CD1, and CD2). Once the display complete bit is set, the processor must update the appropriate DMA parameters within the allotted time frame or a subsequent field or frame may output invalid data. In this case, the video port continues to generate DMA requests but it issues a DCNA (display complete not acknowledged) interrupt to indicate that the DMA parameters may not have been updated and bad data is being sent to the video port.

When a field or frame has not been enabled for display, no DMA events are sent for that field or frame. The video port still generates all timings for the field but outputs the default data values rather than data from the display FIFO during the display image window.

The CON, FRAME, DF1, and DF2 bits encode the display operations as listed in Table 4–4.

Table 4−4. Display Operation

VDCTL Bit				
CON	**FRAME**	**DF2**	**DF1**	**Operation**
0	0	0	0	Reserved
0	0	0	1	†Noncontinuous field 1 display. Display only field 1. F1D is set after field 1 display and causes DCMPx to be set. The F1D bit must be cleared by the DSP or a DCNA interrupt occurs. (The DSP has the entire field 2 time to clear F1D before next field 1 begins.) Can also be used for single progressive frame display (internal timing codes only). (The DSP has vertical blanking time to clear F1D before next frame begins.)
0	0	1	0	†Noncontinuous field 2 display. Display only field 2. F2D is set after field 2 display and causes DCMPx to be set. The F2D bit must be cleared by the DSP or a DCNA interrupt occurs. (The DSP has the entire field 1 time to clear F2D before next field 2 begins.)
0	0	1	1	†Noncontinuous field 1 and field 2 display. Display both fields. F1D is set after field 1 display and causes DCMPx to be set. The F1D bit must be cleared by the DSP before the next field 1 display or a DCNA interrupt occurs. (The DSP has the entire field 2 time to clear F1D before next field 1 begins.) F2D is set after field 2 display and also causes DCMPx to be set. The F2D bit must be cleared by the DSP before the next field 2 display or a DCNA interrupt occurs. (The DSP has the entire field 1 time to clear F2D before next field 2 begins.)
0	1	0	0	†Noncontinuous frame display. Display both fields. FRMD is set after field 2 display and causes DCMPx to be set. A DCNA interrupt occurs upon completion of the next frame unless the FRMD bit is cleared. (The DSP has the entire next frame time to clear FRMD.)
0	1	0	1	†Noncontinuous progressive frame display. Display field 1. FRMD is set after field 1 display and causes DCMPx to be set. A DCNA interrupt occurs upon completion of the next frame unless the FRMD bit is cleared. (The DSP has the entire next frame time to clear FRMD.) If external control signals are used, they must follow progressive format.
0	1	1	0	Reserved
0	1	1	1	†Single frame display. Display both fields. FRMD is set after field 2 display and causes DCMPx to be set. A DCNA interrupt occurs unless the FRMD bit is cleared. (The DSP has the field 2 to field 1 vertical blanking time to clear FRMD.)
1	0	0	0	Reserved

† Noncontinuous display and single frame display are currently not supported at this time.

Table 4−4. Display Operation (Continued)

VDCTL Bit				
CON	**FRAME**	**DF2**	**DF1**	**Operation**
1	0	0	1	Continuous field 1 display. Display only field 1. F1D is set after field 1 display and causes DCMPx to be set (DCMPx interrupt can be disabled). No DCNA interrupt occurs, regardless of the state of F1D.
1	0	1	0	Continuous field 2 display. Display only field 2. F2D is set after field 2 display and causes DCMPx to be set (DCMPx interrupt can be disabled). No DCNA interrupt occurs, regardless of the state of F2D.
1	0	1	1	Reserved
1	1	0	0	Continuous frame display. Display both fields. FRMD is set after field 2 display and causes DCMPx to be set (DCMPx interrupt can be disabled. No DCNA interrupt occurs, regardless of the state of FRMD.
1	1	0	1	Continuous progressive frame display. Display field 1. FRMD is set after field 1 display and causes DCMPx to be set (DCMPx interrupt can be disabled). No DCNA interrupt occurs, regardless of the state of FRMD. (Functions identically to continuous field 1 display mode except the FRMD bit is used instead of the F1D bit.) If external control signals are used, they must follow progressive format.
1	1	1	0	Reserved
1	1	1	1	Reserved

† Noncontinuous display and single frame display are currently not supported at this time.

4.7.2 Video Display Event Generation

The display FIFOs are filled using DMAs as requested by the video port DMA events. The VDTHRLD value indicates the level at which the FIFO has enough room to receive another DMA block of data. Depending on the size of the DMA, the FIFO may have room for multiple transfers before reaching the VDTHRLD level. Once the threshold is reached, another DMA event is generated as soon as the FIFO again falls below the VDTHRLD level.

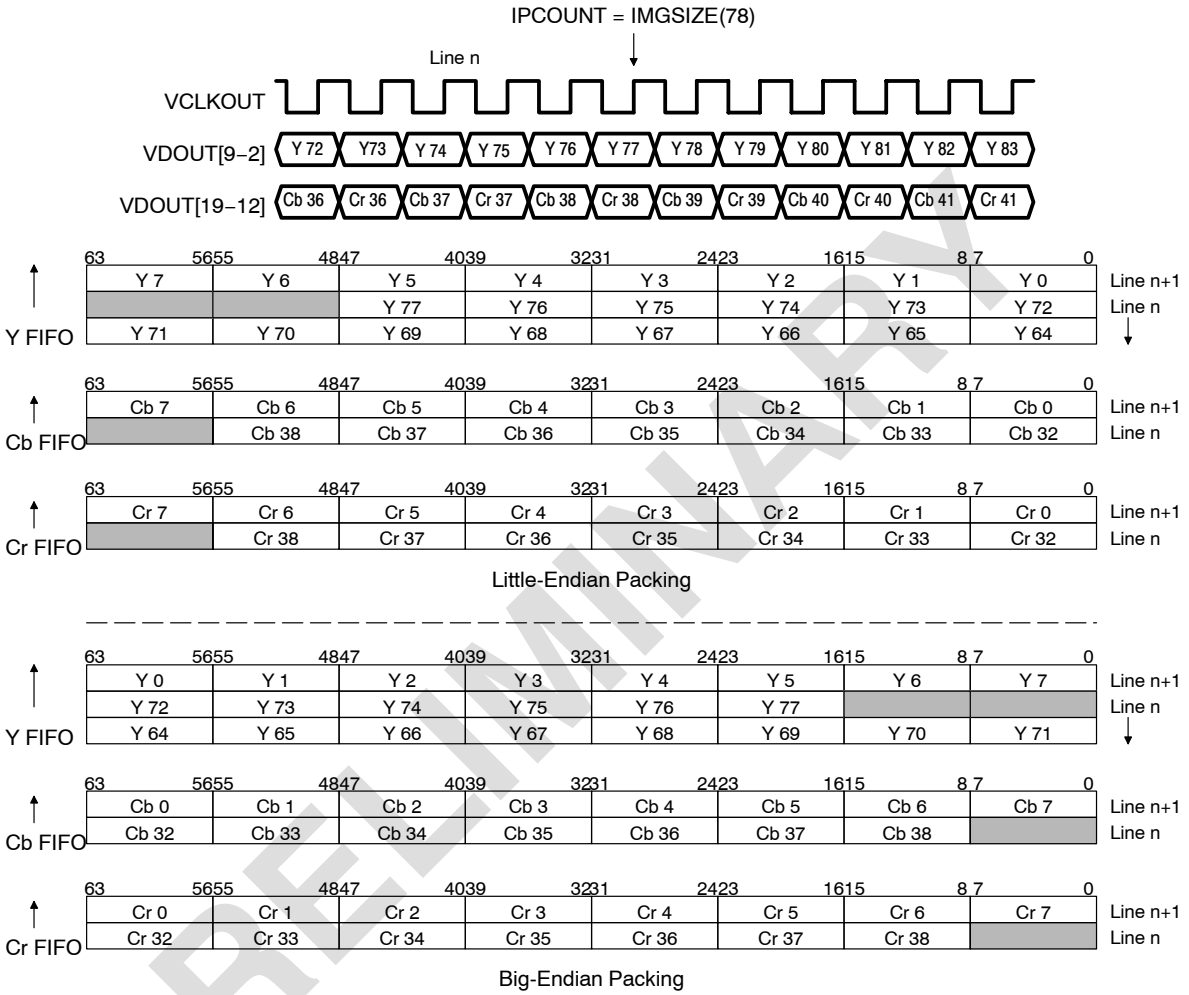
Once an entire field worth of data has been sent to the FIFO, the video port may need to stop generating events in order to allow the DSP to change DMA. Since display may not yet be complete (the FIFO continues to empty after falling below VDTHRLD), a display event counter (DEVTCT) is provided to track the number of requested YEVT events. The counter is loaded with the number of events needed in a display field (DISPEVT1 or DISPEVT2) and is decremented each time the event is requested. Once the counter reaches 0, further display events are inhibited. At the start of the next field, DEVTCT is reloaded and display events are reenabled.

4.8 Display Line Boundary Conditions

In order to simplify DMA transfers, FIFO doublewords do not contain data from more than one display line. This means that a FIFO read must be performed whenever 8-bytes have been output or when the line complete condition (IPCOUNT = IMGHSIZE) occurs. Thus, every display line begins on a doubleword boundary and non-doubleword length lines are truncated at the end. An example is shown in Figure 4−32.

In Figure 4−32 (8-bit Y/C mode), the line length is not a doubleword. When the condition IPCOUNT = IMGHSIZE occurs, the remaining bytes of the FIFO doubleword are ignored and the output switches to the default output value (or the EAV code followed by blanking, if the end of the active video line has been reached). The next display line then begins in the next FIFO location at byte 0. This operation extends to all display modes.

Figure 4−32. Display Line Boundary Example



4.9 Display Timing Examples

The following are examples of display output for several modes of operation.

4.9.1 Interlaced BT.656 Timing Example

This section shows an example of BT.656 display output for a 704 × 408 interlaced output image as might be generated by MPEG decoding.

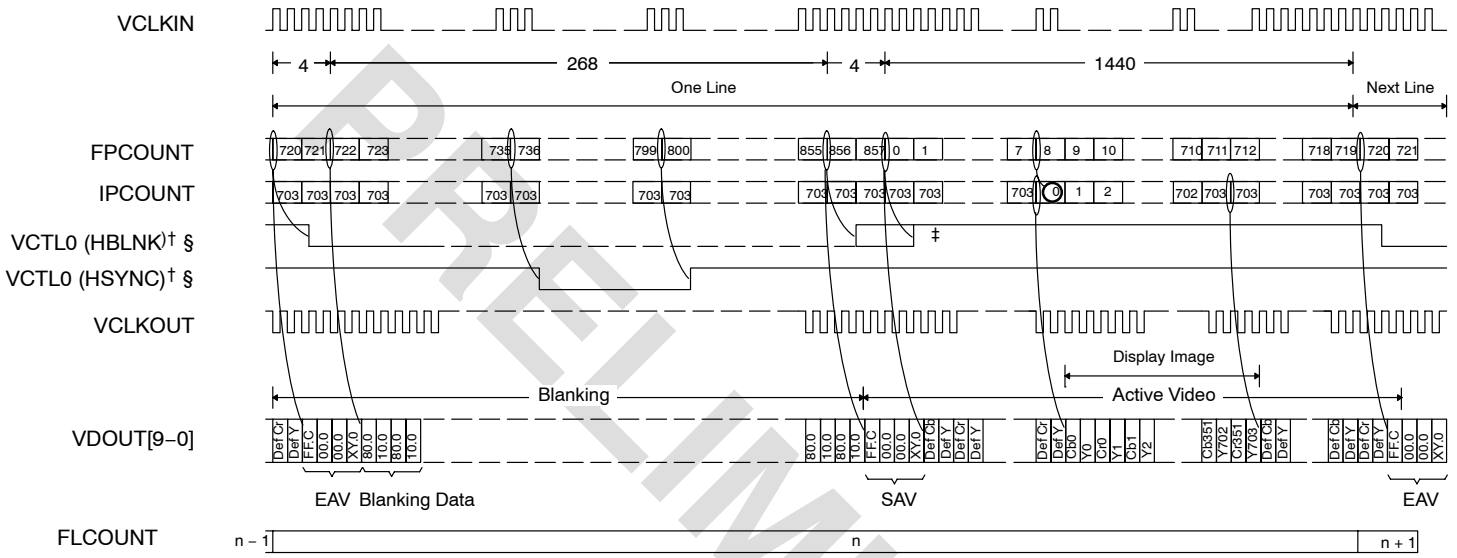
The horizontal output timing is shown in Figure 4–33. This diagram assumes that there is a two VCLK pipeline delay between the internal counter changing and the output on external pins. The actual delay can be longer or shorter as long as it is consistent within any display mode. The BT.656 active line is 720-pixels wide. Figure 4–33 shows the 704-pixel image window centered in the screen that results in an IMGHOFF*x* of 8 pixels.

The HBLNK and HSYNC signals are shown as they would be output for active-low operation. Note that only one of the two signals is actually available externally. The HBLNK inactive edge occurs either on sample 856 coincident with the start of SAV or on sample 0 (after SAV) if the HBDLA bit is set. For true BT.656 operation, neither HBLNK nor HSYNC would be used.

The IPCOUNT operation follows the description in section 4.1.2. IPCOUNT resets to 0 at the first displayed pixel (FPCOUNT = IMGHOFF*x*) and stops counting at the last displayed pixel (IPCOUNT = IMGHSIZE*x*). The operation during nondisplay time is not a requirement, it could continue counting until the next FPCOUNT = IMGHOFF*x* point or it could reset immediately after IMGHSIZE*x* or when FPCOUNT is reset.

VDOUT shows the output data and switching between EAV, Blanking Data, SAV, Default Data, and FIFO Data. It is assumed that the DVEN bit in VDCTL is set to enable the default output.

Figure 4–33. BT.656 Interlaced Display Horizontal Timing Example



FRMWIDTH = 858
HBLNKSTART = 720
HBLNKSTOP = 856
IMGHOFF1 = 8
IMGHSIZE1 = 704
IMGHOFF2 = 8
IMGHSIZE2 = 704
HSYNCSTART = 736
HSYNCSTOP = 800

† Assumes VCT0P bit in VPCTL is set to 1 (active-low output). HSYNC output when VCTL0S bit in VDCTL is set to 00, HBLNK output when VCTL0S bit is set 01.
‡ HBLNK operation when HBDLA bit in VDHBLNK is set to 1.
§ Diagram assumes a two VCLK pipeline delay between internal counters and output signals.

The interlaced BT.656 vertical output timing is shown in Figure 4−34. The BT.656 active field 1 is 244-lines high and active field 2 is 243-lines high. This example shows the 480-line image window centered in the screen. This results in an IMGVOFF*n* of 3 lines and also results in a nondata line at the end of field 1 due to its extra active line.

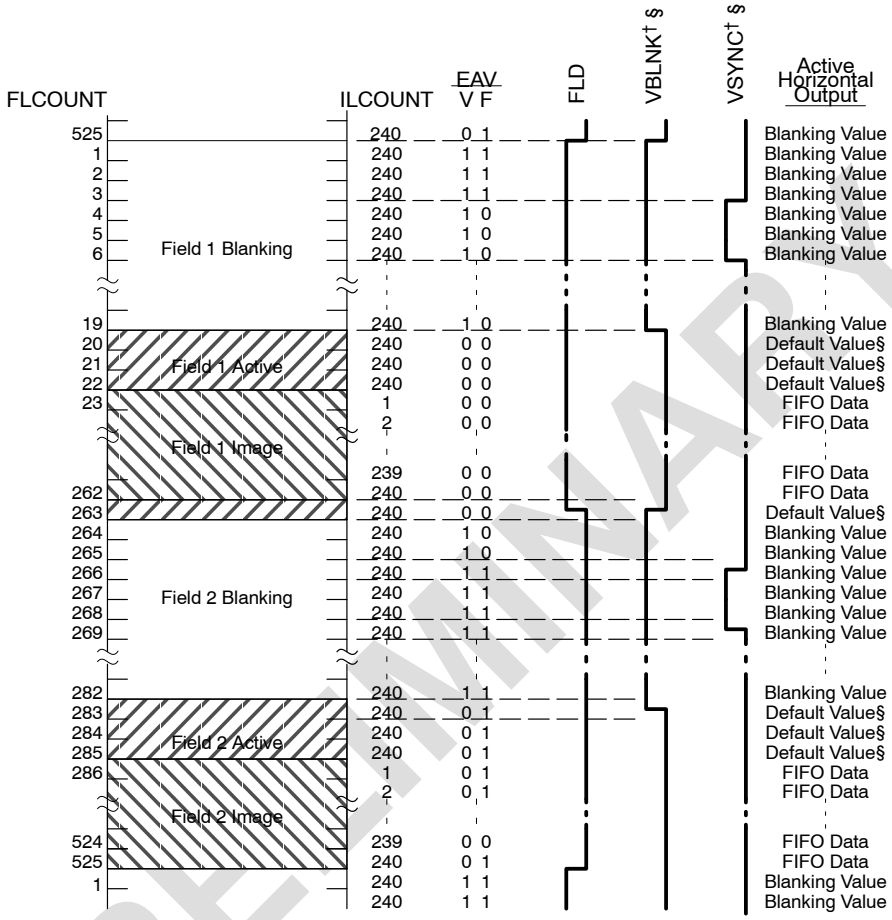
The VBLNK and VSYNC signals are shown as they would be output for active-low operation. Note that only one of the two signals is actually available externally. The VBLNK and VSYNC edges for field 1 occur at the end of an active line so their XSTART/XSTOP values are set to 720 (start of blanking). For field 2, VBLNK and VSYNC edges occur during the middle of the active horizontal line so their XSTART/XSTOP values are set to 360. Note that, from an analog standpoint, vertical blanking begins a half-line before digital blanking so that VBLNKYSTART2 is set to 263 (with VBLNKXSTART2 set to 360) while VBITSET2 is programmed to 264. For true BT.656 operation, neither VBLNK nor VSYNC would be used.

The FLD output is setup to transition at the start of each analog field (start of vertical blanking). Since EAV[F] transitions on lines 4 and 266, this requires programming FBITCLR to 4, FBITSET to 266, FLD1YSTART to 1, and FLD2YSTART to 263. Note that FLD2XSTRT is 360 so that the field indicator output changes halfway through the line.

The ILCOUNT operation follows the description in section 4.1.2. ILCOUNT resets to 1 at the first displayed line (FLCOUNT = VBLNKSTOP*x* + IMGVOFF*x*) and stops counting at the last displayed pixel (IPCOUNT = IMGVSIZE*x*). The operation during nondisplay time is not a requirement, it could continue counting until the next FLCOUNT = VBLNKSTOP*x* + IMGVOFF*x* point or it could reset immediately after IMGVSIZE*x* or when FLCOUNT is reset.

The active horizontal output column shows the output data during the active portion of the horizontal line. It is assumed that the DVEN bit in VDCTL is set to enable the default output.

Figure 4−34. BT.656 Interlaced Display Vertical Timing Example



IMGVOFF1 = 3
IMGVSIZE1 = 240
IMGVOFF2 = 3
IMGVSIZE2 = 240
FRMHEIGHT = 525
VBITSET1 = 1
VBITCLR1 = 20
VBITSET2 = 264
VBITCLR2 = 283

VBLNKXSTART1 = 720
VBLNKYSTART1 = 1
VBLNKXSTOP1 = 720
VBLNKYSTOP1 = 20
VBLNKXSTART2 = 360
VBLNKYSTART2 = 263
VBLNKXSTOP2 = 360
VBLNKYSTOP2 = 283

VSYNCXSTART1 = 720
VSYNCYSTART1 = 4
VSYNCXSTOP1 = 720
VSYNCYSTOP1 = 7
VSYNCXSTART2 = 360
VSYNCYSTART2 = 266
VSYNCXSTOP2 = 360
VSYNCYSTOP2 = 269

FLD1XSTART = 720
FLD1YSTART = 1
FLD2XSTART = 360
FLD2YSTART = 263

FBITSET = 266
FBITCLR = 4

† Assumes VCT1P bit in VPCTL is set to 1 (active-low output). VSYNC output when VCTL1S bit in VDCTL is set to 00, VBLNK output when VCTL1S bit is set 01.
§ If DVEN bit in VDCTL is set to 1; otherwise, blanking value is output

4.9.2 Interlaced Raw Display Example

This section shows an example of raw display output for the same 704 × 408 interlaced image.

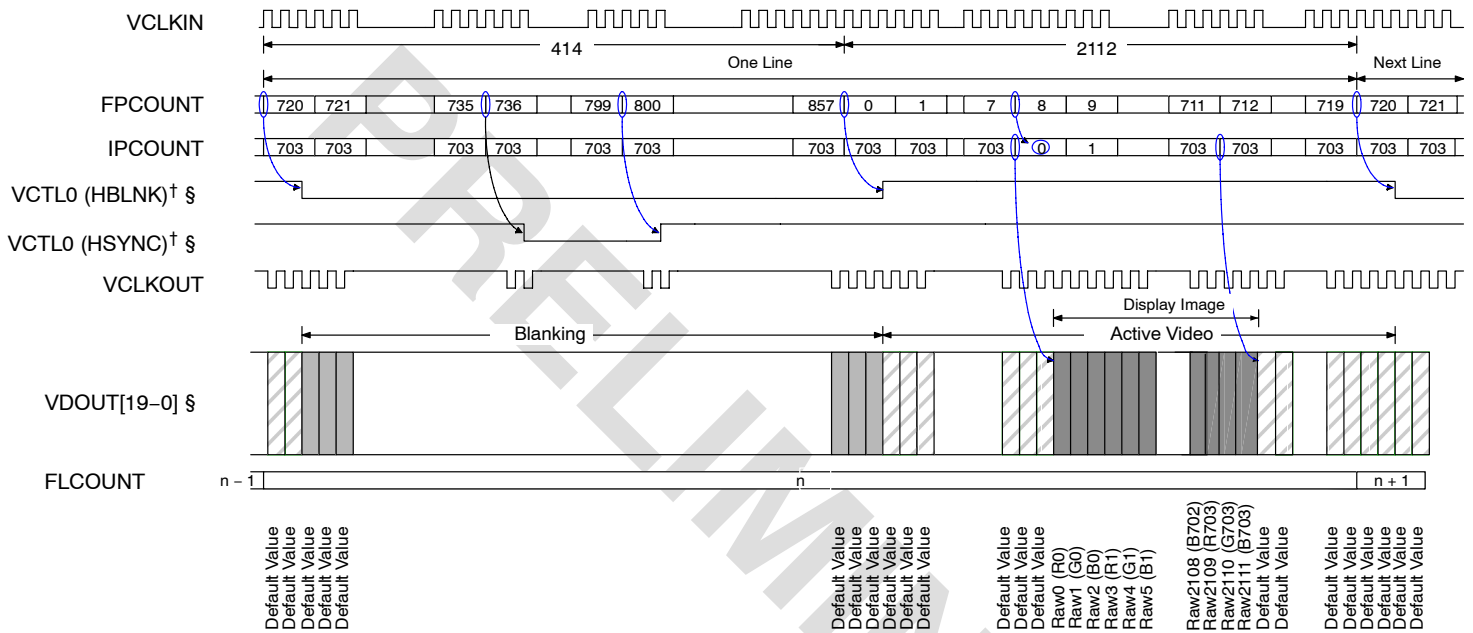
The horizontal output timing is shown in Figure 4−35. This diagram assumes that there is a two VCLK pipeline delay between the internal counter changing and the output on external pins. The actual delay can be longer or shorter as long as it is consistent within any display mode. The active line is 720-pixels wide. Figure 4−35 shows the 704-pixel image window centered in the screen that results in an IMGHOFF*x* of 8 pixels.

The HBLNK and HSYNC signals are shown as they would be output for active-low operation. Note that only one of the two signals is actually available externally. The HBLNK inactive edge occurs on sample 0.

The IPCOUNT operation follows the description in section 4.1.2. IPCOUNT resets to 0 at the first displayed pixel (FPCOUNT = IMGHOFF*x*) and stops counting at the last displayed pixel (IPCOUNT = IMGHSIZE*x*). Both the IPCOUNT and FPCOUNT counters increment on every third VCLKIN rising edge, as programmed by the INCPIX bits in VDTHRLD with a value of 3.

VDOUT shows the output data and switching between Default Data, and FIFO Data. Three values are output sequentially on VDOUT for each pixel count. Note that the default value is output during both the blanking and nondisplay image active video regions.

Figure 4–35. Raw Interlaced Display Horizontal Timing Example



FRMWIDTH = 858	IMGHOFF1 = 8	HSYNCSTART = 736
HBLNKSTART = 720	IMGHSIZE1 = 704	HSYNCSTOP = 800
HBLNKSTOP = 0	IMGHOFF2 = 8	
	IMGHSIZE2 = 704	INCPIX = 3

† Assumes VCT0P bit in VPCTL is set to 1 (active-low output). HSYNC output when VCTL0S bit in VDCTL is set to 00, HBLNK output when VCTL0S bit is set 01.

§ Diagram assumes a two VCLK pipeline delay between internal counters and output signals.

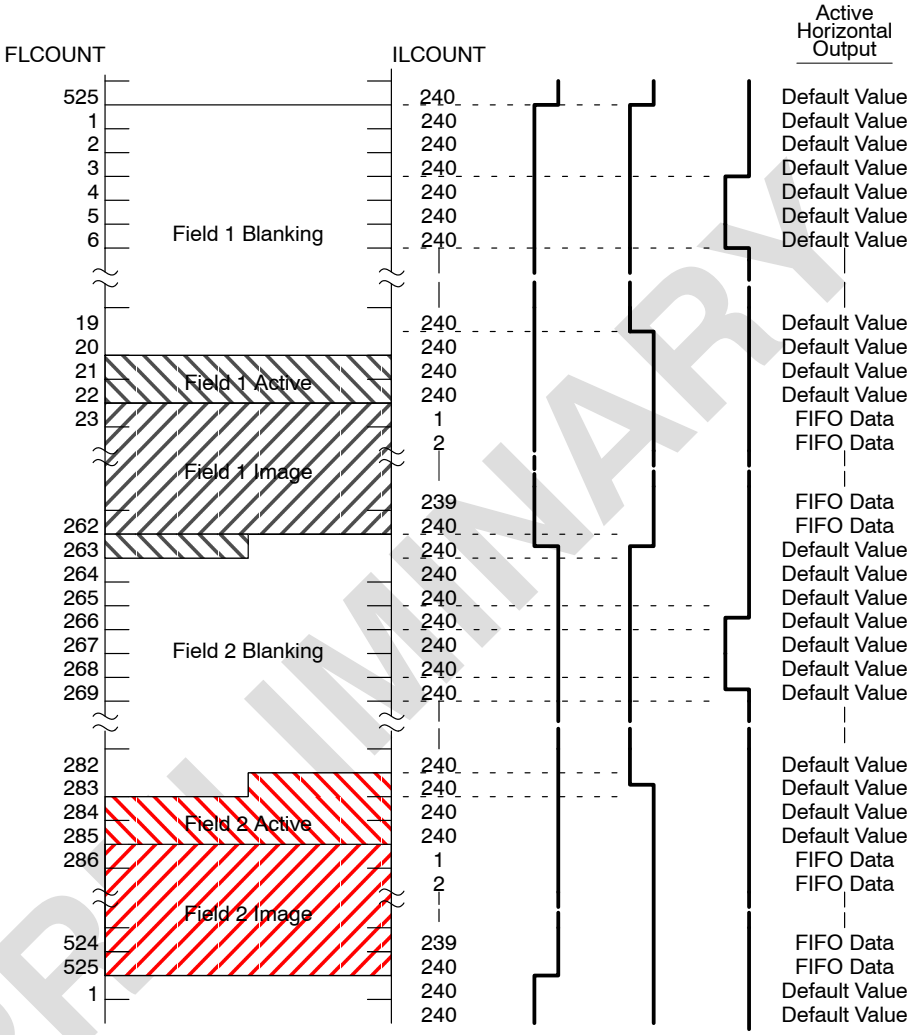
The vertical output timing for raw mode is shown in Figure 4–36. This example outputs the same 480-line window. Note that the raw display mode is typically noninterlaced for output to a monitor. This example shows the more complex interlaced case. The active field 1 is 242.5-lines high and active field 2 is 242.5-lines high. This example shows the 480-line image window centered in the screen. This results in an IMGVOFF1 of 2 lines and an IMGVOFF2 of 3 lines and also results in a nondata half-line at the end of field 1 and at the beginning of field 2 due to their noninteger line lengths.

The VBLNK and VSYNC signals are shown as they would be output for active-low operation. Note that only one of the two signals is actually available externally. The VBLNK and VSYNC edges for field 1 occur at the end of an active line so their XSTART/XSTOP values are set to 720 (start of blanking). For field 2, VBLNK and VSYNC edges occur during the middle of the active horizontal line so their XSTART/XSTOP values are set to 360.

The FLD output is setup to transition at the start of each analog field (start of vertical blanking). There is no EAV[F] bit in raw mode, so FLD1YSTRT is set to 1, FLD2YSTART is set to 263, FBITCLR and FBITSET are ignored. Note that FLD2XSTRT is 360 so that the field indicator output changes halfway through the line.

The active horizontal output column shows the output data during the active portion of the horizontal line. Note that in raw mode there is no blanking data value so the default value is output for the active portion of all nonimage window lines.

Figure 4–36. Raw Interlaced Display Vertical Timing Example



IMGVOFF1 = 2
IMGVSIZE1 = 240
IMGVOFF2 = 3
IMGVSIZE2 = 240
FRMHEIGHT = 525
VBITSET1 = n/a
VBITCLR1 = n/a
VBITSET2 = n/a
VBITCLR2 = n/a

VBLNKXSTART1 = 720
VBLNKYSTART1 = 1
VBLNKXSTOP1 = 720
VBLNKYSTOP1 = 21
VBLNKXSTART2 = 360
VBLNKYSTART2 = 263
VBLNKXSTOP2 = 360
VBLNKYSTOP2 = 283

VSYNCXSTART1 = 720
VSYNCYSTART1 = 4
VSYNCXSTOP1 = 720
VSYNCYSTOP1 = 7
VSYNCXSTART2 = 360
VSYNCYSTART2 = 266
VSYNCXSTOP2 = 360
VSYNCYSTOP2 = 269

FLD1XSTART = 720
FLD1YSTART = 1
FLD2XSTART = 360
FLD2YSTART = 263

FBITSET = n/a
FBITCLR = n/a

† Assumes VCT1P bit in VPCTL is set to 1 (active-low output). VSYNC output when VCTL1S bit in VDCTL is set to 00, VBLNK output when VCTL1S bit is set 01.

4.9.3 Y/C Progressive Display Example

This section shows an example of progressive display operation. The output format follows SMPTE 296M-2001 specifications for a 1280 × 720/60 system. The example is for a 1264 × 716 progressive output image.

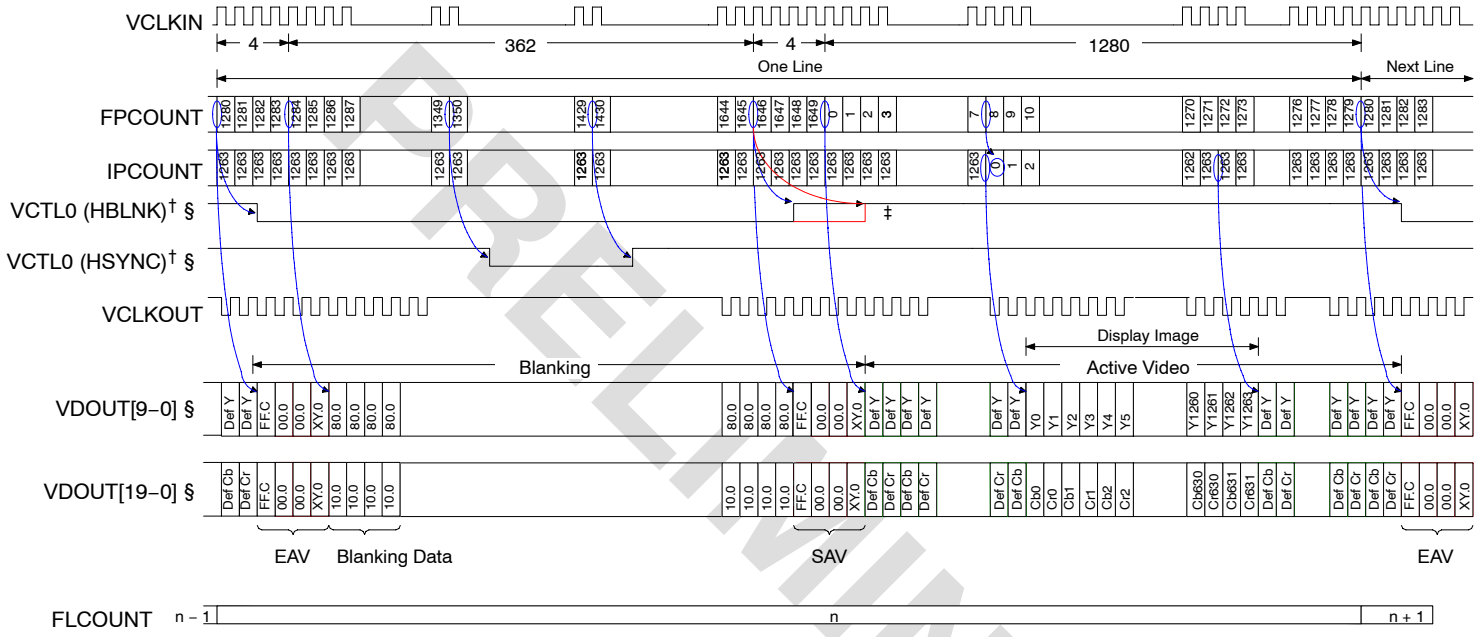
The horizontal output timing is shown in Figure 4–37. This diagram assumes that there is a two VCLK pipeline delay between the internal counter changing and the output on external pins. The actual delay can be longer or shorter as long as it is consistent within any display mode. The SMPTE 296M 60-Hz active line is 1650-pixels wide. Figure 4–37 shows the 1264-pixel image window centered in the screen that results in an IMGHOFF*x* of 8 pixels.

The HBLNK and HSYNC signals are shown as they would be output for active-low operation. Note that only one of the two signals is actually available externally. The HBLNK inactive edge occurs either on sample 1646 coincident with the start of SAV or on sample 0 (after SAV) if the HBDLA bit is set. For true SMPTE 296M operation, neither HBLNK nor HSYNC would be used.

The IPCOUNT operation follows the description in section 4.1.2. IPCOUNT resets to 0 at the first displayed pixel (FPCOUNT = IMGHOFF*x*) and stops counting at the last displayed pixel (IPCOUNT = IMGHSIZE*x*). The operation during nondisplay time is not a requirement, it could continue counting until the next FPCOUNT = IMGHOFF*x* point or it could reset immediately after IMGHSIZE*x* or when FPCOUNT is reset.

VDOUT shows the output data and switching between EAV, Blanking Data, SAV, Default Data, and FIFO Data. It is assumed that the DVEN bit in VDCTL is set to enable the default output.

Figure 4–37. Y/C Progressive Display Horizontal Timing Example



FRMWIDTH = 1650
HBLNKSTART = 1280
HBLNKSTOP = 1646

IMGHOFF1 = 8
IMGHSIZE1 = 1264
IMGHOFF2 = n/a
IMGHSIZE2 = n/a

HSYNCSTART = 1350
HSYNCSTOP = 1430

† Assumes VCT0P bit in VPCTL is set to 1 (active-low output). HSYNC output when VCTL0S bit in VDCTL is set to 00, HBLNK output when VCTL0S bit is set 01.
‡ HBLNK operation when HBDLA bit in VDHBLNK is set to 1.
§ Diagram assumes a two VCLK pipeline delay between internal counters and output signals.

The vertical output timing is shown in Figure 4–38. SMPTE 296M has a single active field 1 that is 720-lines high. This example shows the 716-line image window with an IMGVOFF*n* of 3 lines and also results in a nondata line at the end of the field.

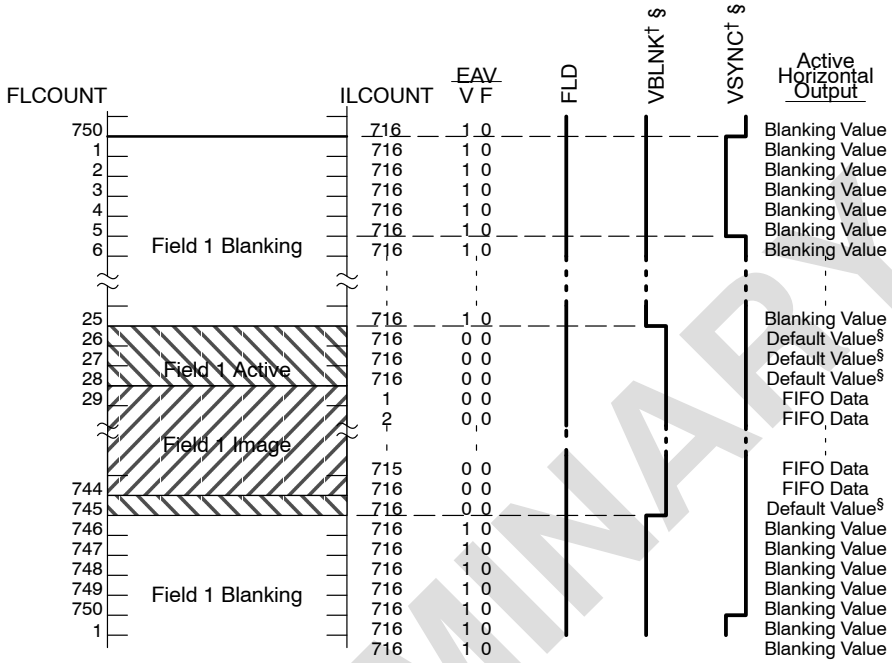
The VBLNK and VSYNC signals are shown as they would be output for active-low operation. Note that only one of the two signals is actually available externally. The VBLNK and VSYNC edges occur at the end of an active line so their XSTART/XSTOP values are set to 1280 (start of blanking). The field 2 vertical timing start and stop registers are programmed to a value greater than 750. Since this value is never reached by FLCOUNT, no extra VBLNK or VSYNC transitions occur. For true SMPTE 296M operation, neither VBLNK nor VSYNC would be used.

The FLD output is setup to transition low at the start of each frame. Since the FLD2YSTART value is never reached by FLCOUNT, the FLD output remains always low.

The ILCOUNT operation follows the description in section 4.1.2. ILCOUNT resets to 1 at the first displayed line (FLCOUNT = VBLNKSTOP*x* + IMGVOFF*n*) and stops counting at the last displayed pixel (IPCOUNT = IMGVSIZE*x*). The operation during nondisplay time is not a requirement, it could continue counting until the next FLCOUNT = VBLNKSTOP*x* + IMGVOFF*n* point or it could reset immediately after IMGVSIZE*x* or when FLCOUNT is reset.

The active horizontal output column shows the output data during the active portion of the horizontal line. It is assumed that the DVEN bit in VDCTL is set to enable the default output.

Figure 4−38. Y/C Progressive Display Vertical Timing Example



IMGVOFF1 = 3	VBLNKXSTART1 = 1280	VSYNCXSTART1 = 1280	FLD1XSTART = n/a
IMGVSIZE1 = 716	VBLNKYSTART1 = 746	VSYNCYSTART1 = 1	FLD1YSTART = 1
IMGVOFF2 = n/a	VBLNKXSTOP1 = 1280	VSYNCXSTOP1 = 1280	FLD2XSTART = n/a
IMGVSIZE2 = n/a	VBLNKYSTOP1 = 26	VSYNCYSTOP1 = 6	FLD2YSTART = > 750
FRMHEIGHT = 750	VBLNKXSTART2 = n/a	VSYNCXSTART2 = n/a	
VBITSET1 = 746	VBLNKYSTART2 = > 750	VSYNCYSTART2 = > 750	FBITSET = 1
VBITCLR1 = 26	VBLNKXSTOP2 = n/a	VSYNCXSTOP2 = n/a	FBITCLR = > 750
VBITSET2 = n/a	VBLNKYSTOP2 = > 750	VSYNCYSTOP2 = > 750	
VBITCLR2 = n/a			

† Assumes VCT1P bit in VPCTL is set to 1 (active-low output). VSYNC output when VCTL1S bit in VDCTL is set to 00, VBLNK output when VCTL1S bit is set 01.

§ If DVEN bit in VDCTL is set to 1; otherwise, blanking value is output

4.10 Displaying Video in BT.656 or Y/C Mode

In order to display video in the BT.656 or Y/C format, the following steps are needed:

1) Set the frame size in VDFRMSZ. Set the number of lines per frame (FRMHIGHT) and the number of pixels per line (FRMWIDTH).

2) Set the horizontal blanking in VDHBLNK. Specify the frame pixel counter value where horizontal blanking starts (HBLNKSTART) and pixel location where horizontal blanking stops (HBLNKSTOP).

3) Set the V bit timing for field 1 in VDVBIT1. Specify the line where the V bit is set (VBITSET1) and the line where the V bit is cleared (VBITCLR1).

4) If external VBLNK signal is needed, set the VBLNK start for field 1 in VDVBLKS1. Specify the frame line (VBLNKYSTART1) and frame pixel counter (VBLNKXSTART1) values for the pixel where VBLNK goes active for field 1. Set the VBLNK end for field 1 in VDVBLKE1. Specify the frame line (VBLNKYSTOP1) and frame pixel counter (VBLNKXSTOP1) values for the pixel where VBLNK goes inactive for field 1.

5) Set the V bit timing for field 2 in VDVBIT2. Specify the line where the V bit is set (VBITSET2) and the line where the V bit is cleared (VBITCLR2).

6) If external VBLNK signal is needed, set the VBLNK start for field 2 in VDVBLKS2. Specify the frame line (VBLNKYSTART2) and frame pixel counter (VBLNKXSTART2) values for the pixel where VBLNK goes active for field 2. Set the VBLNK end for field 2 in VDVBLKE2. Specify the frame line (VBLNKYSTOP2) and frame pixel counter (VBLNKXSTOP2) values for the pixel where VBLNK goes inactive for field 2.

7) Set VDIMGSZ*n*. Adjust the displayed image size by setting the HSIZE and VSIZE bits.

8) Set VDIMOFF. Adjust the displayed image offset within the active video area (by setting HOFFSET and VOFFSET).

9) Set the F bit timing in VDFBIT. Specify the line where the F bit is cleared (FBITCLR) and the line where the F bit is set (FBITSET).

10) If external FLD output is required, set the video display field 1 timing. Specify the line and pixel where FLD goes inactive (VDFLDT1). Set the video display field 2 timing. Specify the line and pixel where FLD goes active (VDFLDT2).

11) Set VDCLIP. Default values for video clipping are 16 for the lower clipping, 235 for the higher clipping of the Y values, and 240 for the higher clipping of the Cb and Cr values.

12) Configure a DMA to move data from the Y buffer in the DSP memory to YDSTA (memory-mapped Y display FIFO). The transfers should be triggered by the YEVT.

13) Configure a DMA to move data from the Cb buffer in the DSP memory to CBDST (memory-mapped Cb display FIFO). The transfers should be triggered by the CbEVT. The size of the transfers should be set to ½ the Y transfer size.

14) Configure a DMA to move data from the Cr buffer in the DSP memory to CRDST (memory-mapped Cr display FIFO). The transfers should be triggered by the CrEVT. The size of the transfers should be set to ½ the Y transfer size.

15) Set DISPEVT1 and DISPEVT2 bits in VDDISPEVT. Event count is total doublewords per field divided by total doublewords per Y DMA.

16) Write to VPIE to enable underrun (DUND) and display complete (DCMP) interrupts, if desired.

17) Write to VDTHRLD to set the display FIFO threshold (VDTHRLD bits).

18) Write to VDCTL to:

- ❏ Set display mode (DMODE = 00x for BT.656 output, 10x for Y/C output).
- ❏ Set desired field/frame operation (CON, FRAME, DF1, DF2 bits).
- ❏ Select control outputs (VCTL0S, VCTL1S, VCTL2S bits) or external sync inputs (HXS, VXS, FXS bits).
- ❏ Enable scaling (SCALE and RESMPL bits), if desired and in 8-bit mode.
- ❏ Select 10-bit unpacking mode (DPK bit), if appropriate.
- ❏ Set VDEN bit to enable the display.

19) Wait for 2 or more frame times, to allow the display counters and control signals to become properly synchronized.

20) Write to VDCTL to clear the BLKDIS bit.

21) Display is enabled at the start of the first frame after BLKDIS = 0 and begins with the first selected field. DMA events are generated as triggered by VDTHRLD and the DEVTCT counter. When a selected field has been displayed (FLCOUNT = FRMHEIGHT and FPCOUNT = FRMWIDTH), the appropriate F1D, F2D, or FRMD bits are set and cause the DCMP bit in VPIS to be set. This generates a DSP interrupt, if the DCMP bit is enabled in VPIE.

22) If continuous display is enabled, the video port begins displaying again at the start of the next field or frame. If noncontinuous field 1 and field 2 or frame display is enabled, the next field or frame is displayed, during which the DSP must clear the appropriate completion status bit or a DCNA interrupt occurs and incorrect data may be output.

4.11 Displaying Video in Raw Data Mode

In order to display video in the raw data mode, the following steps are needed:

1) Set the frame size in VDFRMSZ. Set the number of lines per frame (FRMHIGHT) and the number of pixels per line (FRMWIDTH).
2) Set the horizontal blanking in VDHBLNK. Specify the frame pixel counter value where horizontal blanking starts (HBLNKSTART) and pixel location where horizontal blanking stops (HBLNKSTOP).
3) Set the vertical blanking start for field 1 in VDVBLKS1. Specify the frame line (VBLNKYSTART1) and frame pixel counter (VBLNKXSTART1) values for the pixel where vertical blanking starts for field 1.
4) Set the vertical blanking end for field 1 in VDVBLKE1. Specify the frame line (VBLNKYSTOP1) and frame pixel counter (VBLNKXSTOP1) values for the pixel where vertical blanking ends for field 1.
5) Set the vertical blanking start for field 2 in VDVBLKS2. Specify the frame line (VBLNKYSTART2) and frame pixel counter (VBLNKXSTART2) values for the pixel where vertical blanking starts for field 2.
6) Set the vertical blanking end for field 2 in VDVBLKE2. Specify the frame line (VBLNKYSTOP2) and frame pixel counter (VBLNKXSTOP2) values for the pixel where vertical blanking ends for field 2.
7) Set the vertical synchronization start for field 1 in VDVSYNS1. Specify the frame line (VSYNCYSTART1) and frame pixel counter (VSYNCXSTART1) values for the pixel where vertical synchronization starts for field 1.
8) Set the vertical synchronization end for field 1 in VDVSYNE1. Specify the frame line (VSYNCYSTOP1) and frame pixel counter (VSYNCXSTOP1) values for the pixel where vertical synchronization ends for field 1.
9) Set the vertical synchronization start for field 2 in VDVSYNS2. Specify the frame line (VSYNCYSTART2) and frame pixel counter (VSYNCXSTART2) values for the pixel where vertical synchronization starts for field 2.
10) Set the vertical synchronization end for field 2 in VDVSYNE2. Specify the frame line (VSYNCYSTOP2) and frame pixel counter (VSYNCXSTOP2) values for the pixel where vertical synchronization ends for field 2.

11) Set the horizontal synchronization in VDHSYNC. Specify the frame pixel counter value for a pixel where HSYNC gets asserted (HSYNCYSTART) and width of the HSYNC pulse (HSYNCSTOP) in frame pixel clocks.

12) Set the video display field 1 timing. Specify the first line and pixel of field 1 in VDFLDT1.

13) Set the video display field 2 timing. Specify the first line and pixel of field 2 in VDFLDT2.

14) Configure a DMA to move data from table in the DSP memory to YDSTA (memory-mapped display FIFO). The transfers should be triggered by the YEVT.

15) Set DISPEVT1 and DISPEVT2 bits in VDDISPEVT. Event count is total doublewords per field divided by total doublewords per Y DMA.

16) Write to VPIE to enable underrun (DUND) and display complete (DCMP) interrupts, if desired.

17) Write to VDTHRLD to set the display FIFO threshold (VDTHRLD bits) and the FPCOUNT increment rate (INCPIX bit).

18) Write to VDCTL to:

- ❏ Set display mode (DMODE =01x for 8/10-bit output, 11x for 16/20 bit output).
- ❏ Set desired field/frame operation (CON, FRAME, DF1, DF2 bits).
- ❏ Select control outputs (VCTL0S, VCTL1S, VCTL2S bits) or external sync inputs (HXS, VXS, FXS bits).
- ❏ Select 10-bit unpacking mode (DPK bit), if appropriate.
- ❏ Set VDEN bit to enable the display.

19) Wait for 2 or more frame times, to allow the display counters and control signals to become properly synchronized.

20) Write to VDCTL to clear the BLKDIS bit.

21) Display is enabled at the start of the first frame after BLKDIS = 0 and begins with the first selected field. DMA events are generated as triggered by VDTHRLD and the DEVTCT counter. When a selected field has been displayed (FLCOUNT = FRMHEIGHT and FPCOUNT = FRMWIDTH), the appropriate F1D, F2D, or FRMD bits are set and cause the DCMP bit in VPIS to be set. This generates a DSP interrupt, if the DCMP bit is enabled in VPIE.

22) If continuous display is enabled, the video port begins displaying again at the start of the next field or frame. If noncontinuous field 1 and field 2 or frame display is enabled, the next field or frame is displayed, during which the DSP must clear the appropriate completion status bit or a DCNA interrupt occurs and incorrect data may be output.

4.11.1 Handling Underrun Condition of the Display FIFO

A FIFO underrun occurs when the display FIFO is empty during an active display line because a pending DMA request failed to load the data in time. In case of a FIFO underrun condition, the DUND bit in VPIS is set. This condition initiates an interrupt to the DSP, if the underrun interrupt is enabled (the DUND bit in VPIE is set).

Because video display is typically a continuous real-time output, data output is not halted when a FIFO underrun occurs. (To output a blanking of default value is just as catastrophic to a display as outputting an old data value.) Instead, the FIFO read pointer continues to advance and (old) data continues to be output from the FIFO. This means that if the pending DMA is only slightly late, the data transfer has a chance to catch the FIFO back up to the read pointer and correct data output resumes. If the pending DMA does not complete service within a threshold's worth of output data, then the DMA request sequence is broken and the remainder of the display field is corrupted.

The underrun interrupt routine should set the BLKDIS bit in VDCTL and it should reconfigure the DMA channel settings. Setting the BLKDIS bit flushes the channel display FIFO and prevents channel DMA events from reaching the DMA controller. The DMA must be reconfigured correctly for the next frame display since the current frame transfer failed. The frame line and frame pixel counters continue counting and, from a pin standpoint, the video display module appears to continue to function normally (SAV/EAV codes are generated in the BT.656 or Y/C mode and the default data value is sent out). The BLKDIS bit should then be cleared to reenable DMA events. Clearing the BLKDIS bit does not enable DMA events during the frame where the bit is cleared. Clearing this bit to zero enables DMA events in the frame that follows the frame where the bit is cleared.

4.12 Video Display Registers

The registers for controlling the video display mode of operation are listed in Table 4−5. See the device-specific datasheet for the memory address of these registers.

Table 4−5. Video Display Control Registers

Offset Address†	Acronym	Register Name	Section
200h	VDSTAT	Video Display Status Register	4.12.1
204h	VDCTL	Video Display Control Register	4.12.2
208h	VDFRMSZ	Video Display Frame Size Register	4.12.3
20Ch	VDHBLNK	Video Display Horizontal Blanking Register	4.12.4
210h	VDVBLKS1	Video Display Field 1 Vertical Blanking Start Register	4.12.5
214h	VDVBLKE1	Video Display Field 1 Vertical Blanking End Register	4.12.6
218h	VDVBLKS2	Video Display Field 2 Vertical Blanking Start Register	4.12.7
21Ch	VDVBLKE2	Video Display Field 2 Vertical Blanking End Register	4.12.8
220h	VDIMGOFF1	Video Display Field 1 Image Offset Register	4.12.9
224h	VDIMGSZ1	Video Display Field 1 Image Size Register	4.12.10
228h	VDIMGOFF2	Video Display Field 2 Image Offset Register	4.12.11
22Ch	VDIMGSZ2	Video Display Field 2 Image Size Register	4.12.12
230h	VDFLDT1	Video Display Field 1 Timing Register	4.12.13
234h	VDFLDT2	Video Display Field 2 Timing Register	4.12.14
238h	VDTHRLD	Video Display Threshold Register	4.12.15
23Ch	VDHSYNC	Video Display Horizontal Synchronization Register	4.12.16
240h	VDVSYNS1	Video Display Field 1 Vertical Synchronization Start Register	4.12.17
244h	VDVSYNE1	Video Display Field 1 Vertical Synchronization End Register	4.12.18
248h	VDVSYNS2	Video Display Field 2 Vertical Synchronization Start Register	4.12.19
24Ch	VDVSYNE2	Video Display Field 2 Vertical Synchronization End Register	4.12.20
250h	VDRELOAD	Video Display Counter Reload Register	4.12.21

† The absolute address of the registers is device/port specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

Table 4–5. Video Display Control Registers (Continued)

Offset Address†	Acronym	Register Name	Section
254h	VDDISPEVT	Video Display Display Event Register	4.12.22
258h	VDCLIP	Video Display Clipping Register	4.12.23
25Ch	VDDEFVAL	Video Display Default Display Value Register	4.12.24
260h	VDVINT	Video Display Vertical Interrupt Register	4.12.25
264h	VDFBIT	Video Display Field Bit Register	4.12.26
268h	VDVBIT1	Video Display Field 1 Vertical Blanking Bit Register	4.12.27
26Ch	VDVBIT2	Video Display Field 2 Vertical Blanking Bit Register	4.12.28

† The absolute address of the registers is device/port specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

4.12.1 Video Display Status Register (VDSTAT)

The video display status register (VDSTAT) indicates the current display status of the video port. The VDSTAT is shown in Figure 4–39 and described in Table 4–6.

The VDXPOS and VDYPOS bits track the coordinates of the most-recently displayed pixel. The F1D, F2D, and FRMD bits indicate the completion of fields or frames and may need to be cleared by the DSP to prevent a DCNA interrupt from being generated, depending on the selected frame operation. The F1D, F2D, and FRMD bits are set when the final pixel from the appropriate field has been sent to the output pad.

Figure 4–39. Video Display Status Register (VDSTAT)

31	30	29	28	27–16
—	FRMD	F2D	F1D	VDYPOS
R-0	R/WC-0	R/WC-0	R/WC-0	R-0

15–14	13	12	11–0
Reserved	VBLNK	VDFLD	VDXPOS
R-0	R-0	R-0	R-0

Legend: R = Read only; WC = Write 1 to clear, write of 0 has no effect; *-n* = value after reset

Table 4−6. Video Display Status Register (VDSTAT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
30	FRMD			Frame displayed bit. Write 1 to clear the bit, a write of 0 has no effect.
		NONE	0	Complete frame has not been displayed.
		DISPLAYED	1	Complete frame has been displayed.
29	F2D			Field 2 displayed bit. Write 1 to clear the bit, a write of 0 has no effect.
		NONE	0	Field 2 has not been displayed.
		DISPLAYED	1	Field 2 has been displayed.
28	F1D			Field 1 displayed bit. Write 1 to clear the bit, a write of 0 has no effect.
		NONE	0	Field 1 has not been displayed.
		DISPLAYED	1	Field 1 has been displayed.
27−16	VDYPOS	OF(*value*)	0−FFFh	Current frame line counter (FLCOUNT) value. Index of the current line in the current field being displayed by the module.
15−14	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
13	VBLNK			Vertical blanking bit.
		EMPTY	0	Video display is not in a vertical-blanking interval.
		NOTEMPTY	1	Video display is in a vertical-blanking interval.
12	VDFLD			VDFLD bit indicates which field is currently being displayed. The VDFLD bit is updated at the start of the vertical blanking interval of the next field.
		FIELD1ACT	0	Field 1 is active.
		FIELD2ACT	1	Field 2 is active.
11−0	VDXPOS	OF(*value*)	0−FFFh	Current frame pixel counter (FPCOUNT) value. Index of the most recently output pixel.

† For CSL implementation, use the notation VD_VDSTAT_*field_symval*

4.12.2 Video Display Control Register (VDCTL)

The video display is controlled by the video display control register (VDCTL). The VDCTL is shown in Figure 4−40 and described in Table 4−7.

Figure 4−40. Video Display Control Register (VDCTL)

31	30	29	28	27	24
RSTCH	BLKDIS	Reserved	PVPSYN	Reserved	
R/WS-0	R/W-1	R-0	R/W-0	R-0	

23	22	21	20	19	18	17	16
FXS	VXS	HXS	VCTL2S	VCTL1S		VCTL0S	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0		R/W-0	

15	14	13	12	11	10	9	8
VDEN	DPK	RGBX	RSYNC	DVEN	RESMPL	Reserved	SCALE
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0

7	6	5	4	3	2	0
CON	FRAME	DF2	DF1	Reserved	DMODE	
R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	

Legend: R = Read only; R/W = Read/Write; WS = Write 1 to reset, write of 0 has no effect; *-n* = value after reset

Table 4−7. Video Display Control Register (VDCTL) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31	RSTCH			Reset channel bit. Write 1 to reset the bit, a write of 0 has no effect.	
		NONE	0	No effect.	
		RESET	1	Resets the video display module and sets its registers to their initial values. Also clears the VDEN bit. The video display module automatically clears RSTCH after software reset is completed.	

† For CSL implementation, use the notation VP_VDCTL_*field_symval*
‡ For complete encoding of these bits, see Table 4−4.

Table 4−7. Video Display Control Register (VDCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
30	BLKDIS			Block display events bit. BLKDIS functions as a display FIFO reset without affecting the current programmable register values.	
				The video display module continues to function normally, the counters count, control outputs are generated, EAV/SAV codes are generated for BT.656 and Y/C modes, and default or blanking data is output during active display time. No data is moved to the display FIFOs because no events occur. The F1D, F2D, and FRMD bits in VDSTAT are still set when fields or frames are complete.	
		CLEAR	0	Clearing BLKDIS does not enable DMA events during the field in which the bit is cleared. DMA events are enabled at the start of the next frame after the one in which the bit is cleared. This allows the DMA to always be synced to the proper field.	
		BLOCK	1	Blocks DMA events and flushes the display FIFOs.	
29	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
28	PVPSYN			Previous video port synchronization enable bit.	
		DISABLE	0		
		ENABLE	1	Output timing is locked to preceding video port (VP2 is locked to VP1 or VP1 is locked to VP0, see Figure 4−7 on page 4-9).	
27−24	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
23	FXS			Field external synchronization enable bit.	
		OUTPUT	0	VCTL2 is an output.	
		FSINPUT	1	VCTL2 is an external field sync input.	
22	VXS			Vertical external synchronization enable bit.	
		OUTPUT	0	VCTL1 is an output.	
		VSINPUT	1	VCTL1 is an external vertical sync input.	

† For CSL implementation, use the notation VP_VDCTL_*field_symval*
‡ For complete encoding of these bits, see Table 4−4.

Table 4−7. Video Display Control Register (VDCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
21	HXS			Horizontal external synchronization enable bit.	
		OUTPUT	0	VCTL0 is an output.	
		HSINPUT	1	VCTL0 is an external horizontal sync input.	
20	VCTL2S			VCTL2 output select bit.	
		CBLNK	0	Output CBLNK	
		FLD	1	Output FLD	
19−18	VCTL1S			VCTL1 output select bit.	
		VYSYNC	0	Output VSYNC	
		VBLNK	1h	Output VBLNK	
		CSYNC	2h	Output CSYNC	
		FLD	3h	Output FLD	
17−16	VCTL0S			VCTL0 output select bit.	
		HYSYNC	0	Output HSYNC	
		HBLNK	1h	Output HBLNK	
		AVID	2h	Output AVID	
		FLD	3h	Output FLD	
15	VDEN			Video display enable bit. Other bits in VDCTL (except RSTCH and BLKDIS bits) may only be changed when VDEN = 0.	
		DISABLE	0	Video display is disabled.	
		ENABLE	1	Video display is enabled.	
14	DPK			10-bit packing format select bit.	
		N10UNPK	0	Normal 10-bit unpacking	
		D10UNPK	1	Dense 10-bit unpacking	

† For CSL implementation, use the notation VP_VDCTL_*field_symval*
‡ For complete encoding of these bits, see Table 4−4.

Table 4−7. Video Display Control Register (VDCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	**Raw Data Mode**
13	RGBX			RGB extract enable bit.	
		DISABLE	0	Not used.	
		ENABLE	1	Not used.	Perform ¾ FIFO unpacking.
12	RSYNC			Second, synchronized raw data channel enable bit.	
		DISABLE	0	Not used.	Second, synchronized raw data channel is disabled.
		ENABLE	1	Not used.	Second, synchronized raw data channel is enabled.
11	DVEN			Default value enable bit.	
		BLANKING	0	Blanking value is output during non-sourced active pixels.	Not used.
		DV	1	Default value is output during non-sourced active pixels.	Not used.
10	RESMPL			Chroma resampling enable bit.	
		DISABLE	0	Chroma resampling is disabled.	Not used.
		ENABLE	1	Chroma is horizontally resampled from 4:2:0 interspersed to 4:2:2 co-sited before output.	Not used.
9	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
8	SCALE			Scaling select bit.	
		NONE	0	No scaling	Not used.
		X2	1	2× scaling	Not used.
7	CON‡			Continuous display enable bit.	
		DISABLE	0	Continuous display is disabled.	
		ENABLE	1	Continuous display is enabled.	

† For CSL implementation, use the notation VP_VDCTL_*field_symval*
‡ For complete encoding of these bits, see Table 4−4.

Table 4–7. Video Display Control Register (VDCTL) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
6	FRAME‡			Display frame bit.	
		NONE	0	Do not display frame.	
		FRMDIS	1	Display frame.	
5	DF2‡			Display field 2 bit.	
		NONE	0	Do not display field 2.	
		FLDDIS	1	Display field 2.	
4	DF1‡			Display field 1 bit.	
		NONE	0	Do not display field 1.	
		FLDDIS	1	Display field 1.	
3	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
2–0	DMODE			Display mode select bit.	
		BT656B	0	Enables 8-bit BT.656 mode.	
		BT656D	1h	Enables 10-bit BT.656 mode.	
		RAWB	2h	Enables 8-bit raw data mode.	
		RAWD	3h	Enables 10-bit raw data mode.	
		YC16	4h	Enables 8-bit Y/C mode.	
		YC20	5h	Enables 10-bit Y/C mode.	
		RAW16	6h	Enables 16-bit raw data mode.	
		RAW20	7h	Enables 20-bit raw data mode.	

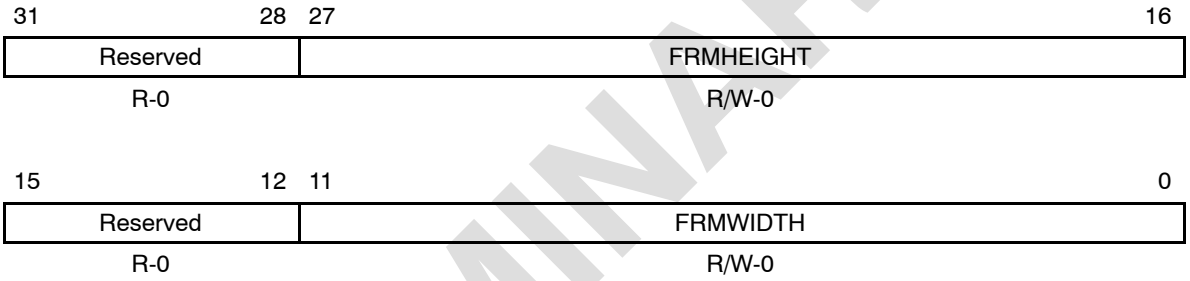
† For CSL implementation, use the notation VP_VDCTL_*field_symval*
‡ For complete encoding of these bits, see Table 4–4.

4.12.3 Video Display Frame Size Register (VDFRMSZ)

The video display frame size register (VDFRMSZ) sets the display channel frame size by setting the ending values for the frame line counter (FLCOUNT) and the frame pixel counter (FPCOUNT). The VDFRMSZ is shown in Figure 4−41 and described in Table 4−8.

The FPCOUNT starts at 0 and counts to FRMWIDTH − 1 before restarting. The FLCOUNT starts at 1 and counts to FRMHEIGHT before restarting.

Figure 4−41. Video Display Frame Size Register (VDFRMSZ)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−8. Video Display Frame Size Register (VDFRMSZ) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	FRMHEIGHT	OF(*value*)	0−FFFh	Defines the total number of lines per frame. The number is the ending value of the frame line counter (FLCOUNT). For BT.656 operation, the FRMHIGHT is set to 525 (525/60 operation) or 625 (625/50 operation).
15−12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	FRMWIDTH	OF(*value*)	0−FFFh	Defines the total number of pixels per line including blanking. The number is the frame pixel counter (FPCOUNT) ending value + 1. For BT.656 operation, the FRMWIDTH is typically 858 or 864.

† For CSL implementation, use the notation VP_VDFRMSZ_*field_symval*

4.12.4 Video Display Horizontal Blanking Register (VDHBLNK)

The video display horizontal blanking register (VDHBLNK) controls the display horizontal blanking. The VDHBLNK is shown in Figure 4−42 and described in Table 4−9.

Every time the frame pixel counter (FPCOUNT) is equal to HBLNKSTART, HBLNK is asserted. HBLNKSTART also determines where the EAV code is inserted in the BT.656 and Y/C output.

Every time FPCOUNT = HBLNKSTOP, the HBLNK signal is deasserted (this is shown in Figure 4−5, page 4-7). In BT.656 and Y/C modes, HBLNKSTOP determines the SAV code insertion point and HBLNK deassertion point. The HBLNK inactive edge may optionally be delayed by 4 pixel clocks using the HBDLA bit.

Figure 4−42. Video Display Horizontal Blanking Register (VDHBLNK)

31	28	27	16
Reserved		HBLNKSTOP	
R-0		R/W-0	

15	14	12	11	0
HBDLA	Reserved		HBLNKSTART	
R/W-0	R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−9. Video Display Horizontal Blanking Register (VDHBLNK) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	HBLNKSTOP	OF(*value*)	0−FFFh	Location of SAV code and HBLNK inactive edge within the line. HBLNK inactive edge may be optionally delayed by 4 VCLKs.	Ending pixel (FPCOUNT) of blanking video area (HBLNK inactive) within the line.
15	HBDLA			Horizontal blanking delay enable bit.	
		NONE	0	Horizontal blanking delay is disabled.	Not used.
		DELAY	1	HBLNK inactive edge is delayed by 4 VCLKs.	Not used.
14−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	HBLNKSTART	OF(*value*)	0−FFFh	Location of EAV code and HBLNK active edge within the line.	Starting pixel (FPCOUNT) of blanking video area (HBLNK active) within the line.

† For CSL implementation, use the notation VP_VDHBLNK_*field_symval*

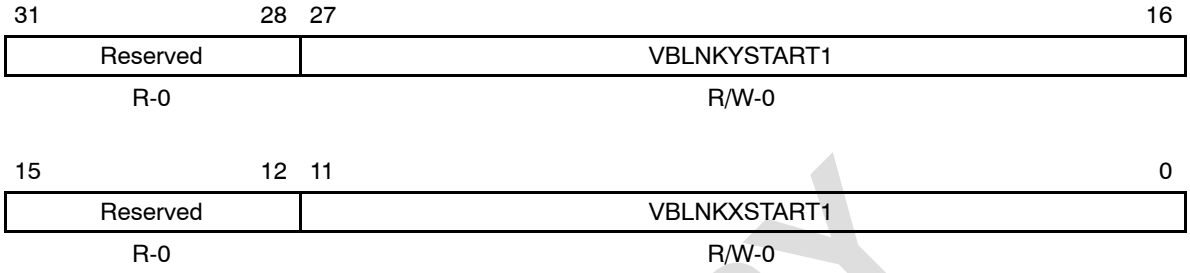
4.12.5 Video Display Field 1 Vertical Blanking Start Register (VDVBLKS1)

The video display field 1 vertical blanking start register (VDVBLKS1) controls the start of vertical blanking in field 1. The VDVBLKS1 is shown in Figure 4−43 and described in Table 4−10.

In raw data mode, VBLNK is asserted whenever the frame line counter (FLCOUNT) is equal to VBLNKYSTART1 and the frame pixel counter (FPCOUNT) is equal to VBLNKXSTART1 (this is shown in Figure 4−6, page 4-8).

In BT.656 and Y/C mode, VBLNK is asserted whenever FLCOUNT = VBLNKYSTART1 and FPCOUNT = VBLNKXSTART1. This VBLNK output control is completely independent of the timing control codes. The V bit in the EAV/SAV codes for field 1 is controlled by the VDVBIT1 register.

Figure 4−43. Video Display Field 1 Vertical Blanking Start Register (VDVBLKS1)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−10. Video Display Field 1 Vertical Blanking Start Register (VDVBLKS1) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	VBLNKYSTART1	OF(*value*)	0−FFFh	Specifies the line (in FLCOUNT) where VBLNK active edge occurs for field 1. Does not affect EAV/SAV V bit operation.	Specifies the line (in FLCOUNT) where vertical blanking begins (VBLNK active edge) for field 1.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	VBLNKXSTART1	OF(*value*)	0−FFFh	Specifies the pixel (in FPCOUNT) where VBLNK active edge occurs for field 1.	Specifies the pixel (in FPCOUNT) where vertical blanking begins (VBLNK active edge) for field 1.

† For CSL implementation, use the notation VP_VDVBLKS1_*field_symval*

4.12.6 Video Display Field 1 Vertical Blanking End Register (VDVBLKE1)

The video display field 1 vertical blanking end register (VDVBLKE1) controls the end of vertical blanking in field 1. The VDVBLKE1 is shown in Figure 4−44 and described in Table 4−11.

In raw data mode, VBLNK is deasserted whenever the frame line counter (FLCOUNT) is equal to VBLNKYSTOP1 and the frame pixel counter (FPCOUNT) is equal to VBLNKXSTOP1 (this is shown in Figure 4−6, page 4-8).

In BT.656 and Y/C mode, VBLNK is deasserted whenever FLCOUNT = VBLNKYSTOP1 and FPCOUNT = VBLNKXSTOP1. This VBLNK output control is completely independent of the timing control codes. The V bit in the EAV/SAV codes for field 1 is controlled by the VDVBIT1 register.

Figure 4−44. Video Display Field 1 Vertical Blanking End Register (VDVBLKE1)

31	28	27	16
Reserved		VBLNKYSTOP1	
R-0		R/W-0	

15	12	11	0
Reserved		VBLNKXSTOP1	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−11. Video Display Field 1 Vertical Blanking End Register (VDVBLKE1) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	VBLNKYSTOP1	OF(*value*)	0−FFFh	Specifies the line (in FLCOUNT) where VBLNK inactive edge occurs for field 1. Does not affect EAV/SAV V bit operation.	Specifies the line (in FLCOUNT) where vertical blanking ends (VBLNK inactive edge) for field 1.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	VBLNKXSTOP1	OF(*value*)	0−FFFh	Specifies the pixel (in FPCOUNT) where VBLNK inactive edge occurs for field 1.	Specifies the pixel (in FPCOUNT) where vertical blanking ends (VBLNK inactive edge) for field 1.

† For CSL implementation, use the notation VP_VDVBLKE1_*field_symval*

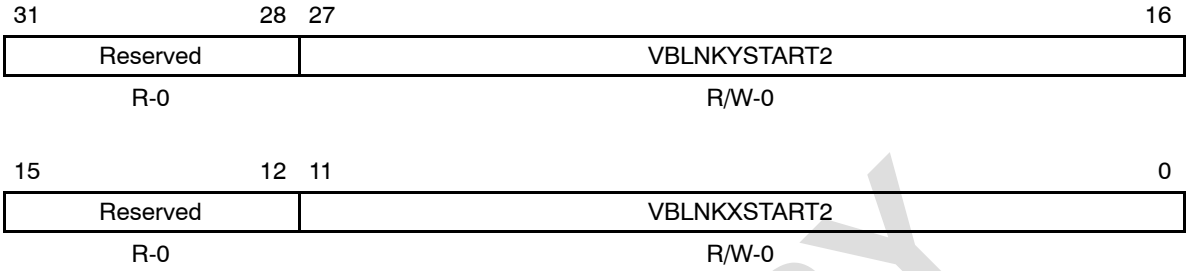
4.12.7 Video Display Field 2 Vertical Blanking Start Register (VDVBLKS2)

The video display field 2 vertical blanking start register (VDVBLKS2) controls the start of vertical blanking in field 2. The VDVBLKS2 is shown in Figure 4−45 and described in Table 4−12.

In raw data mode, VBLNK is asserted whenever the frame line counter (FLCOUNT) is equal to VBLNKYSTART2 and the frame pixel counter (FPCOUNT) is equal to VBLNKXSTART2 (this is shown in Figure 4−6, page 4-8).

In BT.656 and Y/C mode, VBLNK is asserted whenever FLCOUNT = VBLNKYSTART2 and FPCOUNT = VBLNKXSTART2. This VBLNK output control is completely independent of the timing control codes. The V bit in the EAV/SAV codes for field 2 is controlled by the VDVBIT2 register.

Figure 4−45. Video Display Field 2 Vertical Blanking Start Register (VDVBLKS2)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−12. Video Display Field 2 Vertical Blanking Start Register (VDVBLKS2) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	VBLNKYSTART2	OF(*value*)	0−FFFh	Specifies the line (in FLCOUNT) where VBLNK active edge occurs for field 2. Does not affect EAV/SAV V bit operation.	Specifies the line (in FLCOUNT) where vertical blanking begins (VBLNK active edge) for field 2.
15−12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	VBLNKXSTART2	OF(*value*)	0−FFFh	Specifies the pixel (in FPCOUNT) where VBLNK active edge occurs for field 2.	Specifies the pixel (in FPCOUNT) where vertical blanking begins (VBLNK active edge) for field 2.

† For CSL implementation, use the notation VP_VDVBLKS2_*field_symval*

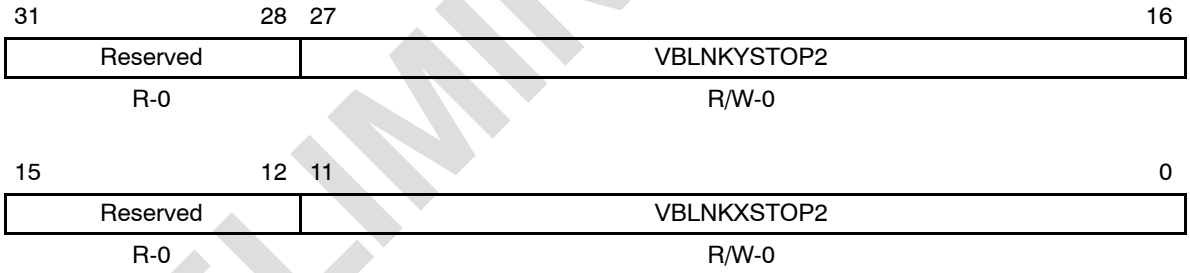
4.12.8 Video Display Field 2 Vertical Blanking End Register (VDVBLKE2)

The video display field 2 vertical blanking end register (VDVBLKE2) controls the end of vertical blanking in field 2. The VDVBLKE2 is shown in Figure 4−46 and described in Table 4−13.

In raw data mode, VBLNK is deasserted whenever the frame line counter (FLCOUNT) is equal to VBLNKYSTOP2 and the frame pixel counter (FPCOUNT) is equal to VBLNKXSTOP2 (this is shown in Figure 4−6, page 4-8).

In BT.656 and Y/C mode, VBLNK is deasserted whenever FLCOUNT = VBLNKYSTOP2 and FPCOUNT = VBLNKXSTOP2. This VBLNK output control is completely independent of the timing control codes. The V bit in the EAV/SAV codes for field 2 is controlled by the VDVBIT2 register.

Figure 4−46. Video Display Field 2 Vertical Blanking End Register (VDVBLKE2)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−13. Video Display Field 2 Vertical Blanking End Register (VDVBLKE2) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	VBLNKYSTOP2	OF(*value*)	0−FFFh	Specifies the line (in FLCOUNT) where VBLNK inactive edge occurs for field 2. Does not affect EAV/SAV V bit operation.	Specifies the line (in FLCOUNT) where vertical blanking ends (VBLNK inactive edge) for field 2.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	VBLNKXSTOP2	OF(*value*)	0−FFFh	Specifies the pixel (in FPCOUNT) where VBLNK inactive edge occurs for field 2.	Specifies the pixel (in FPCOUNT) where vertical blanking ends (VBLNK inactive edge) for field 2.

† For CSL implementation, use the notation VP_VDVBLKE2_*field_symval*

4.12.9 Video Display Field 1 Image Offset Register (VDIMGOFF1)

The video display field 1 image offset register (VDIMGOFF1) defines the field 1 image offset and specifies the starting location of the displayed image relative to the start of the active display. The VDIMGOFF1 is shown in Figure 4−47 and described in Table 4−14.

The image line counter (ILCOUNT) is reset to 1 on the first image line (when FLCOUNT = VBLNKYSTOP1 + IMGVOFF1). If the NV bit is set, ILCOUNT is reset to 1 when FLCOUNT = VBLNKYSTOP1 − IMGVOFF1. Display image pixels are output in field 1 beginning on the line where ILCOUNT = 1. The default output values or blanking values are output during active lines prior to ILCOUNT = 1. For a negative offset, IMGVOFF1 must not be greater than VBLNKYSTOP1. The field 1 active image must not overlap the field 2 active image.

The image pixel counter (IPCOUNT) is reset to 0 at the start of an active line image. Once ILCOUNT = 1, image pixels from the FIFO are output on each line in field 1 beginning when FPCOUNT = IMGHOFF1. If the NH bit is set, IPCOUNT is reset when FPCOUNT = FRMWIDTH − IMGHOFF1. The default output values or blanking values are output during active pixels prior to IMGHOFF1.

Figure 4−47. Video Display Field 1 Image Offset Register (VDIMGOFF1)

31	30 28	27 16
NV	Reserved	IMGVOFF1
R/W-0	R-0	R/W-0

15	14 12	11 0
NH	Reserved	IMGHOFF1
R/W-0	R-0	R/W-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−14. Video Display Field 1 Image Offset Register (VDIMGOFF1) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31	NV			Negative vertical image offset enable bit.	
		NONE	0		Not used.
		NEGOFF	1	Display image window begins before the first active line of field 1. (Used for VBI data output.)	Not used.
30−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	IMGVOFF1	OF(*value*)	0−FFFh	Specifies the display image vertical offset in lines from the first active line of field 1.	
15	NH			Negative horizontal image offset.	
		NONE	0		Not used.
		NEGOFF	1	Display image window begins before the start of active video. (Used for HANC data output.)	Not used.
14−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	IMGHOFF1	OF(*value*)	0−FFFh	Specifies the display image horizontal offset in pixels from the start of each line of active video in field 1. This must be an even number (the LSB is treated as 0).	Specifies the display image horizontal offset in pixels from the start of each line of active video in field 1.

† For CSL implementation, use the notation VP_VDIMGOFF1_*field_symval*

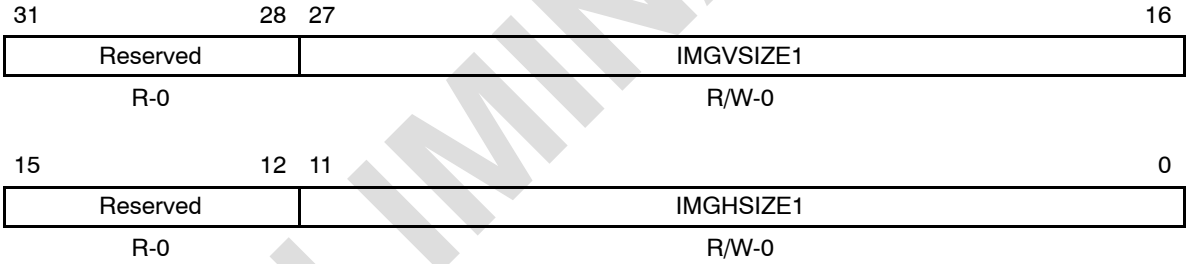
4.12.10 Video Display Field 1 Image Size Register (VDIMGSZ1)

The video display field 1 image size register (VDIMGSZ1) defines the field 1 image area and specifies the size of the displayed image within the active display. The VDIMGSZ1 is shown in Figure 4−48 and described in Table 4−15.

The image pixel counter (IPCOUNT) counts displayed image pixel output on each of the displayed image. Displayed image pixel output stops when IPCOUNT = IMGHSIZE1. The default output values or blanking values are output for the remainder of the active line.

The image line counter (ILCOUNT) counts displayed image lines. Displayed image output stops when ILCOUNT = IMGVSIZE1. The default output values or blanking values are output for the remainder of the active field.

Figure 4−48. Video Display Field 1 Image Size Register (VDIMGSZ1)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−15. Video Display Field 1 Image Size Register (VDIMGSZ1) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	IMGVSIZE1	OF(*value*)	0−FFFh	Specifies the display image height in lines.	
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	IMGHSIZE1	OF(*value*)	0−FFFh	Specifies the display image width in pixels. This number must be even (the LSB is treated as 0)	Specifies the display image width in pixels.

† For CSL implementation, use the notation VP_VDIMGSZ1_*field_symval*

4.12.11 Video Display Field 2 Image Offset Register (VDIMGOFF2)

The video display field 2 image offset register (VDIMGOFF2) defines the field 2 image offset and specifies the starting location of the displayed image relative to the start of the active display. The VDIMGOFF2 is shown in Figure 4−49 and described in Table 4−16.

The image line counter (ILCOUNT) is reset to 1 on the first image line (when FLCOUNT = VBLNKYSTOP2 + IMGVOFF2). If the NV bit is set, ILCOUNT is reset to 1 when FLCOUNT = VBLNKYSTOP2 − IMGVOFF2. Display image pixels are output in field 2 beginning on the line where ILCOUNT = 1. The default output values or blanking values are output during active lines prior to ILCOUNT = 1. For a negative offset, IMGVOFF2 must not be greater than VBLNKYSTOP2. The field 2 active image must not overlap the field 2 active image.

The image pixel counter (IPCOUNT) is reset to 0 at the start of an active line image. Once ILCOUNT = 1, image pixels from the FIFO are output on each line in field 2 beginning when FPCOUNT = IMGHOFF2. If the NH bit is set, IPCOUNT is reset when FPCOUNT = FRMWIDTH − IMGHOFF2. The default output values or blanking values are output during active pixels prior to IMGHOFF2.

Figure 4−49. Video Display Field 2 Image Offset Register (VDIMGOFF2)

31	30	28	27	16
NV	Reserved		IMGVOFF2	
R/W-0	R-0		R/W-0	

15	14	12	11	0
NH	Reserved		IMGHOFF2	
R/W-0	R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−16. Video Display Field 2 Image Offset Register (VDIMGOFF2) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	**Raw Data Mode**
31	NV			Negative vertical image offset enable bit.	
		NONE	0		Not used.
		NEGOFF	1	Display image window begins before the first active line of field 2. (Used for VBI data output.)	Not used.
30−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	IMGVOFF2	OF(*value*)	0−FFFh	Specifies the display image vertical offset in lines from the first active line of field 2.	
15	NH			Negative horizontal image offset.	
		NONE	0		Not used.
		NEGOFF	1	Display image window begins before the start of active video. (Used for HANC data output.)	Not used.
14−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	IMGHOFF2	OF(*value*)	0−FFFh	Specifies the display image horizontal offset in pixels from the start of each line of active video in field 2. This must be an even number (the LSB is treated as 0).	Specifies the display image horizontal offset in pixels from the start of each line of active video in field 2.

† For CSL implementation, use the notation VP_VDIMGOFF2_*field_symval*

4.12.12 Video Display Field 2 Image Size Register (VDIMGSZ2)

The video display field 2 image size register (VDIMGSZ2) defines the field 2 image area and specifies the size of the displayed image within the active display. The VDIMGSZ2 is shown in Figure 4−50 and described in Table 4−17.

The image pixel counter (IPCOUNT) counts displayed image pixel output on each of the displayed image. Displayed image pixel output stops when IPCOUNT = IMGHSIZE2. The default output values or blanking values are output for the remainder of the active line.

The image line counter (ILCOUNT) counts displayed image lines. Displayed image output stops when ILCOUNT = IMGVSIZE2. The default output values or blanking values are output for the remainder of the active field.

Figure 4−50. Video Display Field 2 Image Size Register (VDIMGSZ2)

31	28	27	16
Reserved		IMGVSIZE2	
R-0		R/W-0	

15	12	11	0
Reserved		IMGHSIZE2	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−17. Video Display Field 2 Image Size Register (VDIMGSZ2) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	IMGVSIZE2	OF(*value*)	0−FFFh	Specifies the display image height in lines.	
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	IMGHSIZE2	OF(*value*)	0−FFFh	Specifies the display image width in pixels. This number must be even (the LSB is treated as 0)	Specifies the display image width in pixels.

† For CSL implementation, use the notation VP_VDIMGSZ2_*field_symval*

4.12.13 Video Display Field 1 Timing Register (VDFLDT1)

The video display field 1 timing register (VDFLDT1) sets the timing of the field identification signal. The VDFLDT1 is shown in Figure 4−51 and described in Table 4−18.

In raw data mode, the FLD signal is deasserted to indicate field 1 display whenever the frame line counter (FLCOUNT) is equal to FLD1YSTART and the frame pixel counter (FPCOUNT) is equal to FLD1XSTART (this is shown in Figure 4−6, page 4-8).

In BT.656 and Y/C mode, the FLD signal is deasserted to indicate field 1 display whenever FLCOUNT = FLD1YSTART and FPCOUNT = FLD1XSTART. The FLD output is completely independent of the timing control codes. The F bit in the EAV/SAV codes is controlled by the VDFBIT register.

Figure 4−51. Video Display Field 1 Timing Register (VDFLDT1)

31	28	27	16
Reserved		FLD1YSTART	
R-0		R/W-0	

15	12	11	0
Reserved		FLD1XSTART	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−18. Video Display Field 1 Timing Register (VDFLDT1) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	FLD1YSTART	OF(*value*)	0−FFFh	Specifies the first line of field 1. (The line where FLD is deasserted.)
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	FLD1XSTART	OF(*value*)	0−FFFh	Specifies the pixel on the first line of field 1 where the FLD output is deasserted.

† For CSL implementation, use the notation VP_VDFLDT1_*field_symval*

4.12.14 Video Display Field 2 Timing Register (VDFLDT2)

The video display field 2 timing register (VDFLDT2) sets the timing of the field identification signal. The VDFLDT2 is shown in Figure 4−52 and described in Table 4−19.

In raw data mode, the FLD signal is asserted whenever the frame line counter (FLCOUNT) is equal to FLD2YSTART and the frame pixel counter (FPCOUNT) is equal to FLD2XSTART (this is shown in Figure 4−6, page 4-8).

In BT.656 and Y/C mode, the FLD signal is asserted to indicate field 2 display whenever FLCOUNT = FLD2YSTART and FPCOUNT = FLD2XSTART. The FLD output is completely independent of the timing control codes. The F bit in the EAV/SAV codes is controlled by the VDFBIT register.

Figure 4−52. Video Display Field 2 Timing Register (VDFLDT2)

31	28	27	16
Reserved		FLD2YSTART	
R-0		R/W-0	

15	12	11	0
Reserved		FLD2XSTART	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−19. Video Display Field 2 Timing Register (VDFLDT2) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	FLD2YSTART	OF(*value*)	0−FFFh	Specifies the first line of field 2. (The line where FLD is asserted.)
15−12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	FLD2XSTART	OF(*value*)	0−FFFh	Specifies the pixel on the first line of field 2 where the FLD output is asserted.

† For CSL implementation, use the notation VP_VDFLDT2_*field_symval*

4.12.15 Video Display Threshold Register (VDTHRLD)

The video display threshold register (VDTHRLD) sets the display FIFO threshold to determine when to load more display data. The VDTHRLD is shown in Figure 4−53 and described in Table 4−20.

The VDTHRLD*n* bits determines how much space must be available in the display FIFOs before the appropriate DMA event may be generated. The Y FIFO uses the VDTHRLD*n* value directly while the Cb and Cr values use ½ the VDTHRLD*n* value rounded up to the next doubleword (1/2 (VDTHRLD*n* + VTHRLD*n* mod 2). The DMA transfer size must be less than the value used for each FIFO. Typically, VDTHRLD*n* is set to the horizontal line length rounded up to the next doubleword boundary. For nonline length thresholds, the display data unpacking mechanism places certain restrictions of what VDTHRLD*n* values are valid (see section 2.3.3).

The VDTHRLD2 bits behaves identically to VDTHRLD1, but are used during field 2 capture. It is only used if the field 2 DMA size needs to be different from the field 1 DMA size for some reason (for example, different display line lengths in field 1 and field 2).

In raw display mode, the INCPIX bits determine when the frame pixel counter (FPCOUNT) is incremented . If, for example, each output value represents the R, G, or B portion of a display pixel, then the INCPIX bits are set to 3h so that the pixel counter is incremented only on every third output clock. An INCPIX value of 0h represents a count of 16 rather than 0.

Figure 4−53. Video Display Threshold Register (VDTHRLD)

31	26	25	16
Reserved		VDTHRLD2	
R-0		R/W-0	

15	12	11	10	9	0
INCPIX		Reserved		VDTHRLD1	
R/W-0001		R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−20. Video Display Threshold Register (VDTHRLD) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
31−26	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
25−16	VDTHRLD2	OF(*value*)	0−3FFh	Field 2 threshold. Whenever there are at least VDTHRLD doublewords of space in the Y display FIFO, a new Y DMA event may be generated. Whenever there are at least ½ VDTHRLD doublewords of space in the Cb or Cr display FIFO, a new Cb or Cr DMA event may be generated.	Field 2 threshold. Whenever there are at least VDTHRLD doublewords of space in the display FIFO, a new Y DMA event may be generated.
15−12	INCPIX	OF(*value*)	0−Fh	Not used.	FPCOUNT is incremented every INCPIX output clocks.
11−10	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
9−0	VDTHRLD1	OF(*value*)	0−3FFh	Field 1 threshold. Whenever there are at least VDTHRLD doublewords of space in the Y display FIFO, a new Y DMA event may be generated. Whenever there are at least ½ VDTHRLD doublewords of space in the Cb or Cr display FIFO, a new Cb or Cr DMA event may be generated.	Field 1 threshold. Whenever there are at least VDTHRLD doublewords of space in the display FIFO, a new Y DMA event may be generated.

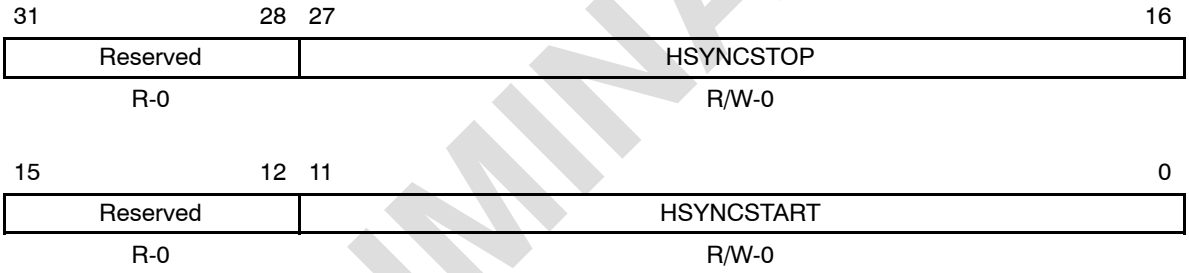
† For CSL implementation, use the notation VP_VDTHRLD_*field_symval*

4.12.16 Video Display Horizontal Synchronization Register (VDHSYNC)

The video display horizontal synchronization register (VDHSYNC) controls the timing of the horizontal synchronization signal. The VDHSYNC is shown in Figure 4−54 and described in Table 4−21.

Generation of the horizontal synchronization is shown in Figure 4−5, page 4-7. The HSYNC signal is asserted to indicate the start of the horizontal sync pulse whenever the frame pixel counter (FPCOUNT) is equal to HSYNC-START. The HSYNC signal is deasserted to indicate the end of the horizontal sync pulse whenever FPCOUNT = HSYNCSTOP.

Figure 4−54. Video Display Horizontal Synchronization Register (VDHSYNC)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−21. Video Display Horizontal Synchronization Register (VDHSYNC) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	HSYNCSTOP	OF(*value*)	0−FFFh	Specifies the pixel where HSYNC is deasserted.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	HSYNCSTART	OF(*value*)	0−FFFh	Specifies the pixel where HSYNC is asserted.

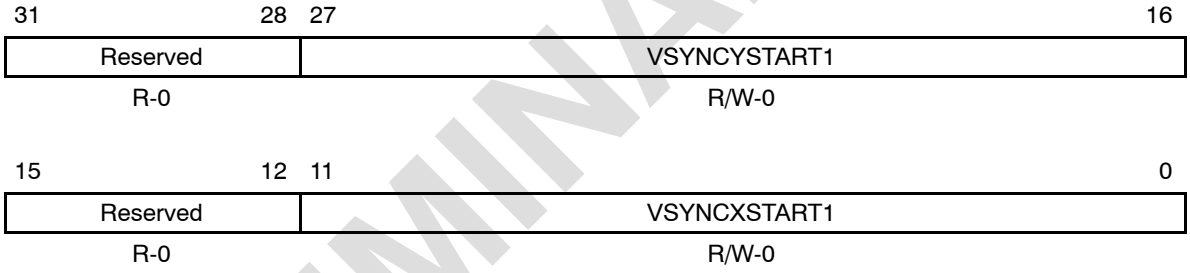
† For CSL implementation, use the notation VP_VDHSYNC_*field_symval*

4.12.17 Video Display Field 1 Vertical Synchronization Start Register (VDVSYNS1)

The video display field 1 vertical synchronization start register (VDVSYNS1) controls the start of vertical synchronization in field 1. The VDVSYNS1 is shown in Figure 4−55 and described in Table 4−22.

Generation of the vertical synchronization is shown in Figure 4−6, page 4-8. The VSYNC signal is asserted whenever the frame line counter (FLCOUNT) is equal to VSYNCYSTART1 and the frame pixel counter (FPCOUNT) is equal to VSYNCXSTART1.

Figure 4−55. Video Display Field 1 Vertical Synchronization Start Register (VDVSYNS1)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−22. Video Display Field 1 Vertical Synchronization Start Register (VDVSYNS1) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	VSYNCYSTART1	OF(*value*)	0−FFFh	Specifies the line where VSYNC is asserted for field 1.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	VSYNCXSTART1	OF(*value*)	0−FFFh	Specifies the pixel where VSYNC is asserted in field 1.

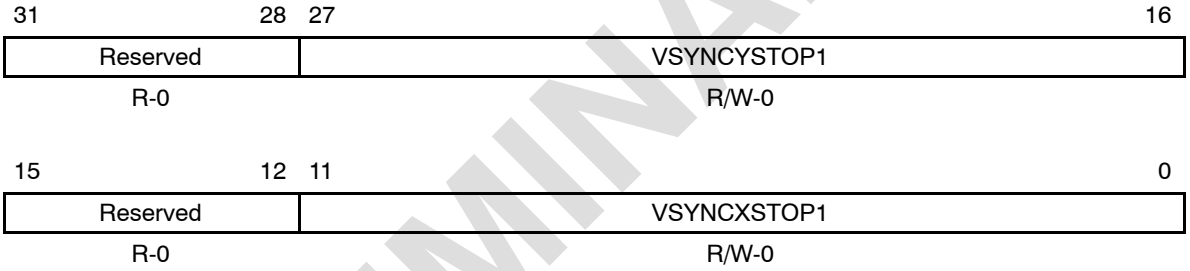
† For CSL implementation, use the notation VP_VDVSYNS1_*field_symval*

4.12.18 Video Display Field 1 Vertical Synchronization End Register (VDVSYNE1)

The video display field 1 vertical synchronization end register (VDVSYNE1) controls the end of vertical synchronization in field 1. The VDVSYNE1 is shown in Figure 4−56 and described in Table 4−23.

Generation of the vertical synchronization is shown in Figure 4−6, page 4-8. The VSYNC signal is deasserted whenever the frame line counter (FLCOUNT) is equal to VSYNCYSTOP1 and the frame pixel counter (FPCOUNT) is equal to VSYNCXSTOP1.

Figure 4−56. Video Display Field 1 Vertical Synchronization End Register (VDVSYNE1)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−23. Video Display Field 1 Vertical Synchronization End Register (VDVSYNE1) Field Descriptions

Bit	***field***†	***symval***†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	VSYNCYSTOP1	OF(*value*)	0−FFFh	Specifies the line where VSYNC is deasserted for field 1.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	VSYNCXSTOP1	OF(*value*)	0−FFFh	Specifies the pixel where VSYNC is deasserted in field 1.

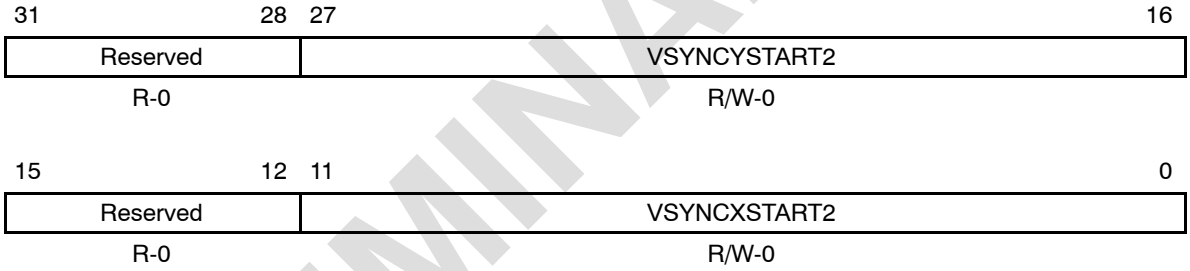
† For CSL implementation, use the notation VP_VDVSYNE1_*field_symval*

4.12.19 Video Display Field 2 Vertical Synchronization Start Register (VDVSYNS2)

The video display field 2 vertical synchronization start register (VDVSYNS2) controls the start of vertical synchronization in field 2. The VDVSYNS2 is shown in Figure 4−57 and described in Table 4−24.

Generation of the vertical synchronization is shown in Figure 4−6, page 4-8. The VSYNC signal is asserted whenever the frame line counter (FLCOUNT) is equal to VSYNCYSTART2 and the frame pixel counter (FPCOUNT) is equal to VSYNCXSTART2.

Figure 4−57. Video Display Field 2 Vertical Synchronization Start Register (VDVSYNS2)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−24. Video Display Field 2 Vertical Synchronization Start Register (VDVSYNS2) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	VSYNCYSTART2	OF(*value*)	0−FFFh	Specifies the line where VSYNC is asserted for field 2.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	VSYNCXSTART2	OF(*value*)	0−FFFh	Specifies the pixel where VSYNC is asserted in field 2.

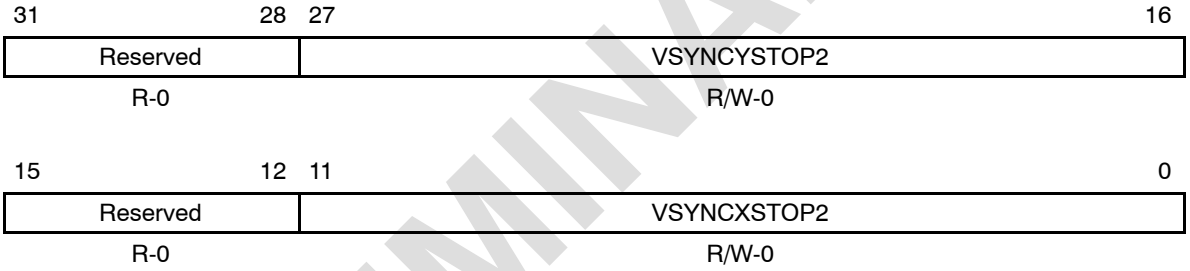
† For CSL implementation, use the notation VP_VDVSYNS2_*field_symval*

4.12.20 Video Display Field 2 Vertical Synchronization End Register (VDVSYNE2)

The video display field 2 vertical synchronization end register (VDVSYNE2) controls the end of vertical synchronization in field 2. The VDVSYNE2 is shown in Figure 4−58 and described in Table 4−25.

Generation of the vertical synchronization is shown in Figure 4−6, page 4-8. The VSYNC signal is deasserted whenever the frame line counter (FLCOUNT) is equal to VSYNCYSTOP2 and the frame pixel counter (FPCOUNT) is equal to VSYNCXSTOP2.

Figure 4−58. Video Display Field 2 Vertical Synchronization End Register (VDVSYNE2)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−25. Video Display Field 2 Vertical Synchronization End Register (VDVSYNE2) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	VSYNCYSTOP2	OF(*value*)	0−FFFh	Specifies the line where VSYNC is deasserted for field 2.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	VSYNCXSTOP2	OF(*value*)	0−FFFh	Specifies the pixel where VSYNC is deasserted in field 2.

† For CSL implementation, use the notation VP_VDVSYNE2_*field_symval*

4.12.21 Video Display Counter Reload Register (VDRELOAD)

When external horizontal or vertical synchronization are used, the video display counter reload register (VDRELOAD) determines what values are loaded into the counters when an external sync is activated. The VDRELOAD is shown in Figure 4−59 and described in Table 4−26.

Figure 4−59. Video Display Counter Reload Register (VDRELOAD)

31	28	27	16
Reserved		VRLD	
R-0		R/W-0	

15	12	11	0
CRLD		HRLD	
R/W-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−26. Video Display Counter Reload Register (VDRELOAD) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	VRLD	OF(*value*)	0−FFFh	Value loaded into frame line counter (FLCOUNT) when external VSYNC occurs.
15−12	CRLD	OF(*value*)	0−Fh	Value loaded into video clock counter (VCCOUNT) when external HSYNC occurs.
11−0	HRLD	OF(*value*)	0−FFFh	Value loaded into frame pixel counter (FPCOUNT) when external HSYNC occurs.

† For CSL implementation, use the notation VP_VDRELOAD_*field_symval*

4.12.22 Video Display Display Event Register (VDDISPEVT)

The video display display event register (VDDISPEVT) is programmed with the number of DMA events to be generated for display field 1 and field 2. The VDDISPEVET is shown in Figure 4−60 and described in Table 4−27.

Figure 4−60. Video Display Display Event Register (VDDISPEVT)

31	28	27	16
Reserved		DISPEVT2	
R-0		R/W-0	

15	12	11	0
Reserved		DISPEVT1	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−27. Video Display Display Event Register (VDDISPEVT) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	DISPEVT2	OF(*value*)	0−FFFh	Specifies the number of DMA event sets (YEVT, CbEVT, CrEVT) to be generated for field 2 output.	Specifies the number of DMA events (YEVT) to be generated for field 2 output.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	DISPEVT1	OF(*value*)	0−FFFh	Specifies the number of DMA event sets (YEVT, CbEVT, CrEVT) to be generated for field 1 output.	Specifies the number of DMA events (YEVT) to be generated for field 1 output.

† For CSL implementation, use the notation VP_VDDISPEVT_DISPEVT*n_symval*

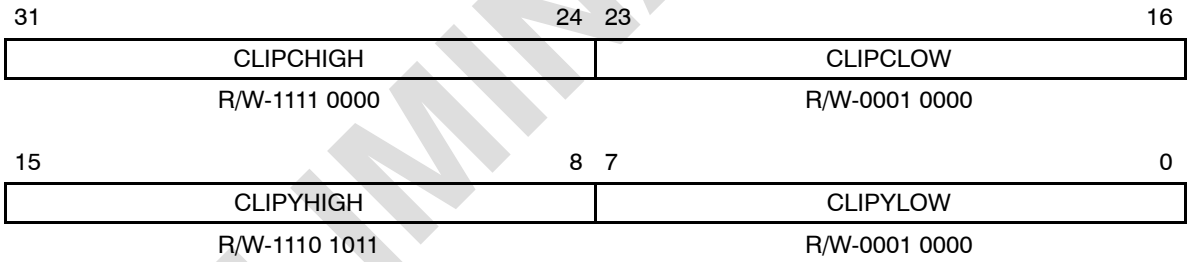
4.12.23 Video Display Clipping Register (VDCLIP)

The video display clipping register (VDCLIP) is shown in Figure 4−61 and described in Table 4−28.

The video display module in the BT.656 and Y/C modes performs programmable clipping. The clipping is performed as the last step of the video pipeline. It is applied only on the image areas defined by VDIMGSZ*n* and VDIMGOFF*n* inside the active video area (blanking values are not clipped).

VDCLIP allows output values to be clamped within the specified values. The default values are the BT.601-specified peak black level of 16 and peak white level of 235 for luma and the maximum quantization levels of 16 and 240 for chroma. For 10-bit operation, the clipping is applied to the 8 MSBs of the value with the 2 LSBs cleared. (For example, a Y value of FF.8h is clipped to EB.0h and a Y value of 0F.4h is clipped to 10.0h.)

Figure 4−61. Video Display Clipping Register (VDCLIP)



Legend: R/W = Read/Write; -*n* = value after reset

Table 4−28. Video Display Clipping Register (VDCLIP) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−24	CLIPCHIGH	OF(*value*)	0−FFh	A Cb or Cr value greater than CLIPCHIGH is forced to the CLIPCHIGH value.	Not used.
23−16	CLIPCLOW	OF(*value*)	0−FFh	A Cb or Cr value less than CLIPCLOW is forced to the CLIPCLOW value.	Not used.
15−8	CLIPYHIGH	OF(*value*)	0−FFh	A Y value greater than CLIPYHIGH is forced to the CLIPYHIGH value.	Not used.
7−0	CLIPYLOW	OF(*value*)	0−FFh	A Y value less than CLIPYLOW is forced to the CLIPYLOW value.	Not used.

† For CSL implementation, use the notation VP_VDCLIP_*field_symval*

4.12.24 Video Display Default Display Value Register (VDDEFVAL)

The video display default display value register (VDDEFVAL) defines the default value to be output during the portion of the active video window that is not part of the displayed image. The VDDEFVAL is shown in Figure 4−62 for the BT.656 and Y/C modes and in Figure 4−63 for the raw data mode, and described in Table 4−29.

The default value is output during the nonimage display window portions of the active video. This is the region between ILCOUNT = 0 and ILCOUNT = IMGVOFF*n* vertically, and between IPCOUNT = 0 and IPCOUNT = IMGHOFF*n* horizontally. In BT.656 mode, CBDEFVAL, YDEFVAL, and CRDEFVAL are multiplexed on the output in the standard CbYCrY manner. In Y/C mode, YDEFVAL is output on the VDOUT[9−0] bus and CBDEFVAL and CRDEFVAL are multiplexed on the VDOUT[19−10] bus. In all cases, the default values are output on the 8 MSBs of the bus ([9−2] or [19−12]) and the 2 LSBs ([1−0] or [11−10]) are driven as 0s.

In raw data mode, the least significant 8, 10, 16, or 20 bits of DEFVAL are output depending on the bus width. The default value is also output during the horizontal and vertical blanking periods in raw data mode.

The default value is also output during the entire active video region when the BLKDIS bit in VDCTL is set and the FIFO is empty.

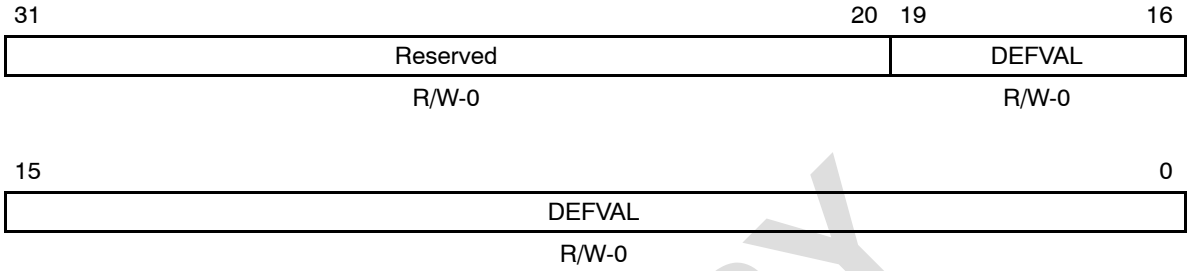
Figure 4−62. Video Display Default Display Value Register (VDDEFVAL)

31	24	23	16
CRDEFVAL		CBDEFVAL	
R/W-0		R/W-0	

15	8	7	0
Reserved		YDEFVAL	
R/W-0		R/W-0	

Legend: R/W = Read/Write; -*n* = value after reset

Figure 4−63. Video Display Default Display Value Register (VDDEFVAL)—Raw Data Mode



Legend: R/W = Read/Write; *-n* = value after reset

Table 4−29. Video Display Default Display Value Register (VDDEFVAL) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−24	CRDEFVAL	OF(*value*)	0−FFh	Specifies the 8 MSBs of the default Cr display value.	Not used.
31−20‡	Reserved	–	0	Not used.	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
19−0‡	DEFVAL	OF(*value*)	0−FFFFFh	Not used.	Specifies the default raw data display value.
23−16	CBDEFVAL	OF(*value*)	0−FFh	Specifies the 8 MSBs of the default Cb display value.	Not used.
15−8	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	Not used.
7−0	YDEFVAL	OF(*value*)	0−FFh	Specifies the 8 MSBs of the default Y display value.	Not used.

† For CSL implementation, use the notation VP_VDDEFVAL_*field_symval*
‡ Raw data mode only.

4.12.25 Video Display Vertical Interrupt Register (VDVINT)

The video display vertical interrupt register (VDVINT) controls the generation of vertical interrupts in field 1 and field 2. The VDVINT is shown in Figure 4−64 and described in Table 4−30.

An interrupt can be generated upon completion of the specified line in a field (when FLCOUNT = VINT*n*). This allows the software to synchronize itself to the frame or field. The interrupt can be programmed to occur in one, both, or no fields using the VIF1 and VIF2 bits.

Figure 4−64. Video Display Vertical Interrupt Register (VDVINT)

31	30 28	27 16
VIF2	Reserved	VINT2
R/W-0	R-0	R/W-0

15	14 12	11 0
VIF1	Reserved	VINT1
R/W-0	R-0	R/W-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−30. Video Display Vertical Interrupt Register (VDVINT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31	VIF2			Vertical interrupt (VINT) in field 2 enable bit.
		DISABLE	0	Vertical interrupt (VINT) in field 2 is disabled.
		ENABLE	1	Vertical interrupt (VINT) in field 2 is enabled.
30−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
27−16	VINT2	OF(*value*)	0−FFFh	Line where vertical interrupt (VINT) occurs, if VIF2 bit is set.
15	VIF1			Vertical interrupt (VINT) in field 1 enable bit.
		DISABLE	0	Vertical interrupt (VINT) in field 1 is disabled.
		ENABLE	1	Vertical interrupt (VINT) in field 1 is enabled.
14−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11−0	VINT1	OF(*value*)	0−FFFh	Line where vertical interrupt (VINT) occurs, if VIF1 bit is set.

† For CSL implementation, use the notation VP_VDVINT_*field_symval*

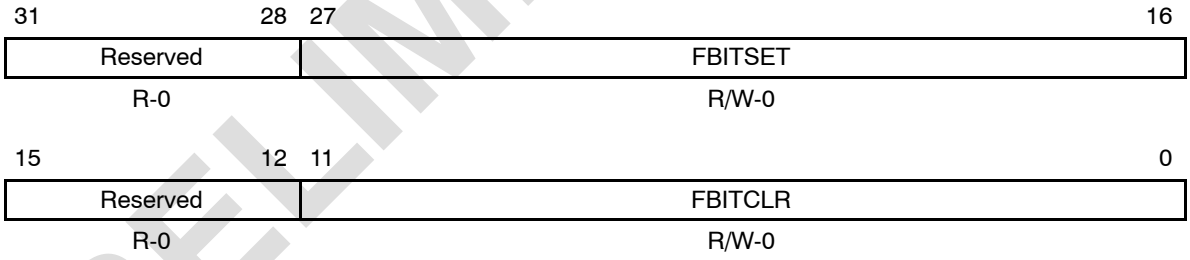
4.12.26 Video Display Field Bit Register (VDFBIT)

The video display field bit register (VDFBIT) controls the F bit value in the EAV and SAV timing control codes. The VDFBIT is shown in Figure 4−65 and described in Table 4−31.

The FBITCLR and FBITSET bits control the F bit value in the EAV and SAV timing control codes. The F bit is cleared to 0 (indicating field 1 display) in the EAV code at the beginning of the line whenever the frame line counter (FLCOUNT) is equal to FBITCLR. It remains a 0 for all subsequent EAV/SAV codes until the EAV at the beginning of the line when FLCOUNT = FBITSET where it changes to 1 (indicating field 2 display). The F bit operation is completely independent of the FLD control signal.

For interlaced operation, FBITCLR and FBITSET are typically programmed such that the F bit changes coincidently with or some time after the V bit transitions from 1 to 0 (as determined by VBITCLR1 and VBITCLR2 in VDVBIT*n*). For progressive scan operation no field 2 output occurs, so FBITSET should be programmed to a value greater than FRMHEIGHT so that the condition FLCOUNT = FBITSET never occurs and the F bit is always 0.

Figure 4−65. Video Display Field Bit Register (VDFBIT)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4−31. Video Display Field Bit Register (VDFBIT) Field Descriptions

				Description	
Bit	***field*†**	***symval*†**	**Value**	**BT.656 and Y/C Mode**	**Raw Data Mode**
31−28	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27−16	FBITSET	OF(*value*)	0−FFFh	Specifies the first line with an EAV of F = 1 indicating field 2 display.	Not used.
15−12	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11−0	FBITCLR	OF(*value*)	0−FFFh	Specifies the first line with an EAV of F = 0 indicating field 1 display.	Not used.

† For CSL implementation, use the notation VP_VDFBIT_*field_symval*

4.12.27 Video Display Field 1 Vertical Blanking Bit Register (VDVBIT1)

The video display field 1 vertical blanking bit register (VDVBIT1) controls the V bit value in the EAV and SAV timing control codes for field 1. The VDVBIT1 is shown in Figure 4−66 and described in Table 4−32.

The VBITSET1 and VBITCLR1 bits control the V bit value in the EAV and SAV timing control codes. The V bit is set to 1 (indicating the start of field 1 digital vertical blanking) in the EAV code at the beginning of the line whenever the frame line counter (FLCOUNT) is equal to VBITSET1. It remains a 1 for all EAV/SAV codes until the EAV at the beginning of the line on when FLCOUNT = VBITCLR1 where it changes to 0 (indicating the start of the field 1 digital active display). The V bit operation is completely independent of the VBLNK control signal.

The VBITSET1 and VBITCLR1 bits should be programmed so that FLCOUNT becomes set to 1 during field 1 vertical blanking. The hardware only starts generating field 1 EDMA events when FLCOUNT = 1.

Figure 4−66. Video Display Field 1 Vertical Blanking Bit Register (VDVBIT1)

31	28	27	16
Reserved		VBITCLR1	
R-0		R/W-0	

15	12	11	0
Reserved		VBITSET1	
R-0		R/W-0	

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4–32. Video Display Field 1 Vertical Blanking Bit Register (VDVBIT1) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	**Raw Data Mode**
31–28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27–16	VBITCLR1	OF(*value*)	0–FFFh	Specifies the first line with an EAV of V = 0 indicating the start of field 1 active display.	Not used.
15–12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11–0	VBITSET1	OF(*value*)	0–FFFh	Specifies the first line with an EAV of V = 1 indicating the start of field 1 vertical blanking.	Not used.

† For CSL implementation, use the notation VP_VDVBIT1_*field_symval*

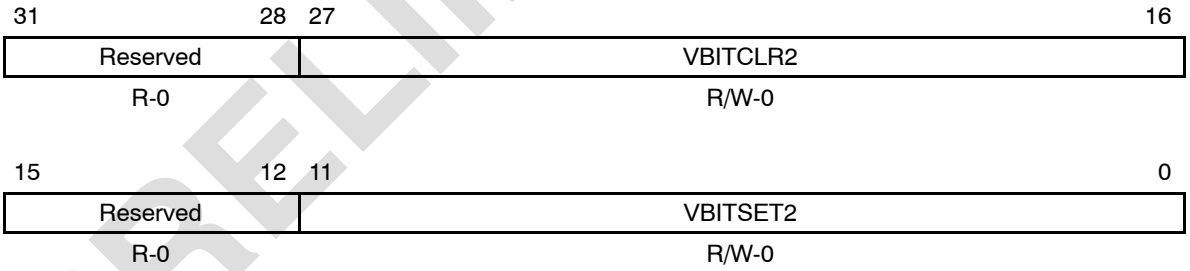
4.12.28 Video Display Field 2 Vertical Blanking Bit Register (VDVBIT2)

The video display field 2 vertical blanking bit register (VDVBIT2) controls the V bit in the EAV and SAV timing control words for field 2. The VDVBIT2 is shown in Figure 4−67 and described in Table 4−33.

The VBITSET2 and VBITCLR2 bits control the V bit value in the EAV and SAV timing control codes. The V bit is set to 1 (indicating the start of field 2 digital vertical blanking) in the EAV code at the beginning of the line whenever the frame line counter (FLCOUNT) is equal to VBITSET2. It remains a 1 for all EAV/SAV codes until the EAV at the beginning of the line on when FLCOUNT = VBITCLR2 where it changes to 0 (indicating the start of the field 2 digital active display). The V bit operation is completely independent of the VBLNK control signal.

For correct interlaced operation, the region defined by VBITSET2 and VBITCLR2 must not overlap the region defined by VBITSET1 and VBITCLR1. For progressive scan operation, VBITSET2 and VBITCLR2 should be programmed to a value greater than FRMHEIGHT.

Figure 4−67. Video Display Field 2 Vertical Blanking Bit Register (VDVBIT2)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 4–33. Video Display Field 2 Vertical Blanking Bit Register (VDVBIT2) Field Descriptions

Bit	*field*†	*symval*†	Value	Description	
				BT.656 and Y/C Mode	Raw Data Mode
31–28	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
27–16	VBITCLR2	OF(*value*)	0–FFFh	Specifies the first line with an EAV of V = 0 indicating the start of field 2 active display.	Not used.
15–12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.	
11–0	VBITSET2	OF(*value*)	0–FFFh	Specifies the first line with an EAV of V = 1 indicating the start of field 2 vertical blanking.	Not used.

† For CSL implementation, use the notation VP_VDVBIT2_*field_symval*

4.13 Video Display Registers Recommended Values

Sample recommended values (decimal) for video display registers for BT.656 output are given in Table 4−34.

Table 4−34. Video Display Register Recommended Values

Register	Field	525/60 Value	625/50 Value
VDFRMSZ	FRMWIDTH	858	864
	FRMHEIGHT	525	625
VDHBLNK	HBLNKSTART	720	720
	HBLNKSTOP	856	862
VDVBLKS1	VBLNKXSTART1	720†	720†
	VBLNKYSTART1	1†	624†
VDVBLKE1	VBLNKXSTOP1	720†	720†
	VBLNKYSTOP1	20†	23†
VDVBLKS2	VBLNKXSTART2	360†	360†
	VBLNKYSTART2	263†	311†
VDVBLKE2	VBLNKXSTOP2	360†	360†
	VBLNKYSTOP2	283†	336†
VDFLDT1	FLD1XSTART	720†	720†
	FLD1YSTART	1†	1†
VDFLDT2	FLD2XSTART	360†	360†
	FLD2YSTART	263†	313†
VDHSYNC	HSYNCSTART	736	732
	HSYNCSTOP	800	782
VDVSYNS1	VSYNCXSTART1	720†	720†
	VSYNCYSTART1	4†	1†
VDVSYNE1	VSYNCXSTOP1	720†	360†
	VSYNCYSTOP1	7†	3†

† Programming only required if external control signal is used.

Table 4–34. Video Display Register Recommended Values (Continued)

Register	Field	525/60 Value	625/50 Value
VDVSYNS2	VSYNCXSTART2	360†	360†
	VSYNCYSTART2	266†	313†
VDVSYNE2	VSYNCXSTOP2	360†	720†
	VSYNCYSTOP2	269†	316†
VDFBIT	FBITCLR	4	1
	FBITSET	266	313
VDVBIT1	VBITSET1	1	624
	VBITCLR1	20	23
VDVBIT2	VBITSET2	264	311
	VBITCLR2	283	336

† Programming only required if external control signal is used.

4.14 Video Display FIFO Registers

The display FIFO mapping registers are listed in Table 4−35. These registers provide DMA write access to the display FIFOs. These pseudo-registers should be mapped into DSP memory space rather than configuration register space in order to provide high-speed access. See the device-specific datasheet for the memory address of these registers.

The function of the video display FIFO mapping registers is listed in Table 4−36.

Table 4−35. Video Display FIFO Registers

Offset Address†	Acronym	Register Name
20h	YDSTA	Y FIFO Destination Register A
28h	CBDST	Cb FIFO Destination Register
30h	CRDST	Cr FIFO Destination Register
20h	YDSTB	Y FIFO Destination Register B

† The absolute address of the registers is device/port specific and is equal to the FIFO base address + offset address. See the device-specific datasheet to verify the register addresses.

Table 4−36. Video Display FIFO Registers Function

	Display Mode	
Register	**BT.656 or Y/C**	**Raw Data**
YDST*x*	Maps Y display FIFO into the DSP memory.	Maps data display buffer into the DSP memory.
CBDST	Maps Cb display FIFO into the DSP memory.	Not used.
CRDST	Maps Cr display FIFO into the DSP memory.	Not used.

In BT.656 or Y/C display mode, three DMAs move data from the DSP memory to Y, Cb, and Cr display FIFOs by using the memory-mapped YDST*x*, CBDST, and CRDST registers. The DMA transfers are triggered by the YEVT, CbEVT, and CrEVT events, respectively.

In raw display mode, one DMA channel moves data from the DSP memory to the Y display FIFO by using the memory-mapped YDST*x* register. The DMA transfers are triggered by a YEVT event.

The video display FIFO registers are write-only locations. Reads of these addresses returns arbitrary values and do not affect the status of the display FIFOs.

Chapter 5

General Purpose I/O Operation

Signals not used for video display or video capture can be used as general-purpose input/output (GPIO) signals.

Topic **Page**

5.1 GPIO Registers

The GPIO register set includes required registers such as peripheral identification and emulation control. The GPIO registers are listed in Table 5−1. See the device-specific datasheet for the memory address of these registers.

Table 5−1. Video Port Registers

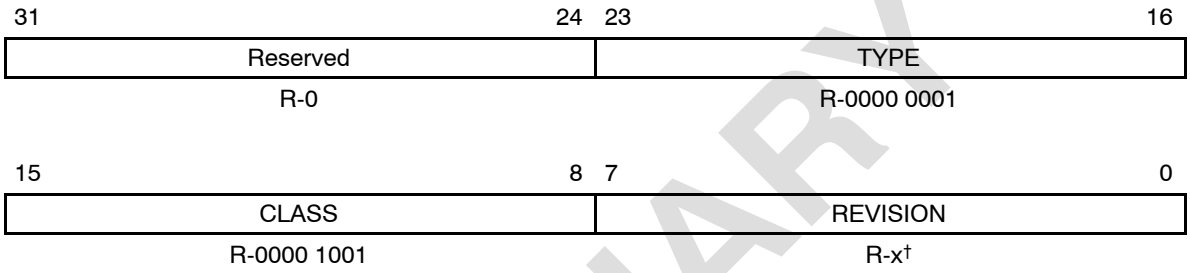
Offset Address†	Acronym	Register Name	Section
00h	VPPID	Video Port Peripheral Identification Register	5.1.1
04h	PCR	Video Port Power Management Register	5.1.2
20h	PFUNC	Video Port Pin Function Register	5.1.3
24h	PDIR	Video Port GPIO Direction Control Register 0	5.1.5
28h	PDIN	Video Port GPIO Data Input Register	5.1.6
2Ch	PDOUT	Video Port GPIO Data Output Register	5.1.7
30h	PDSET	Video Port GPIO Data Set Register	5.1.8
34h	PDCLR	Video Port GPIO Data Clear Register	5.1.8
38h	PIEN	Video Port GPIO Interrupt Enable Register	5.1.9
3Ch	PIPOL	Video Port GPIO Interrupt Polarity Register	5.1.10
40h	PISTAT	Video Port GPIO Interrupt Status Register	5.1.11
44h	PICLR	Video Port GPIO Interrupt Clear Register	5.1.12

† The absolute address of the registers is device/port specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

5.1.1 Video Port Peripheral Identification Register (VPPID)

The video port peripheral identification register (VPPID) is a read-only register used to store information about the peripheral. The VPPID is shown in Figure 5−1 and described in Table 5−2.

Figure 5−1. Video Port Peripheral Identification Register (VPPID)



Legend: R = Read only; -*n* = value after reset
† See the device-specific datasheet for the default value of this field.

Table 5−2. Video Port Peripheral Identification Register (VPPID) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−24	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
23−16	TYPE			Identifies type of peripheral.
		OF(*value*)	01h	Video port.
15−8	CLASS			Identifies class of peripheral.
		OF(*value*)	09h	Video
7−0	REVISION			Identifies revision of peripheral.
		OF(*value*)	x	See the device-specific datasheet for the value.

† For CSL implementation, use the notation VP_VPPID_*field_symval*

5.1.2 Video Port Peripheral Control Register (PCR)

The video port peripheral control register (PCR) determines operation during emulation. The video port peripheral control register is shown in Figure 5–2 and described in Table 5–3.

Normal operation is to not halt the port during emulation suspend. This allows a displayed image to remain visible during suspend. However, this will only work if one of the continuous capture/display modes is selected because non-continuous modes require CPU intervention for DMAs to continue indefinitely (and the CPU is halted during emulation suspend).

When FREE = 0, emulation suspend can occur. Clocks and counters continue to run in order to maintain synchronization with external devices. The video port waits until a field boundary to halt DMA event generation, so that upon restart the video port can begin generating events again at the precise point it left off. After exiting suspend, the video port waits for the correct field boundary to occur and then reenables DMA events. The DMA pointers will be at the correct location for capture/display to resume where it left off. The emulation suspend operation is similar to the BLKCAP or BLKDISP operation with the difference being that BLKCAP and BLKDISP operations take effect immediately rather than at field completion and rely on you to reset the DMA mechanism before they are cleared.

There is no separate emulation suspend mechanism on the video capture side. The field and frame operation (see Table 3–6 on page 3-19) can be used as emulation suspend.

Figure 5–2. Video Port Peripheral Control Register (PCR)

31	16
Reserved	
R-0	

15	3	2	1	0
Reserved		PEREN	SOFT	FREE
R-0		R/W-0	R-0	R/W-1

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 5–3. Video Port Peripheral Control Register (PCR) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–3	Reserved			Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
2	PEREN			Peripheral enable bit.
		DISABLE	0	Video port is disabled. Port clock (VCLK0, VCLK1, STCLK) inputs are gated off to save power. DMA access to the video port is still acknowledged but indeterminate read data is returned and write data is discarded.
		ENABLE	1	Video port is enabled.
1	SOFT			Soft bit enable mode bit. This bit is used in conjunction with FREE bit to determine state of video port clock during emulation suspend. This bit has no effect if FREE = 1.
		STOP	0	The current field is completed upon emulation suspend. After completion, no new DMA events are generated. The port clocks and counters continue to run in order to maintain synchronization. No interrupts are generated. If the port is in display mode, video control signals continue to be output and the default data value is output during the active video window.
		COMP	1	Is not defined for this peripheral; the bit is hardwired to 0.
0	FREE			Free-running enable mode bit. This bit is used in conjunction with SOFT bit to determine state of video port during emulation suspend.
		SOFT	0	Free-running mode is disabled. During emulation suspend, SOFT bit determines operation of video port.
			1	Free-running mode is enabled. Video port ignores the emulation suspend signal and continues to function as normal.

† For CSL implementation, use the notation VP_PCR_*field_symval*

5.1.3 Video Port Pin Function Register (PFUNC)

The video port pin function register (PFUNC) selects the video port pins as GPIO. The PFUNC is shown in Figure 5−3 and described in Table 5−4. Each bit controls either one pin or a set of pins. When a bit is set to 1, it enables the pin(s) that map to it as GPIO. The GPIO feature should not be used for pins that are used as part of the capture or display operation. For pins that have been muxed out for use by another peripheral, the PFUNC bits will have no effect.

The VDATA pins are broken into two functional groups: VDATA[9−0] and VDATA[19−10]. Thus, each entire half of the data bus must be configured as either functional pins or GPIO pins. In the case of single BT.656 or raw 8/10-bit mode, the upper 10 VDATA pins (VDATA[19−10]) can be used as GPIOs. If the video port is disabled, all pins can be used as GPIO.

Figure 5−3. Video Port Pin Function Register (PFUNC)

31	23	22	21	20	19	16
Reserved		PFUNC22	PFUNC21	PFUNC20	Reserved	
R-0		R/W-0	R/W-0	R/W-0	R/W-0	

15	11	10	9	1	0
Reserved		PFUNC10	Reserved		PFUNC0
R-0		R/W-0	R-0		R/W-0

Legend: R = Read only; R/W = Read/Write; *-n* = value after reset

Table 5−4. Video Port Pin Function Register (PFUNC) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PFUNC22			PFUNC22 bit determines if VCTL2 pin functions as GPIO.
		NORMAL	0	Pin functions normally.
		VCTL2	1	Pin functions as GPIO pin.
21	PFUNC21			PFUNC21 bit determines if VCTL1 pin functions as GPIO.
		NORMAL	0	Pin functions normally.
		VCTL1	1	Pin functions as GPIO pin.

† For CSL implementation, use the notation VP_PFUNC_*field_symval*

Table 5–4. Video Port Pin Function Register (PFUNC) Field Descriptions (Continued)

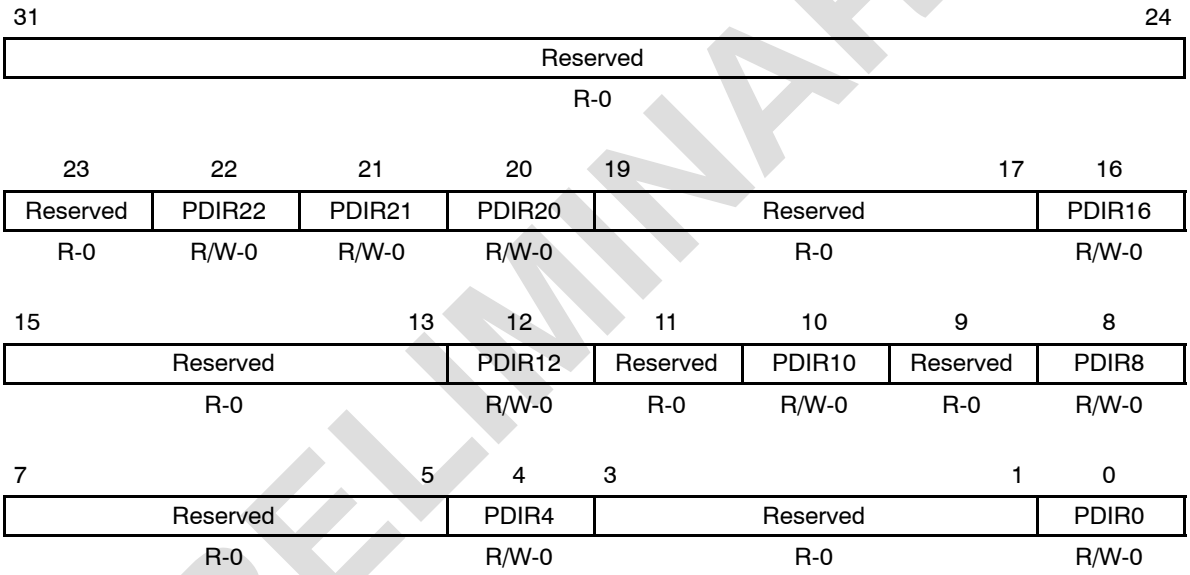
Bit	*field*†	*symval*†	Value	Description
20	PFUNC20			PFUNC20 bit determines if VCTL0 pin functions as GPIO.
		NORMAL	0	Pin functions normally.
		VCTL0	1	Pin functions as GPIO pin.
19–11	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
10	PFUNC10			PFUNC10 bit determines if VDATA[19–10] pins function as GPIO.
		NORMAL	0	Pins function normally.
		VDATA10TO19	1	Pins function as GPIO pin.
9–1	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
0	PFUNC0			PFUNC0 bit determines if VDATA[9–0] pins function as GPIO.
		NORMAL	0	Pins function normally.
		VDATA0TO9	1	Pins function as GPIO pin.

† For CSL implementation, use the notation VP_PFUNC_*field_symval*

5.1.4 Video Port Pin Direction Register (PDIR)

The video port pin direction register (PDIR) is shown in Figure 5−4 and described in Table 5−5. The PDIR controls the direction of IO pins in the video port for those pins set by PFUNC. If a bit is set to 1, the relevant pin or pin group acts as an output. If a bit is cleared to 0, the pin or pin group functions as an input. The PDIR settings do not affect pins where the corresponding PFUNC bit is not set.

Figure 5−4. Video Port Pin Direction Register (PDIR)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 5−5. Video Port Pin Direction Register (PDIR) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PDIR22			PDIR22 bit controls the direction of the VCTL2 pin.
		VCTL2IN	0	Pin functions as input.
		VCTL2OUT	1	Pin functions as output.

† For CSL implementation, use the notation VP_PDIR_*field_symval*

Table 5–5. Video Port Pin Direction Register (PDIR) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
21	PDIR21			PDIR21 bit controls the direction of the VCTL1 pin.
		VCTL1IN	0	Pin functions as input.
		VCTL1OUT	1	Pin functions as output.
20	PDIR20			PDIR20 bit controls the direction of the VCTL0 pin.
		VCTL0IN	0	Pin functions as input.
		VCTL0OUT	1	Pin functions as output.
19–17	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
16	PDIR16			PDIR16 bit controls the direction of the VDATA[19–16] pins.
		VDATA16TO19IN	0	Pins function as input.
		VDATA16TO19OUT	1	Pins function as output.
15–13	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
12	PDIR12			PDIR12 bit controls the direction of the VDATA[15–12] pins.
		VDATA12TO15IN	0	Pins function as input.
		VDATA12TO15OUT	1	Pins function as output.
11	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
10	PDIR10			PDIR10 bit controls the direction of the VDATA[11–10] pins.
		VDATA10TO11IN	0	Pins function as input.
		VDATA10TO11OUT	1	Pins function as output.
9	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation VP_PDIR_*field_symval*

Table 5–5. Video Port Pin Direction Register (PDIR) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
8	PDIR8			PDIR8 bit controls the direction of the VDATA[9–8] pins.
		VDATA8TO9IN	0	Pins function as input.
		VDATA8TO9OUT	1	Pins function as output.
7–5	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
4	PDIR4			PDIR4 bit controls the direction of the VDATA[7–4] pins.
		VDATA4TO7IN	0	Pins function as input.
		VDATA4TO7OUT	1	Pins function as output.
3–1	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
0	PDIR0			PDIR0 bit controls the direction of the VDATA[3–0] pins.
		VDATA0TO3IN	0	Pins function as input.
		VDATA0TO3OUT	1	Pins function as output.

† For CSL implementation, use the notation VP_PDIR_*field_symval*

5.1.5 Video Port Pin Data Input Register (PDIN)

The read-only video port pin data input register (PDIN) is shown in Figure 5–5 and described in Table 5–6. PDIN reflects the state of the video port pins. When read, PDIN returns the value from the pin's input buffer (with appropriate synchronization) regardless of the state of the corresponding PFUNC or PDIR bit.

Figure 5–5. Video Port Pin Data Input Register (PDIN)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PDIN22	PDIN21	PDIN20	PDIN19	PDIN18	PDIN17	PDIN16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

15	14	13	12	11	10	9	8
PDIN15	PDIN14	PDIN13	PDIN12	PDIN11	PDIN10	PDIN9	PDIN8
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
PDIN7	PDIN6	PDIN5	PDIN4	PDIN3	PDIN2	PDIN1	PDIN0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

Legend: R = Read only; -*n* = value after reset

Table 5−6. Video Port Pin Data Input Register (PDIN) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PDIN22			PDIN22 bit returns the logic level of the VCTL2 pin.
		VCTL2LO	0	Pin is logic low.
		VCTL2HI	1	Pin is logic high.
21	PDIN21			PDIN21 bit returns the logic level of the VCTL1 pin.
		VCTL1LO	0	Pin is logic low.
		VCTL1HI	1	Pin is logic high.
20	PDIN20			PDIN20 bit returns the logic level of the VCTL0 pin.
		VCTL0LO	0	Pin is logic low.
		VCTL0HI	1	Pin is logic high.
19−0	PDIN[19−0]			PDIN[19−0] bit returns the logic level of the corresponding VDATA[*n*] pin.
		VDATA*n*LO	0	Pin is logic low.
		VDATA*n*HI	1	Pin is logic high.

† For CSL implementation, use the notation VP_PDIN_PDIN*n_symval*

5.1.6 Video Port Pin Data Output Register (PDOUT)

The video port pin data output register (PDOUT) is shown in Figure 5−6 and described in Table 5−7. The bits of PDOUT determine the value driven on the corresponding GPIO pin, if the pin is configured as an output. Writes do not affect pins not configured as GPIO outputs. The bits in PDOUT are set or cleared by writing to this register directly. A read of PDOUT returns the value of the register not the value at the pin (that might be configured as an input). An alternative way to set bits in PDOUT is to write a 1 to the corresponding bit of PDSET. An alternative way to clear bits in PDOUT is to write a 1 to the corresponding bit of PDCLR.

PDOUT has these aliases:

- PDSET — writing a 1 to a bit in PDSET sets the corresponding bit in PDOUT to 1; writing a 0 has no effect and keeps the bits in PDOUT unchanged.
- PDCLR — writing a 1 to a bit in PDCLR clears the corresponding bit in PDOUT to 0; writing a 0 has no effect and keeps the bits in PDOUT unchanged.

Figure 5−6. Video Port Pin Data Output Register (PDOUT)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PDOUT22	PDOUT21	PDOUT20	PDOUT19	PDOUT18	PDOUT17	PDOUT16
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8
PDOUT15	PDOUT14	PDOUT13	PDOUT12	PDOUT11	PDOUT10	PDOUT9	PDOUT8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
PDOUT7	PDOUT6	PDOUT5	PDOUT4	PDOUT3	PDOUT2	PDOUT1	PDOUT0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 5−7. Video Port Pin Data Out Register (PDOUT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PDOUT22			PDOUT22 bit drives the VCTL2 pin only when the GPIO is configured as output. When reading data, returns the bit value in PDOUT22, does not return input from pin. When writing data, writes to PDOUT22 bit.
		VCTL2LO	0	Pin drives low.
		VCTL2HI	1	Pin drives high.
21	PDOUT21			PDOUT21 bit drives the VCTL1 pin only when the GPIO is configured as output. When reading data, returns the bit value in PDOUT21, does not return input from pin. When writing data, writes to PDOUT21 bit.
		VCTL1LO	0	Pin drives low.
		VCTL1HI	1	Pin drives high.
20	PDOUT20			PDOUT20 bit drives the VCTL0 pin only when the GPIO is configured as output. When reading data, returns the bit value in PDOUT20, does not return input from pin. When writing data, writes to PDOUT20 bit.
		VCTL0LO	0	Pin drives low.
		VCTL0HI	1	Pin drives high.
19−0	PDOUT[19−0]			PDOUT[19−0] bit drives the corresponding VDATA[19−0] pin only when the GPIO is configured as output. When reading data, returns the bit value in PDOUT[*n*], does not return input from pin. When writing data, writes to PDOUT[*n*] bit.
		VDATA*n*LO	0	Pin drives low.
		VDATA*n*HI	1	Pin drives high.

† For CSL implementation, use the notation VP_PDOUT_PDOUT*n_symval*

5.1.7 Video Port Pin Data Set Register (PDSET)

The video port pin data set register (PDSET) is shown in Figure 5−7 and described in Table 5−8. PDSET is an alias of the video port pin data output register (PDOUT) for writes only and provides an alternate means of driving GPIO outputs high. Writing a 1 to a bit of PDSET sets the corresponding bit in PDOUT. Writing a 0 has no effect. Register reads return all 0s.

Figure 5−7. Video Port Pin Data Set Register (PDSET)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PDSET22	PDSET21	PDSET20	PDSET19	PDSET18	PDSET17	PDSET16
R-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

15	14	13	12	11	10	9	8
PDSET15	PDSET14	PDSET13	PDSET12	PDSET11	PDSET10	PDSET9	PDSET8
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

7	6	5	4	3	2	1	0
PDSET7	PDSET6	PDSET5	PDSET4	PDSET3	PDSET2	PDSET1	PDSET0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

Legend: R = Read only; W = Write only; *-n* = value after reset

Table 5−8. Video Port Pin Data Set Register (PDSET) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PDSET22			Allows PDOUT22 bit to be set to a logic high without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VCTL2HI	1	Sets PDOUT22 (VCTL2) bit to 1.
21	PDSET21			Allows PDOUT21 bit to be set to a logic high without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VCTL1HI	1	Sets PDOUT21 (1) bit to 1.
20	PDSET20			Allows PDOUT20 bit to be set to a logic high without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VCTL0HI	1	Sets PDOUT20 (VCTL0) bit to 1.
19−0	PDSET[19−0]			Allows PDOUT[19−0] bit to be set to a logic high without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VDATA*n*HI	1	Sets PDOUT[*n*] (VDATA[*n*]) bit to 1.

† For CSL implementation, use the notation VP_PDSET_PDSET*n_symval*

5.1.8 Video Port Pin Data Clear Register (PDCLR)

The video port pin data clear register (PDCLR) is shown in Figure 5−8 and described in Table 5−9. PDCLR is an alias of the video port pin data output register (PDOUT) for writes only and provides an alternate means of driving GPIO outputs low. Writing a 1 to a bit of PDCLR clears the corresponding bit in PDOUT. Writing a 0 has no effect. Register reads return all 0s.

Figure 5−8. Video Port Pin Data Clear Register (PDCLR)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PDCLR22	PDCLR21	PDCLR20	PDCLR19	PDCLR18	PDCLR17	PDCLR16
R-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

15	14	13	12	11	10	9	8
PDCLR15	PDCLR14	PDCLR13	PDCLR12	PDCLR11	PDCLR10	PDCLR9	PDCLR8
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

7	6	5	4	3	2	1	0
PDCLR7	PDCLR6	PDCLR5	PDCLR4	PDCLR3	PDCLR2	PDCLR1	PDCLR0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

Legend: R = Read only; W = Write only; -*n* = value after reset

Table 5−9. Video Port Pin Data Clear Register (PDCLR) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PDCLR22			Allows PDOUT22 bit to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VCTL2CLR	1	Clears PDOUT22 (VCTL2) bit to 0.
21	PDCLR21			Allows PDOUT21 bit to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VCTL1CLR	1	Clears PDOUT21 (VCTL1) bit to 0.
20	PDCLR20			Allows PDOUT20 bit to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VCTL0CLR	1	Clears PDOUT20 (VCTL0) bit to 0.
19−0	PDCLR[19−0]			Allows PDOUT[19−0] bit to be cleared to a logic low without affecting other I/O pins controlled by the same port.
		NONE	0	No effect.
		VDATA*n*CLR	1	Clears PDOUT[*n*] (VDATA[*n*]) bit to 0.

† For CSL implementation, use the notation VP_PDCLR_PDCLR*n_symval*

5.1.9 Video Port Pin Interrupt Enable Register (PIEN)

The video port pin interrupt enable register (PIEN) is shown in Figure 5−9 and described in Table 5−10. The GPIOs can be used to generate DSP interrupts or DMA events. The PIEN selects which pins may be used to generate an interrupt. Only pins whose corresponding bits in PIEN are set may cause their corresponding PISTAT bit to be set.

Interrupts are enabled on a GPIO pin when the corresponding bit in PIEN is set, the pin is enabled for GPIO in PFUNC, and the pin is configured as an input in PDIR.

Figure 5−9. Video Port Pin Interrupt Enable Register (PIEN)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PIEN22	PIEN21	PIEN20	PIEN19	PIEN18	PIEN17	PIEN16
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8
PIEN15	PIEN14	PIEN13	PIEN12	PIEN11	PIEN10	PIEN9	PIEN8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
PIEN7	PIEN6	PIEN5	PIEN4	PIEN3	PIEN2	PIEN1	PIEN0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 5−10. Video Port Pin Interrupt Enable Register (PIEN) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PIEN22			PIEN22 bit enables the interrupt on the VCTL2 pin.
		VCTL2LO	0	Interrupt is disabled.
		VCTL2HI	1	Pin enables the interrupt.
21	PIEN21			PIEN21 bit enables the interrupt on the VCTL1 pin.
		VCTL1LO	0	Interrupt is disabled.
		VCTL1HI	1	Pin enables the interrupt.
20	PIEN20			PIEN20 bit enables the interrupt on the VCTL0 pin.
		VCTL0LO	0	Interrupt is disabled.
		VCTL0HI	1	Pin enables the interrupt.
19−0	PIEN[19−0]			PIEN[19−0] bits enable the interrupt on the corresponding VDATA[*n*] pin.
		VDATA*n*LO	0	Interrupt is disabled.
		VDATA*n*HI	1	Pin enables the interrupt.

† For CSL implementation, use the notation VP_PIEN_PIEN*n_symval*

5.1.10 Video Port Pin Interrupt Polarity Register (PIPOL)

The video port pin interrupt polarity register (PIPOL) is shown in Figure 5−10 and described in Table 5−11. The PIPOL determines the GPIO pin signal polarity that generates an interrupt.

Figure 5−10. Video Port Pin Interrupt Polarity Register (PIPOL)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PIPOL22	PIPOL21	PIPOL20	PIPOL19	PIPOL18	PIPOL17	PIPOL16
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8
PIPOL15	PIPOL14	PIPOL13	PIPOL12	PIPOL11	PIPOL10	PIPOL9	PIPOL8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
PIPOL7	PIPOL6	PIPOL5	PIPOL4	PIPOL3	PIPOL2	PIPOL1	PIPOL0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 5−11. Video Port Pin Interrupt Polarity Register (PIPOL) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PIPOL22			PIPOL22 bit determines the VCTL2 pin signal polarity that generates an interrupt.
		VCTL2ACTHI	0	Interrupt is caused by a low-to-high transition on the VCTL2 pin.
		VCTL2ACTLO	1	Interrupt is caused by a high-to-low transition on the VCTL2 pin.
21	PIPOL21			PIPOL21 bit determines the VCTL1 pin signal polarity that generates an interrupt.
		VCTL1ACTHI	0	Interrupt is caused by a low-to-high transition on the VCTL1 pin.
		VCTL1ACTLO	1	Interrupt is caused by a high-to-low transition on the VCTL1 pin.
20	PIPOL20			PIPOL20 bit determines the VCTL0 pin signal polarity that generates an interrupt.
		VCTL0ACTHI	0	Interrupt is caused by a low-to-high transition on the VCTL0 pin.
		VCTL0ACTLO	1	Interrupt is caused by a high-to-low transition on the VCTL0 pin.
19−0	PIPOL[19−0]			PIPOL[19−0] bit determines the corresponding VDATA[*n*] pin signal polarity that generates an interrupt.
		VDATA*n*ACTHI	0	Interrupt is caused by a low-to-high transition on the VDATA[*n*] pin.
		VDATA*n*ACTLO	1	Interrupt is caused by a high-to-low transition on the VDATA[*n*] pin.

† For CSL implementation, use the notation VP_PIPOL_PIPOL*n_symval*

5.1.11 Video Port Pin Interrupt Status Register (PISTAT)

The video port pin interrupt status register (PISTAT) is shown in Figure 5−11 and described in Table 5−12. PISTAT is a read-only register that indicates the GPIO pin that has a pending interrupt.

A bit in PISTAT is set when the corresponding GPIO pin is configured as an interrupt (the corresponding bit in PIEN is set, the pin is enabled for GPIO in PFUNC, and the pin is configured as an input in PDIR) and the appropriate transition (as selected by the corresponding PIPOL bit) occurs on the pin. Whenever a PISTAT bit is set to 1, the GPIO bit in VPIS is set. The PISTAT bits are cleared by writing a 1 to the corresponding bit in PICLR. Writing a 0 has no effect. Clearing all the PISTAT bits does not clear the GPIO bit in VPIS, it must be explicitly cleared. If any bits in PISTAT are still set when the GPIO bit is cleared, the GPIO bit is set again.

Figure 5−11. Video Port Pin Interrupt Status Register (PISTAT)

31	30	29	28	27	26	25	24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PISTAT22	PISTAT21	PISTAT20	PISTAT19	PISTAT18	PISTAT17	PISTAT16
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

15	14	13	12	11	10	9	8
PISTAT15	PISTAT14	PISTAT13	PISTAT12	PISTAT11	PISTAT10	PISTAT9	PISTAT8
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
PISTAT7	PISTAT6	PISTAT5	PISTAT4	PISTAT3	PISTAT2	PISTAT1	PISTAT0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

Legend: R = Read only; -*n* = value after reset

Table 5−12. Video Port Pin Interrupt Status Register (PISTAT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PISTAT22			PISTAT22 bit indicates if there is a pending interrupt on the VCTL2 pin.
		NONE	0	No pending interrupt on the VCTL2 pin.
		VCTL2INT	1	Pending interrupt on the VCTL2 pin.
21	PISTAT21			PISTAT21 bit indicates if there is a pending interrupt on the VCTL1 pin.
		NONE	0	No pending interrupt on the VCTL1 pin.
		VCTL1INT	1	Pending interrupt on the VCTL1 pin.
20	PISTAT20			PISTAT20 bit indicates if there is a pending interrupt on the VCTL0 pin.
		NONE	0	No pending interrupt on the VCTL0 pin.
		VCTL0INT	1	Pending interrupt on the VCTL0 pin.
19−0	PISTAT[19−0]			PISTAT[19−0] bit indicates if there is a pending interrupt on the corresponding VDATA[*n*] pin.
		NONE	0	No pending interrupt on the VDATA[*n*] pin.
		VDATA*n*INT	1	Pending interrupt on the VDATA[*n*] pin.

† For CSL implementation, use the notation VP_PISTAT_PISTAT*n_symval*

5.1.12 Video Port Pin Interrupt Clear Register (PICLR)

The video port pin interrupt clear register (PICLR) is shown in Figure 5−12 and described in Table 5−13. PICLR is an alias of the video port pin interrupt status register (PISTAT) for writes only. Writing a 1 to a bit of PICLR clears the corresponding bit in PISTAT. Writing a 0 has no effect. Register reads return all 0s.

Figure 5−12. Video Port Pin Interrupt Clear Register (PICLR)

31							24
Reserved							
R-0							

23	22	21	20	19	18	17	16
Reserved	PICLR22	PICLR21	PICLR20	PICLR19	PICLR18	PICLR17	PICLR16
R-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

15	14	13	12	11	10	9	8
PICLR15	PICLR14	PICLR13	PICLR12	PICLR11	PICLR10	PICLR9	PICLR8
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

7	6	5	4	3	2	1	0
PICLR7	PICLR6	PICLR5	PICLR4	PICLR3	PICLR2	PICLR1	PICLR0
W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0

Legend: R = Read only; W = Write only; -*n* = value after reset

Table 5−13. Video Port Pin Interrupt Clear Register (PICLR) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−23	Reserved	−	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22	PICLR22			Allows PISTAT22 bit to be cleared to a logic low.
		NONE	0	No effect.
		VCTL2CLR	1	Clears PISTAT22 (VCTL2) bit to 0.
21	PICLR21			Allows PISTAT21 bit to be cleared to a logic low.
		NONE	0	No effect.
		VCTL1CLR	1	Clears PISTAT21 (VCTL1) bit to 0.
20	PICLR20			Allows PISTAT20 bit to be cleared to a logic low.
		NONE	0	No effect.
		VCTL0CLR	1	Clears PISTAT20 (VCTL0) bit to 0.
19−0	PICLR[19−0]			Allows PISTAT[19−0] bit to be cleared to a logic low.
		NONE	0	No effect.
		VDATA*n*CLR	1	Clears PISTAT[*n*] (VDATA[*n*]) bit to 0.

† For CSL implementation, use the notation VP_PICLR_PICLR*n_symval*

Chapter 6

VCXO Interpolated Control Port

This chapter provides an overview of the VCXO interpolated control (VIC) port.

Topic **Page**

6.1 Overview

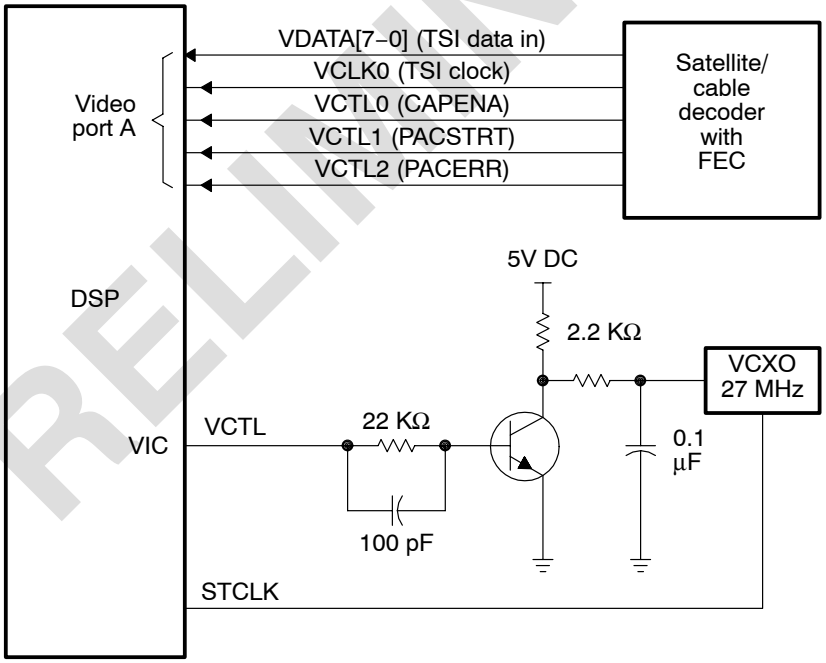
The VCXO interpolated control (VIC) port provides single-bit interpolated VCXO control with resolution from 9 bits to up to 16 bits. The frequency of interpolation is dependent on the resolution needed.

When the video port is used in transport stream interface (TSI) mode, the VIC port is used to control the system clock, VCXO, for MPEG transport stream (Figure 6–1).

The VIC port supports following features:

- Single-bit interpolated VCXO control
- Programmable precision from 9 to 16 bits

Figure 6–1. TSI System Block Diagram



6.2 Interface

The pin list for VIC port is shown in Table 6−1 (pins are 3.3V I/Os).

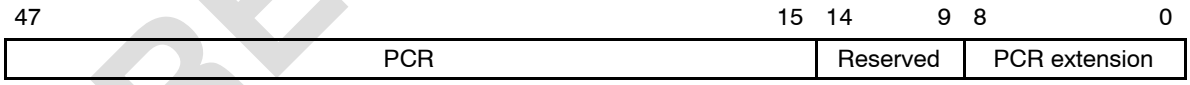
Table 6−1. VIC Port Interface Signals

VIC Port Signal	Direction	Description
VCTL	Output	VCXO control
STCLK	Input	System time clock

6.3 Operational Details

Synchronization is an important aspect of decoding and presenting data in real-time digital data delivery systems. This is addressed in the MPEG transport packets by transmitting timing information in the adaptation fields of selected data packets. This serves as a reference for timing comparison in the receiving system. A sample of the 27-MHz clock, the program clock reference (PCR) header is shown in Figure 6−2, is transmitted within the bit stream, which indicates the expected time at the completion of reading the field from the bit stream at the transport decoder. The sample is a 42-bit field, 9 bits cycle from 0 to 299 at 27 MHz, while the other 33-bit field is incremented by 1 each time the 9-bit field reaches a value of 299. The transport data packets are in sync with the server system clock.

Figure 6−2. Program Clock Reference (PCR) Header Format



The video port in conjunction with the VIC port uses a combined hardware and software solution to synchronize the transport system time clock (STC) with the clock reference transmitted in the bitstream.

The video port maintains a hardware counter that counts the system time. The counter is driven by system time clock (STCLK) input driven by an external VCXO, controlled by the VIC port.

On reception of a packet, the video port captures a snapshot of the counter. Software uses this timestamp to determine the deviation of the system time clock from the server clock, and drives VCTL output of the VIC port to keep it synchronized.

Any time a packet with a PCR is received, the timestamp for that packet is compared with the PCR value in software. A PLL is implemented in software to synchronize the STCLK with the system time clock. The DSP updates the VIC input register (VICIN) using the output from this algorithm, which in turn drives the VCTL output that controls the system time clock VCXO.

If *f* is the frequency of PCRs in the incoming bit stream, the interpolation rate *R* of the VCTL output is given in Equation 6–1, where *k* is determined by the precision β specified by you.

Equation 6–1. Relationship Between Interpolation Rate and Input Frequency

$$R = kf$$

Equation 6–2 gives the relation between *k* and the precision β.

Equation 6–2. Relationship of Frequency Multiplier to Precision

$$k > \sqrt[3]{((\pi^2(2^\beta - 1)^2)/3)}$$

Table 6–2 gives some *k* and R values for different β's with *f* fixed at 40 kHz. Once a suitable interpolation frequency is determined, the clock divider can be set.

Table 6–2. Example Values for Interpolation Rate

β	k	R
9	96.0	3.8 MHz
10	151.0	6.0 MHz
11	240.0	9.6 MHz
12	381.0	15.2 MHz
13	605.0	24.2 MHz
14	960.0	38.4 MHz
15	1523.0	60.9 MHz
16	2418.0	96.7 MHz

6.4 Enabling VIC Port

Perform the following steps to enable the VIC port.

1) Clear the GO bit in the VIC control register (VICCTL) to 0.
2) Set the PRECISION bits in VICCTL to the desired precision.
3) Set the VIC clock divider register (VICDIV) bits to appropriate value based on the precision and interpolation frequency.
4) Set the GO bit in VICCTL to 1.
5) The VIC input register (VICIN) is written into every time a new input code is available for interpolation. Repeat step 3 as often as needed.

6.5 VIC Port Registers

The VIC port registers are listed in Table 6–3. See the device-specific datasheet for the memory address of these registers.

Table 6–3. VIC Port Registers

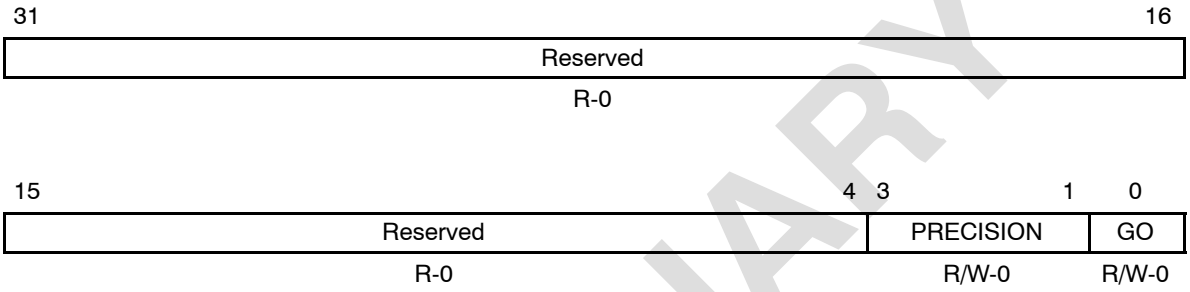
Offset Address†	Acronym	Register Name	Section
00h	VICCTL	VIC Control Register	6.5.1
04h	VICIN	VIC Input Register	6.5.2
08h	VICDIV	VIC Clock Divider Register	6.5.3

† The absolute address of the registers is device specific and is equal to the base address + offset address. See the device-specific datasheet to verify the register addresses.

6.5.1 VIC Control Register (VICCTL)

The VIC control register (VICCTL) is shown in Figure 6−3 and described in Table 6−4.

Figure 6−3. VIC Control Register (VICCTL)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 6−4. VIC Control Register (VICCTL) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−4	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3−1	PRECISION		0−7h	Precision bits determine the resolution of the interpolation. The PRECISION bits can only be written when the GO bit is cleared to 0. If the GO bit is set to 1, a write to the PRECISION bits does not change the bits.
		16BITS	0	16 bits
		15BITS	1	15 bits
		14BITS	2h	14 bits
		13BITS	3h	13 bits
		12BITS	4h	12 bits
		11BITS	5h	11 bits
		10BITS	6h	10 bits
		9BITS	7h	9 bits

† For CSL implementation, use the notation VIC_VICCTL_*field_symval*

Table 6–4. VIC Control Register (VICCTL) Field Descriptions (Continued)

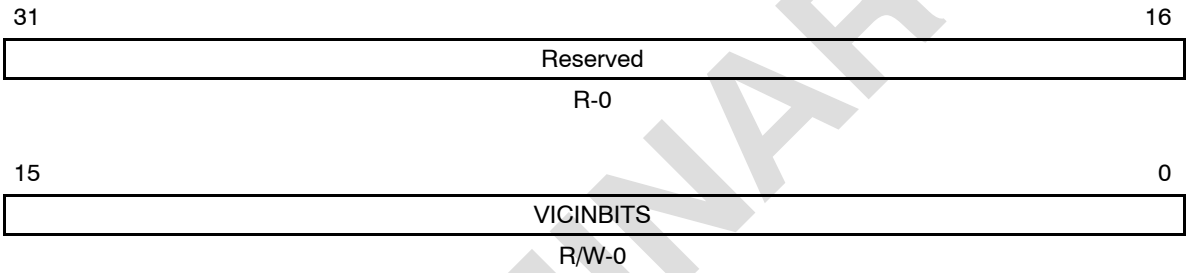
Bit	*field*†	*symval*†	Value	Description
0	GO			The GO bit can be written to at any time.
		0	0	The VICDIV and VICCTL registers can be written to without affecting the operation of the VIC port. All the logic in the VIC port is held in reset state and a 0 is output on the VCTL output line. A write to VICCTL bits as well as setting GO to 1 is allowed in a single write operation. The VICCTL bits change and the GO bit is set, disallowing any further changes to the VICCTL and VICDIV registers.
		1	1	The VICDIV and VICCTL (except for the GO bit) registers cannot be written. If a write is performed to the VICDIV or VICCTL registers when the GO bit is set, the values of these registers remain unchanged. If a write is performed that clears the GO bit to 0 and changes the values of other VICCTL bits, it results in GO = 0 while keeping the rest of the VICCTL bits unchanged. The VIC port is in its normal working mode in this state.

† For CSL implementation, use the notation VIC_VICCTL_*field_symval*

6.5.2 VIC Input Register (VICIN)

The DSP writes the input bits for VCXO interpolated control in the VIC input register (VICIN). The DSP decides how often to update VICIN. The DSP can write to VICIN only when the GO bit in the VIC control register (VICCTL) is set to 1. The VIC module uses the MSBs of VICIN for precision values less than 16. The VICIN is shown in Figure 6−4 and described in Table 6−5.

Figure 6−4. VIC Input Register (VICIN)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 6−5. VIC Input Register (VICIN) Field Descriptions

Bit	Field	*symval*†	Value	Description
31−16	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15−0	VICINBITS	OF(*value*)	0−FFFFh	The DSP writes the input bits for VCXO interpolated control to the VIC input bits.

† For CSL implementation, use the notation VIC_VICIN_VICINBITS_*symval*

6.5.3 VIC Clock Divider Register (VICDIV)

The VIC clock divider register (VICDIV) defines the clock divider for the VIC interpolation frequency. The VIC interpolation frequency is obtained by dividing the module clock. The divider value written to VICDIV is:

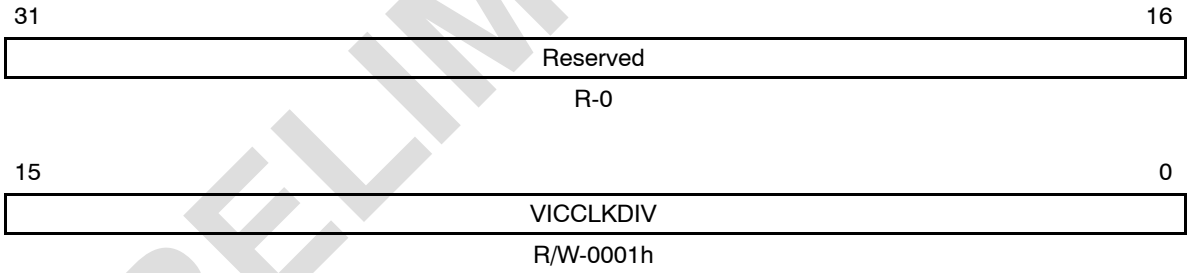
$$Divider = Round[DCLK/R]$$

where *DCLK* is the CPU clock divided by 2, and *R* is the desired interpolation frequency. The interpolation frequency depends on precision β.

The default value of VICDIV is 0001h; 0000h is an illegal value. The VIC module uses a value of 0001h whenever 0000h is written to this register.

The DSP can write to VICDIV only when the GO bit in VICCTL is cleared to 0. If a write is performed when the GO bit is set to 1, the VICDIV bits remain unchanged. The VICDIV is shown in Figure 6–5 and described in Table 6–6.

Figure 6–5. VIC Clock Divider Register (VICDIV)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 6–6. VIC Clock Divider Register (VICDIV) Field Descriptions

Bit	Field	*symval*†	Value	Description
31–16	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15–0	VICCLKDIV	OF(*value*)	0–FFFFh	The VIC clock divider bits define the clock divider for the VIC interpolation frequency.

† For CSL implementation, use the notation VIC_VICDIV_VICCLKDIV_*symval*

Appendix A

Revision History

Table A−1 lists the changes made since the previous version of this document.

Table A−1. Document Revision History

Page	Additions/Modifications/Deletions
1-5	Changed the second sentence in third paragraph of section 1.2.1 to: This means that the transfer size must be equal to the threshold so that it fits into the available space.
2-7	Deleted Figure 2−1.
2-9	Deleted Figure 2−2. Subsequent figures renumbered.
2-12	Deleted Section 2.4, Clocks.
2-12	Deleted Section 2.5, Video Port Functionality Subsets
2-13	Deleted Section 2.6, Video Port Throughput and Latency. Subsequent sections renumbered.
2-16	Added Offset Address column and footnote to Table 2−4.
3-52	Added Offset Address column and footnote to Table 3−13.
3-86	Added Offset Address column and footnote to Table 3−34.
4-54	Added Offset Address column and footnote to Table 4−5.
4-98	Added Offset Address column and footnote to Table 4−35.
5-2	Added Offset Address column and footnote to Table 5−1.
6-5	Added Offset Address column and footnote to Table 6−3.

Appendix B

Video Port Configuration Examples

This appendix describes how to configure the video port in different modes with the help of examples. All examples in this appendix use the video port Chip Support Library (CSL).

Topic **Page**

B.1 Example 1: Noncontinuous Frame Capture for 525/60 Format

This is an example that explains how to configure the video port for 8-bit BT.656 noncontinuous frame capture on channel A for 525/60 format. See ITU−R BT.656-4 and video port specification (Figures 4−11, 4−33, 4−34, and Table 4−37) for more details on 525/60 format.

```
/*********************************************************/
/* Capture parameter definitions based on 525/60 format  */
/*********************************************************/
#define VCA_HBLNK_SIZE     138 /* (858-720),horizontal blanking         */
    /* (EAV/SAV inclusive)                                            */
#define VCA_VBLNK1_SIZE    19  /* (20-1),v.blanking for field1          */
#define VCA_VBLNK2_SIZE    19  /* (283-264),v.blanking for field2       */
#define VCA_IMG_HSIZE1     720 /* field1 horizontal image size          */
#define VCA_IMG_VSIZE1     244 /* (263-20+1), fld1 vertical image size  */
#define VCA_IMG_HSIZE2     720 /* field2 horizontal image size          */
#define VCA_IMG_VSIZE2     243 /* (525-283+1), fld2 vertical image size */
/* Define field image sizes */
#define VCA_IMAGE_SIZE1    (VCA_IMG_HSIZE1 * VCA_IMG_VSIZE1)
#define VCA_IMAGE_SIZE2    (VCA_IMG_HSIZE2 * VCA_IMG_VSIZE2)

/* ------------------------------------------------------ */
/* Define channel A capture window co-ordinates for Field1 */
/* ------------------------------------------------------ */
/* HRST = 0, start of horizontal blanking */
#define VCA_XSTART1        (VCA_HBLNK_SIZE - 2/*EAV*/)
/* VRST = 1, end of vertical blanking */
#define VCA_YSTART1        1
#define VCA_XSTOP1         (VCA_XSTART1 + VCA_IMG_HSIZE1 - 1)
#define VCA_YSTOP1         (VCA_YSTART1 + VCA_IMG_VSIZE1 - 1)

/* ------------------------------------------------------ */
/* Define channel A capture window co-ordinates for Field2 */
/* ------------------------------------------------------ */
/* HRST = 0, start of horizontal blanking                         */
#define VCA_XSTART2        (VCA_HBLNK_SIZE - 2/*EAV*/)
/* VRST = 1, end of vertical blanking                             */
#define VCA_YSTART2        1
#define VCA_XSTOP2         (VCA_XSTART2 + VCA_IMG_HSIZE2 - 1)
#define VCA_YSTOP2         (VCA_YSTART2 + VCA_IMG_VSIZE2 - 1)
/* Define threshold values in double-words. Both fields should    */
/* same threshold value)                                          */
#define VCA_THRLD_FIELD1   (VCA_IMG_HSIZE1/8) /* line length in   */
#define VCA_THRLD_FIELD2   VCA_THRLD_FIELD1   /* double-words     */
/* Define number of events to be generated for field1 and field2  */
#define VCA_CAPEVT1        (VCA_IMAGE_SIZE1 / (VCA_VDTHRLD1 * 8))
#define VCA_CAPEVT2        (VCA_IMAGE_SIZE2 / (VCA_VDTHRLD2 * 8))
#define CAPCHA_FRAME_COUNT 5 /* in this example                   */
```

```
/* ---------------------------------------- */
/* EDMA parameters for capture Y event that are */
/* specific to this example.                */
/* ---------------------------------------- */
#define VCA_Y_EDMA_ELECNT  (VCA_THRLD_FIELD1 * 2)  /* because VCA_THRLD_FIELDn is
in double-words and element size is 32-bit */
#define VCA_Y_EDMA_FRMCNT  ((VCA_CAPEVT1 + VCA_CAPEVT2) * CAPCHA_FRAME_COUNT)

/*****************************************************************/
/* Description :  8-bit BT.656 non-continuous frame capture      */
/*                                                               */
/* Some important field descriptions:                            */
/*                                                               */
/* CMODE = 000, 8-bit BT.656 mode                                */
/* CON   = 0                                                     */
/* FRAME = 1, capture frame                                      */
/* CF2   = 0                                                     */
/* CF1   = 0, (8-bit non-continuous frame capture)               */
/* SCALE = 0, no scaling                                         */
/* RESMPL= 0, no resampling                                      */
/* 10BPK = X, not used in 8-bit capture                          */
/* EXC   = 0, use EAV/SAV codes                                  */
/* VRST  = 1, end of vertical blanking                           */
/* HRST  = 0, start of horizontal blanking                       */
/* FLDD  = 0, 1st line EAV or FID input                          */
/* FINV  = 0, no field invert                                    */
/* RDFE  = X, used in Raw mode only(Enable field identification)  */
/* SSE   = X, used in Raw mode only(Startup synch enable)        */
/*****************************************************************/
#include <csl_vp.h>
#include <csl_edma.h>
#include <csl_irq.h>

/*-------------------------------------------------------------- */
/* global variable declarations                                  */
/* ------------------------------------------------------------- */
VP_Handle vpCaptureHandle; /* handle of vp that to be configured   */

Uint8 capChaAYSpace[];  /* buffer to store captured Y-data         */
Uint8 capChaACbSpace[]; /* buffer to store captured Cb-data        */
Uint8 capChaACrSpace[]; /* buffer to store captured Cb-data        */

EDMA_Handle hEdmaVPCapChaAY;
EDMA_Handle hEdmaVPCapChaACb;
EDMA_Handle hEdmaVPCapChaACr;

Int32 edmaCapChaAYTccNum = 0;  /* EDMA tcc for Y channel           */
Int32 edmaCapChaACbTccNum = 0; /* EDMA tcc for Cb channel          */
Int32 edmaCapChaACrTccNum = 0; /* EDMA tcc for Cb channel          */

volatile Uint32 capChaAFrameCount = 0; /* no of frames captured    */
```

```
/* Error flags                                                          */
volatile Uint32 capChaAOverrun = 0;
volatile Uint32 capChaASyncError = 0;
volatile Uint32 capChaAShortFieldDetect = 0;
volatile Uint32 capChaALongFieldDetect = 0;

/* -------------------------------------------------------------------- */
/* Function    : bt656_8bit_ncfc                                        */
/* Input(s)    : portNumber, video port number i.e. 0, 1 or 2.          */
/* Description : Configures given video port for 8-bit BT.656 non-      */
/*               continuos frame capture on channel A.                  */
/* -------------------------------------------------------------------- */
void bt656_8bit_ncfc(int portNumber)
{
       /* Open video port for capture                                   */
       vpCaptureHandle = VP_open(portNumber, VP_OPEN_RESET);
       if(vpCaptureHandle == INV)
              test_exit(FAIL);

       /* Enable video port functionality in VP Peripheral              */
       /* Control Reg(PCR)                                              */
       VP_FSETH(vpCaptureHandle, PCR, PEREN, VP_PCR_PEREN_ENABLE);

       /* ---------------------- */
       /* Enable all interrupts */
       /* ---------------------- */

       /* Enable capture overrun interrupt(COVRA) for VP channel A      */
       VP_FSETH(vpCaptureHandle, VPIE, COVRA, VP_VPIE_COVRA_ENABLE);

       /* Enable capture complete interrupt(CCMPA) for VP channel A     */
       VP_FSETH(vpCaptureHandle, VPIE, CCMPA, VP_VPIE_CCMPA_ENABLE);

       /* Enable channel synchronization error interrupt(SERRA) for     */
       /* VP channel A                                                  */
       VP_FSETH(vpCaptureHandle, VPIE, SERRA, VP_VPIE_SERRA_ENABLE);

       /* Enable short field detect interrupt(SFDA) for VP channel A     */
       VP_FSETH(vpCaptureHandle, VPIE, SFDA, VP_VPIE_SFDA_ENABLE);

       /* Enable video port global interrupt enable                     */
       VP_FSETH(vpCaptureHandle, VPIE, VIE, VP_VPIE_VIE_ENABLE);

       /* ---------------------- */
       /* Setup all other fields */
       /* ---------------------- */

       /* Enable short field detect                                     */
       VP_FSETH(vpCaptureHandle, VCACTL, SFDE, VP_VCACTL_SFDE_ENABLE);

       /* Set last pixel to be captured in Field1 (VCA_STOP1 reg)       */
       VP_RSETH(vpCaptureHandle, VCASTOP1,
                VP_VCASTOP1_RMK(VCA_YSTOP1, VCA_XSTOP1));
```

```
/* Set last pixel to be captured in Field2 (VCA_STOP2 reg)         */
VP_RSETH(vpCaptureHandle, VCASTOP2,
        VP_VCASTOP2_RMK(VCA_YSTOP2, VCA_XSTOP2));

/* Set first pixel to be captured in Field1 (VCA_STRT1 reg)        */
VP_RSETH(vpCaptureHandle, VCASTRT1, VP_VCASTRT1_RMK(VCA_YSTART1,
        VP_VCASTRT1_SSE_ENABLE, VCA_XSTART1));

/* Set first pixel to be captured in Field2 (VCA_STRT2 reg)        */
VP_RSETH(vpCaptureHandle, VCASTRT2,
        VP_VCASTRT2_RMK(VCA_YSTART2, VCA_XSTART2));

/* Set threshold values                                            */
VP_RSETH(vpCaptureHandle, VCATHRLD,
        VP_VCATHRLD_RMK(VCA_THRLD_FIELD2, VCA_THRLD_FIELD1));

/* Set capture event-register values                               */
VP_RSETH(vpCaptureHandle, VCAEVTCT,
        VP_VCAEVTCT_RMK(VCA_CAPEVT2,VCA_CAPEVT1));

/* Vertical interrupts (VCA_INT) are not enabled in this           */
/* in this example.                                                */

/* Set CMODE to 8-bit BT.656                                       */
VP_FSETH(vpCaptureHandle, VCACTL, CMODE, VP_VCACTL_CMODE_BT656B);

/* Set non-continuous frame capture                                */
VP_FSETH(vpCaptureHandle, VCACTL, CON, VP_VCACTL_CON_DISABLE);
VP_FSETH(vpCaptureHandle, VCACTL, FRAME, VP_VCACTL_FRAME_FRMCAP);
VP_FSETH(vpCaptureHandle, VCACTL, CF2, VP_VCACTL_CF2_NONE);
VP_FSETH(vpCaptureHandle, VCACTL, CF1, VP_VCACTL_CF1_NONE);

/* Let FDD and FINV to be their defaults                           */

/* Set VRST to end of vertical blanking                            */
VP_FSETH(vpCaptureHandle, VCACTL, VRST, VP_VCACTL_VRST_V0EAV);

/* Set HRST to start of horizontal blanking                        */
VP_FSETH(vpCaptureHandle, VCACTL, HRST, VP_VCACTL_HRST_OF(0));

/* 10-bit pack mode(10BPK bit) in this 8-bit example               */

/* No (1/2) scaling and no chroma re-sampling in this example      */

/* Enable video port interrupts                                    */
IRQ_enable(vpCaptureHandle->eventId);

/* Setup Y, Cb and Cr EDMA channels                                */
setupVPCapChaAEDMA(portNumber);

/* Clear VPHLT in VP_CTL to make video port function               */
VP_FSETH(vpCaptureHandle, VPCTL, VPHLT, VP_VPCTL_VPHLT_CLEAR);
```

```
      /* -------------- */
      /* enable capture */
      /* -------------- */

      /* set VCEN bit to enable capture                                  */
      VP_FSETH(vpCaptureHandle, VCACTL, VCEN, VP_VCACTL_VCEN_ENABLE);

      /* clear BLKCAP in VCA_CTL to enable capture DMA events            */
      VP_FSETH(vpCaptureHandle, VCACTL, BLKCAP,
                VP_VCACTL_BLKCAP_CLEAR);
}

/*---------------------------------------------------------------- */
/* Function    : VPCapChaAIsr                                      */
/* Description : This capture ISR clears FRMC to continue capture  */
/*               in this non-continuous mode and also clears other */
/*               status bits.                                      */
/*---------------------------------------------------------------- */
interrupt void VPCapChaAIsr(void)
{
      Uint32 vpis = 0;
      /* Get video port status register value                        */
      vpis = VP_RGETH(vpCaptureHandle, VPIS);

      if(vpis & _VP_VPIS_CCMPA_MASK) /* capture complete               */
      {
            /* Clear frame complete bit in VCX_CTL to                */
            /* continue capture in non-continuous mode               */
            VP_FSETH(vpCaptureHandle, VCASTAT, FRMC,
                      VP_VCASTAT_FRMC_CLEAR);

            /* Clear CCMPA to enable next frame complete             */
            /* interrupts                                            */
            VP_FSETH(vpCaptureHandle, VPIS, CCMPA,VP_VPIS_CCMPA_CLEAR);

            capChaAFrameCount++;   /* increment captured frame count */
      }

      if(vpis & _VP_VPIS_COVRA_MASK) /* overrun error                */
      {
            capChaAOverrun++;
            VP_FSETH(vpCaptureHandle, VPIS, COVRA,VP_VPIS_COVRA_CLEAR);
      }

      if(vpis & _VP_VPIS_SERRA_MASK) /* synchronization error        */
      {
            capChaASyncError++;
            VP_FSETH(vpCaptureHandle, VPIS, SERRA,VP_VPIS_SERRA_CLEAR);
      }
```

```
        if(vpis & _VP_VPIS_SFDA_MASK) /* short field detect                */
        {
                capChaAShortFieldDetect++;
                VP_FSETH(vpCaptureHandle, VPIS, SFDA, VP_VPIS_SFDA_CLEAR);
        }

        if(vpis & _VP_VPIS_LFDA_MASK) /* long field detect                 */
        {
                capChaALongFieldDetect++;
                VP_FSETH(vpCaptureHandle, VPIS, LFDA, VP_VPIS_LFDA_CLEAR);
        }
}

/*-------------------------------------------------------------------- */
/* Function    : setupVPCapChaAEDMA                                    */
/* Input(s)    : portNumber, video port number i.e. 0, 1 or 2.         */
/* Description : Sets up EDMA channels for Y, U, V events for          */
/*               channel A capture.                                    */
/*-------------------------------------------------------------------- */
void setupVPCapChaAEDMA(Int32 portNumber)
{
        Int32 YEvent, UEvent, VEvent;

        /* get channelA Y, U, V EDMA event numbers                      */
        switch(portNumber)
        {
                case VP_DEV0: YEvent = EDMA_CHA_VP0EVTYA;
                              UEvent = EDMA_CHA_VP0EVTUA;
                              VEvent = EDMA_CHA_VP0EVTVA;
                              break;
                case VP_DEV1: YEvent = EDMA_CHA_VP1EVTYA;
                              UEvent = EDMA_CHA_VP1EVTUA;
                              VEvent = EDMA_CHA_VP1EVTVA;
                              break;
                case VP_DEV2: YEvent = EDMA_CHA_VP2EVTYA;
                              UEvent = EDMA_CHA_VP2EVTUA;
                              VEvent = EDMA_CHA_VP2EVTVA;
                              break;
        }

        /* Configure Y EDMA channel to move data from YSRCA             */
        /* (FIFO) to Y-data buffer, capChaAYSpace                       */
        configVPCapEDMAChannel(&hEdmaVPCapChaAY, YEvent,
                &edmaCapChaAYTccNum,
                vpCaptureHandle->ysrcaAddr,
                (Uint32)capChaAYSpace,
                VCA_Y_EDMA_FRMCNT,
                VCA_Y_EDMA_ELECNT);
```

```
        /* Configure Cb EDMA channel to move data from CbSRCA              */
        /* (FIFO) to Cb-data buffer, capChaACbSpace                        */
        configVPCapEDMAChannel(&hEdmaVPCapChaACb, UEvent,
               &edmaCapChaACbTccNum,
               vpCaptureHandle->cbsrcaAddr,
               (Uint32)capChaACbSpace,
               VCA_Y_EDMA_FRMCNT,
               VCA_Y_EDMA_ELECNT/2); /* (1/2) of Y-samples                */

        /* Configure Cr EDMA channel to move data from CrSRCA              */
        /* (FIFO) to Cr-data buffer, capChaACrSpace                        */
        configVPCapEDMAChannel(&hEdmaVPCapChaACr, VEvent,
               &edmaCapChaACrTccNum,
               vpCaptureHandle->crsrcaAddr,
               (Uint32)capChaACrSpace,
               VCA_Y_EDMA_FRMCNT,
               VCA_Y_EDMA_ELECNT/2); /* (1/2) of Y-samples                */

        /* Enable three EDMA channels                                      */
        EDMA_enableChannel(hEdmaVPCapChaAY);
        EDMA_enableChannel(hEdmaVPCapChaACb);
        EDMA_enableChannel(hEdmaVPCapChaACr);
}

/*-------------------------------------------------------------------  */
/* Function : configVPCapEDMAChannel                                   */
/*                                                                     */
/* Input(s) :  edmaHandle   - pointer to EDMA handle.                  */
/*             eventId      - EDMA eventId.                            */
/*             tccNum       - pointer to transfer complete number.     */
/*             srcAddr      - source address for EDMA transfer.        */
/*             dstAddr      - destination address for EDMA transfer    */
/*             frameCount   - frame count.                             */
/*             elementCount - element count(32-bit element size).      */
/*                                                                     */
/* Output(s):  edmaHandle   - edma Handle of the given event.          */
/*             tccNum       - transfer complete code for the given     */
/*                            event.                                   */
/*                                                                     */
/* Description : Configures the given VP capture EDMA channel.         */
/*               The source address update is fixed address mode       */
/*               because the captured data is read from the FIFO.      */
/*               In this example, the destination address mode is      */
/*               auto-increment. But, in real-time applications        */
/*               there is lot of flexibility in the way capture        */
/*               buffers can be managed like ping-pong and round       */
/*               robin,...etc.                                        */
/*-------------------------------------------------------------------  */
```

```
void configVPCapEDMAChannel(EDMA_Handle *edmaHandle, Int32 eventId,
                            Int32 *tccNum, Uint32 srcAddr,
                            Uint32 dstAddr, Uint32 frameCount,
                            Uint32 elementCount)
{
       Int32 tcc = 0;

       /* Open Y EVT EDMA channel                                          */
       *edmaHandle = EDMA_open(eventId, EDMA_OPEN_RESET);

       if(*edmaHandle == EDMA_HINV)
              test_exit(FAIL);

       /* allocate TCC for Y event                                         */
       if((tcc = EDMA_intAlloc(-1)) == -1)
              test_exit(FAIL);

       /* Configure EDMA parameters                                        */
       EDMA_configArgs(
         *edmaHandle,
         EDMA_OPT_RMK(
            EDMA_OPT_PRI_MEDIUM,   /* medium priority               */
            EDMA_OPT_ESIZE_32BIT,  /* Element size 32 bits          */
            EDMA_OPT_2DS_NO,       /* 1-dimensional source(FIFO)    */
            EDMA_OPT_SUM_NONE,     /* fixed src address mode(FIFO)  */
            EDMA_OPT_2DD_YES,      /* 2-dimensional destination     */
            EDMA_OPT_DUM_INC,      /* destination increment         */
            EDMA_OPT_TCINT_YES,    /* Enable transfer complete      */
                                   /* indication                    */
            EDMA_OPT_TCC_OF(tcc & 0xF),
            EDMA_OPT_TCCM_OF(((tcc & 0x30) >> 4)),
            EDMA_OPT_ATCINT_NO,    /* Disable Alternate Transfer    */
                                   /* Complete Interrupt            */
            EDMA_OPT_ATCC_OF(0),
            EDMA_OPT_PDTS_DISABLE, /* disable PDT(peripheral device */
                                   /* transfer) mode for source     */
            EDMA_OPT_PDTD_DISABLE, /* disable PDT mode for dest     */
            EDMA_OPT_LINK_NO,      /* Disable linking               */
            EDMA_OPT_FS_NO         /* Array synchronization         */
            ),
         EDMA_SRC_RMK(srcAddr),
         EDMA_CNT_RMK(EDMA_CNT_FRMCNT_OF((frameCount - 1)),
                      EDMA_CNT_ELECNT_OF(elementCount)),
         EDMA_DST_RMK(dstAddr),
         EDMA_IDX_RMK(EDMA_IDX_FRMIDX_OF((elementCount * 4)),
         EDMA_IDX_ELEIDX_OF(0)),   /* note: 32-bit element size     */
         /* no RLD in 2D and no linking                               */
         EDMA_RLD_RMK(EDMA_RLD_ELERLD_OF(0), EDMA_RLD_LINK_OF(0))
       );

        *tccNum = tcc;
}
```

B.2 Example 2: Noncontinuous Frame Display for 525/60 Format

This is an example that explains how to configure the video port for 8-bit BT.656 noncontinuous frame display for 525/60 format. See ITU−R BT.656−4 and video port specification (Figures 4−11, 4−33, 4−34, and Table 4−37) for more details on 525/60 format. For simplicity, this example does not contain any margins; that is, both vertical and horizontal offsets are zero. In other words, both active area and image area are the same.

```
/*********************************************************/
/* Display parameter definitions based on 525/60 format   */
/*********************************************************/

/* ----------------- */
/* Define frame size */
/* ----------------- */
#define VD_FRM_WIDTH       858 /* no of pixels per frame line     */
                               /* including horizontal blanking   */
#define VD_FRM_HEIGHT      525 /* total noof lines per frame      */
#define VD_FRM_SIZE        (VD_FRM_WIDTH * VD_FRM_HEIGHT)

/* ------------------- */
/* Horizontal blanking */
/* ------------------- */
#define VD_HBLNK_START     720 /* starting location of EAV        */
#define VD_HBLNK_STOP      856 /* starting location of SAV        */
#define VD_HBLNK_SIZE      (VD_HBLNK_STOP - VD_HBLNK_START +
                            2/*EAV*/) /* (138) EAV, SAV inclusive */

/* --------------------------- */
/* Vertical blanking for field1 */
/* --------------------------- */
#define VD_VBLNK_XSTART1   720 /* pixel on which VBLNK active     */
                               /* edge occurs for field1          */
#define VD_VBLNK_YSTART1   1   /* line on which VBLNK active      */
                               /* edge occurs for field1          */
#define VD_VBLNK_XSTOP1    720 /* pixel on which VBLNK inactive   */
                               /* edge occurs for field1          */
#define VD_VBLNK_YSTOP1    20  /* line on which VBLNK inactive    */
                               /* edge occurs for field1          */

/* --------------------------- */
/* Vertical blanking for field2 */
/* --------------------------- */
#define VD_VBLNK_XSTART1   360 /* pixel on which VBLNK active     */
                               /* edge occurs for field2          */
#define VD_VBLNK_YSTART1   263 /* line on which VBLNK active      */
                               /* edge occurs for field2          */
#define VD_VBLNK_XSTOP1    360 /* pixel on which VBLNK inactive   */
                               /* edge occurs for field2          */
#define VD_VBLNK_YSTOP1    283 /* line on which VBLNK inactive    */
                               /* edge occurs for field2          */
```

```
/* ------------------------------------------------ */
/* Define vertical blanking bit(VD_VBITn) reg values */
/* ------------------------------------------------ */
#define VD_VBIT_SET1       1   /* first line with an EAV with V=1          */
                               /* indicating the start of Field1           */
                               /* vertical blanking                        */
#define VD_VBIT_CLR1       20  /* first line with an EAV with V=0          */
                               /* indicating the start of Field1           */
                               /* active display                           */
#define VD_VBLNK1_SIZE     (VD_VBIT_CLR1 - VD_VBIT_SET1) /* 19 lines       */

#define VD_VBIT_SET2       264 /* first line with an EAV with V=1          */
                               /* indicating the start of Field2           */
                               /* vertical blanking                        */
#define VD_VBIT_CLR2       283 /* first line with an EAV with V=0          */
                               /* indicating the start of Field2           */
                               /* active display                           */
#define VD_VBLNK2_SIZE     (VD_VBIT_CLR2 - VD_VBIT_SET2) /* 19 lines       */

/* ------------ */
/* Field timing */
/* ------------ */
#define VD_FIELD1_XSTART   720 /* pixel on the first line of               */
                               /* Field1 on which FLD ouput                */
                               /* is de-asserted                           */
#define VD_FIELD1_YSTART   1   /* line on which FLD is de-asserted         */
#define VD_FIELD1_XSTART   360 /* pixel on the first line of               */
                               /* Field1 on which FLD ouput                */
                               /* is asserted                              */
#define VD_FIELD1_YSTART   263 /* line on which FLD is asserted            */

/* ---------------------------------- */
/* Define field bit(VD_FBIT) reg values */
/* ---------------------------------- */
#define VD_FBIT_CLR        4   /* first line with an EAV with F=0          */
                               /* indicating Field 1 display               */
#define VD_FBIT_SET        266 /* first line with an EAV with F=1          */
                               /* indicating Field 2 display               */

/* -------------------------------- */
/* Define horzontal synchronization */
/* -------------------------------- */
#define VD_HSYNC_START     736
#define VD_HSYNC_STOP      800

/* ---------------------------------------- */
/* Define vertical synchronization for field1 */
/* ---------------------------------------- */
#define VD_VSYNC_XSTART1   720
#define VD_VSYNC_YSTART1   4
#define VD_VSYNC_XSTOP1    720
#define VD_VSYNC_YSTOP1    7
```

```
/* ---------------------------------------- */
/* Define vertical synchronization for field2 */
/* ---------------------------------------- */
#define VD_VSYNC_XSTART2   360
#define VD_VSYNC_YSTART2   266
#define VD_VSYNC_XSTOP2    360
#define VD_VSYNC_YSTOP2    269

/* ---------------------------------------- */
/* Define image offsets for both the fields */
/* which are zero in this example           */
/* ---------------------------------------- */
#define VD_IMG_HOFF1       0
#define VD_IMG_VOFF1       0
#define VD_IMG_HOFF2       0
#define VD_IMG_VOFF2       0

/* ------------------------------------------------ */
/* Define image active vertical and horizontal sizes */
/* ------------------------------------------------ */
#define VD_IMG_HSIZE1      720 /* field1 horizontal image size     */
#define VD_IMG_VSIZE1      244 /* field1 vertical image size       */
#define VD_IMG_HSIZE2      720 /* field2 horizontal image size     */
#define VD_IMG_VSIZE2      243 /* field2 vertical image size       */

/* Manipulate field1 and field2 image sizes                        */
#define VD_IMAGE_SIZE1     (VD_IMG_HSIZE1 * VD_IMG_VSIZE1)
#define VD_IMAGE_SIZE2     (VD_IMG_HSIZE2 * VD_IMG_VSIZE2)

/* Define threshold values in double-words. Both fields should     */
/* have same threshold value                                       */
#define VD_VDTHRLD1        (VD_IMG_HSIZE1/8) /* line length in     */
#define VD_VDTHRLD2        VD_VDTHRLD1       /* double-words       */

/* Define number of events to be generated for field1 and field2   */
#define VD_DISPEVT1        (VD_IMAGE_SIZE1 / (VD_VDTHRLD1 * 8))
#define VD_DISPEVT2        (VD_IMAGE_SIZE2 / (VD_VDTHRLD2 * 8))

#define DISPLAY_FRAME_COUNT    5 /* in this example               */

/* ---------------------------------------- */
/* EDMA parameters for display Y event that are */
/* specific to this example.                */
/* ---------------------------------------- */
#define VD_Y_EDMA_ELECNT   (VD_VDTHRLD1 * 2)  /* VD_VDTHRLDn is in double-words
                                                 and 32-bit element size */
#define VD_Y_EDMA_FRMCNT   ((VD_DISPEVT1 + VD_DISPEVT2) *
                            DISPLAY_FRAME_COUNT)
```

```
/*****************************************************************/
/* Description :  8-bit BT.656 non-continuous frame display      */
/*                                                               */
/* Some important field descriptions:                            */
/*                                                               */
/* DMODE  = 000, 8-bit BT.656 mode                               */
/* CON    = 0                                                    */
/* FRAME  = 1, display frame                                     */
/* DF2    = 0                                                    */
/* DF1    = 0, (8-bit non-continuous frame display)              */
/* SCALE  = 0, no scaling                                        */
/* RESMPL = 0, no resampling                                     */
/* DPK    = X, not used in 8-bit display                         */
/* RSYNC  = X, used in Raw mode(Enable second synchronized raw   */
/*          data channel)                                        */
/* RGBX   = X, used in Raw mode(RGB extract enable. Perform      */
/*          3/4 FIFO unpacking)                                  */
/* VCTL1S = 00, output HSYNC                                     */
/* VCTL2S = 00, output VSYNC                                     */
/* VCTL3S = 0, output CBLNK                                      */
/* HXS    = 0, VCTL1 is an output                                */
/* VXS    = 0, VCTL2 is an output                                */
/* FXS    = 0, VCTL3 is an output                                */
/* PVPSYN = 0, no previous port synchronization                  */
/*****************************************************************/
#include "csl_vp.h"
#include "csl_edma.h"
#include "csl_irq.h"

/*---------------------------------------------------------------- */
/* global variable declarations                                    */
/* ---------------------------------------------------------------- */
VP_Handle vpDisplayHandle;  /* handle of vp that to be configured  */

Uint8 dispYSpace[];  /* Display Y-data buffer  */
Uint8 dispCbSpace[]; /* Display Cb-data buffer */
Uint8 dispCrSpace[]; /* Display Cb-data buffer */

EDMA_Handle hEdmaVPDispY;
EDMA_Handle hEdmaVPDispCb;
EDMA_Handle hEdmaVPDispCr;

Int32 edmaDispYTccNum = 0;  /* EDMA tcc for Y channel  */
Int32 edmaDispCbTccNum = 0; /* EDMA tcc for Cb channel */
Int32 edmaDispCrTccNum = 0; /* EDMA tcc for Cb channel */

volatile Uint32 displayFrameCount = 0;  /* no of frames that are   */
                                        /* displayed               */
volatile Uint32 dispUnderrun = 0;       /* underrun error flag     */
```

```
/*-------------------------------------------------------------------- */
/* Function    : bt656_8bit_ncfd                                       */
/* Input(s)    : portNumber, video port number i.e. 0, 1 or 2.         */
/* Description : Configures given video port for 8-bit BT.656 non-     */
/*               continuous frame display.                             */
/*-------------------------------------------------------------------- */
void bt656_8bit_ncfd(int portNumber)
{
      /* Open video port for display                                    */
      vpDisplayHandle  = VP_open(portNumber, VP_OPEN_RESET);
      if(vpDisplayHandle  == INV)
            test_exit(FAIL);

      /* Enable video port functionality in VP Peripheral               */
      /* Control Reg(PCR)                                               */
      VP_FSETH(vpDisplayHandle , PCR, PEREN, VP_PCR_PEREN_ENABLE);

      /* Set this port to display mode                                  */
      VP_FSETH(vpDisplayHandle , VPCTL, DISP, VP_VPCTL_DISP_DISPLAY);

      /* ---------------------- */
      /* Enable all interrupts */
      /* ---------------------- */

      /* enable display complete interrupt                              */
      VP_FSETH(vpDisplayHandle , VPIE, DCMP, VP_VPIE_DCMP_ENABLE);

      /* enable display underrun interrupt                              */
      VP_FSETH(vpDisplayHandle , VPIE, DUND, VP_VPIE_DUND_ENABLE);

      /* enable video port global interrupt enable                      */
      VP_FSETH(vpDisplayHandle , VPIE, VIE, VP_VPIE_VIE_ENABLE);

      /* -------------------- */
      /* Set all other fields */
      /* -------------------- */

      /* set frame size                                                 */
      VP_RSETH(vpDisplayHandle , VDFRMSZ,
                  VP_VDFRMSZ_RMK(VD_FRM_HEIGHT, VD_FRM_WIDTH));

      /* set horizontal blanking                                        */
      VP_RSETH(vpDisplayHandle , VDHBLNK,
              VP_VDHBLNK_RMK(VD_HBLNK_STOP, VP_VDHBLNK_HBDLA_NONE,
              VD_HBLNK_START));

      /* set vertical blanking start for field1                         */
      VP_RSETH(vpDisplayHandle , VDVBLKS1,
                  VP_VDVBLKS1_RMK(VD_VBLNK_YSTART1,VD_VBLNK_XSTART1));

      /* set vertical blanking end for field1                           */
      VP_RSETH(vpDisplayHandle , VDVBLKE1,
                  VP_VDVBLKE1_RMK(VD_VBLNK_YSTOP1, VD_VBLNK_XSTOP1));
```

```
/* set vertical blanking start for field2                          */
VP_RSETH(vpDisplayHandle , VDVBLKS2,
           VP_VDVBLKS2_RMK(VD_VBLNK_YSTART2, VD_VBLNK_XSTART2));

/* set vertical blanking end for field2                            */
VP_RSETH(vpDisplayHandle , VDVBLKE2,
           VP_VDVBLKE2_RMK(VD_VBLNK_YSTOP2, VD_VBLNK_XSTOP2));

/* set vertical blanking bit register for field 1(VD_VBIT1)        */
VP_RSETH(vpDisplayHandle , VDVBIT1,
            VP_VDVBIT1_RMK(VD_VBIT_CLR1, VD_VBIT_SET1));

/* set vertical blanking bit register for field 2(VD_VBIT2)        */
VP_RSETH(vpDisplayHandle , VDVBIT2,
            VP_VDVBIT2_RMK(VD_VBIT_CLR2, VD_VBIT_SET2));

/* No image offsets in this example                                */

/* set image size for field1                                       */
VP_RSETH(vpDisplayHandle , VDIMGSZ1,
            VP_VDIMGSZ1_RMK(VD_IMG_VSIZE1, VD_IMG_HSIZE1));

/* set image size for field2                                       */
VP_RSETH(vpDisplayHandle , VDIMGSZ2,
            VP_VDIMGSZ1_RMK(VD_IMG_VSIZE2, VD_IMG_HSIZE2));

/* set field1 timing                                               */
VP_RSETH(vpDisplayHandle , VDFLDT1,
            VP_VDFLDT1_RMK(VD_FIELD1_YSTART, VD_FIELD1_XSTART));

/* set field2 timing                                               */
VP_RSETH(vpDisplayHandle , VDFLDT2,
            VP_VDFLDT2_RMK(VD_FIELD2_YSTART, VD_FIELD2_XSTART));

/* set display field bit register(VD_FBIT)                         */
VP_RSETH(vpDisplayHandle , VDFBIT,
            VP_VDFBIT_RMK(VD_FBIT_SET, VD_FBIT_CLR));

/* set horizontal sync control (VCTL1S)                            */
VP_RSETH(vpDisplayHandle , VDHSYNC,
            VP_VDHSYNC_RMK(VD_HSYNC_STOP, VD_HSYNC_START));

/* set vertical sync start for field1 (VCTL2S)                     */
VP_RSETH(vpDisplayHandle , VDVSYNS1,
            VP_VDVSYNS1_RMK(VD_VSYNC_YSTART1,VD_VSYNC_XSTART1));

/* set vertical sync end for field1 (VCTL2S)                       */
VP_RSETH(vpDisplayHandle , VDVSYNE1,
            VP_VDVSYNE1_RMK(VD_VSYNC_YSTOP1, VD_VSYNC_XSTOP1));

/* set vertical sync start for field2 (VCTL2S)                     */
VP_RSETH(vpDisplayHandle , VDVSYNS2,
            VP_VDVSYNS2_RMK(VD_VSYNC_YSTART2,VD_VSYNC_XSTART2));
```

```
/* set vertical sync end for field2 (VCTL2S)                          */
VP_RSETH(vpDisplayHandle , VDVSYNE2,
             VP_VDVSYNE2_RMK(VD_VSYNC_YSTOP2, VD_VSYNC_XSTOP2));

/* Let clipping values to be their defaults (VD_CLIP)                 */

/* No need to set DEF_VAL and VD_RELOAD in this example               */

/* set event register                                                 */
VP_RSETH(vpDisplayHandle , VDDISPEVT,
             VP_VDDISPEVT_RMK(VD_DISPEVT2, VD_DISPEVT1));

/* Vertical interrupts are not used in this example (VD_VINT)         */

/* set threshold value for DMA events                                 */
VP_RSETH(vpDisplayHandle, VDTHRLD,
         VP_VDTHRLD_RMK(VD_VDTHRLD2,
         VP_VDTHRLD_INCPIX_DEFAULT, VD_VDTHRLD1));

/* ------------------------------- */
/* Set display control reg(VD_CTL) */
/* ------------------------------- */

/* set display mode(DMODE) to 8-bit BT.656                            */
VP_FSETH(vpDisplayHandle , VDCTL, DMODE, VP_VDCTL_DMODE_BT656B);

/* set non-continuous frame display                                   */
VP_FSETH(vpDisplayHandle , VDCTL, CON, VP_VDCTL_CON_DISABLE);
VP_FSETH(vpDisplayHandle , VDCTL, FRAME, VP_VDCTL_FRAME_FRMDIS);
VP_FSETH(vpDisplayHandle , VDCTL, DF2, VP_VDCTL_DF2_NONE);
VP_FSETH(vpDisplayHandle , VDCTL, DF1, VP_VDCTL_DF1_NONE);

/* let control outputs(VCTL1S, VCTL2S, VCTL3S, HXS, VXS, FXS)         */
/* be their defaults i.e. VCTLxS are output control signals           */

/* no scaling and no resampling in this example                       */

/* no need to bother about 10-bit unpacking mode(DPK bit)             */
/* in this 8-bit example                                              */

/* Set up Y, Cb and Cr EDMA channels                                  */
setupVPDispEDMA(portNumber);

/* Enable video port interrupts                                       */
IRQ_enable(vpDisplayHandle ->eventId);

/* clear VPHLT in VP_CTL to make video port function                  */
VP_FSETH(vpDisplayHandle , VPCTL, VPHLT, VP_VPCTL_VPHLT_CLEAR);
```

```
      /* -------------- */
      /* enable display */
      /* -------------- */

      /* set VDEN to enable display for loop-back                        */
      VP_FSETH(vpBDisplayHandle, VDCTL, VDEN, VP_VDCTL_VDEN_ENABLE);

      /* clear BLKDIS in VD_CTL to enable display DMA events             */
      VP_FSETH(vpBDisplayHandle, VDCTL, BLKDIS, VP_VDCTL_BLKDIS_CLEAR);
}

/*-------------------------------------------------------------------- */
/* Function    : VPDispIsr                                              */
/* Description : This display ISR clears FRMD to continue display  */
/*               in this non-continuous mode and also clears other */
/*               status bits.                                          */
/*-------------------------------------------------------------------- */
interrupt void VPDispIsr(void)
{
      Uint32 vpis = 0;

      vpis = VP_RGETH(vpDisplayHandle , VPIS);

      if(vpis & _VP_VPIS_DCMP_MASK) /* frame display complete           */
      {

            /* Clear frame complete bit FRMD to continue display        */
            VP_FSETH(vpDisplayHandle , VDSTAT, FRMD,
                        VP_VDSTAT_FRMD_CLEAR);

            /* clear DCMPA to enable next frame complete interrupts     */
            VP_FSETH(vpDisplayHandle , VPIS, DCMP, VP_VPIS_DCMP_CLEAR);

            displayFrameCount++; /* increment displayed frame count     */
      }

      if(vpis & _VP_VPIS_DUND_MASK) /* underrun error                  */
      {
            dispUnderrun++;

            /* clear DUND to enable next underrun interrupts            */
            VP_FSETH(vpDisplayHandle , VPIS, DUND, VP_VPIS_DUND_CLEAR);
      }
}
```

```
/*---------------------------------------------------------------------- */
/* Function    : setupVPDispEDMA                                        */
/* Input(s)    : portNumber, video port number i.e. 0,1 or 2.           */
/* Description : Sets up DMA channels for Y, U, V events for VP         */
/*               display.                                               */
/*---------------------------------------------------------------------- */
void setupVPDispEDMA(Int32 portNumber)
{
       Int32 YEvent, UEvent, VEvent;

       /* get Y, U, V EDMA event numbers                                */
       switch(portNumber)
       {
              case VP_DEV0: YEvent = EDMA_CHA_VP0EVTYA;
                            UEvent = EDMA_CHA_VP0EVTUA;
                            VEvent = EDMA_CHA_VP0EVTVA;
                           break;
              case VP_DEV1: YEvent = EDMA_CHA_VP1EVTYA;
                            UEvent = EDMA_CHA_VP1EVTUA;
                            VEvent = EDMA_CHA_VP1EVTVA;
                           break;
              case VP_DEV2: YEvent = EDMA_CHA_VP2EVTYA;
                            UEvent = EDMA_CHA_VP2EVTUA;
                            VEvent = EDMA_CHA_VP2EVTVA;
                           break;
       }

       /* Configure Y EDMA channel to move data from                    */
       /* Y-data buffer, dispYSpace to YDSTA (FIFO)                     */
       configVPDispEDMAChannel(&hEdmaVPDispY, YEvent,
                               &edmaDispYTccNum,
                               (Uint32)dispYSpace,
                               vpDisplayHandle ->ydstaAddr,
                               VD_Y_EDMA_FRMCNT,
                               VD_Y_EDMA_ELECNT);

       /* Configure Cb EDMA channel to move data from                   */
       /* Cb-data buffer, dispCbSpace to CbDSTA (FIFO)                  */
       configVPDispEDMAChannel(&hEdmaVPDispCb, UEvent,
                               &edmaDispCbTccNum,
                               (Uint32)dispCbSpace,
                               vpDisplayHandle ->cbdstAddr,
                               VD_Y_EDMA_FRMCNT,
                               VD_Y_EDMA_ELECNT/2); /* (1/2) of Y      */

       /* Configure Cr EDMA channel to move data from                   */
       /* Cr-data buffer, dispCrSpace to CrDSTA (FIFO)                  */
       configVPDispEDMAChannel(&hEdmaVPDispCr, VEvent,
                               &edmaDispCrTccNum,
                               (Uint32)dispCrSpace,
                               vpDisplayHandle ->crdstAddr,
                               VD_Y_EDMA_FRMCNT,
                               VD_Y_EDMA_ELECNT/2);
```

```
      /* enable three EDMA channels                                        */
      EDMA_enableChannel(hEdmaVPDispY);
      EDMA_enableChannel(hEdmaVPDispCb);
      EDMA_enableChannel(hEdmaVPDispCr);
}

/*-------------------------------------------------------------------    */
/* Function : configVPDispEDMAChannel                                    */
/*                                                                        */
/* Input(s) :   edmaHandle    - pointer to EDMA handle.                   */
/*              eventId       - EDMA eventId.                             */
/*              tccNum        - pointer to transfer complete number.      */
/*              srcAddr       - source address for EDMA transfer.         */
/*              dstAddr       - destination address for EDMA transfer     */
/*              frameCount    - frame count.                              */
/*              elementCount  - element count(32-bit element size).       */
/*                                                                        */
/* Output(s):   edmaHandle    - edma Handle of the given event.           */
/*              tccNum        - transfer complete code for the given      */
/*                              event.                                    */
/*                                                                        */
/* Description : Configures the given VP display EDMA channel.            */
/*               The destination address update is fixed because          */
/*               the displayed data is write to the FIFO.                 */
/*               In this example, the source address mode is              */
/*               auto-increment. But, in real-time applications           */
/*               there is lot of flexibility in the way display           */
/*               buffers can be managed like ping-pong and round          */
/*               robin,...etc.                                           */
/*-------------------------------------------------------------------    */
void configVPDispEDMAChannel(EDMA_Handle *edmaHandle,
                             Int32 eventId, Int32 *tccNum,
                             Uint32 srcAddr, Uint32 dstAddr,
                             Uint32 frameCount, Uint32 elementCount)
{
      Int32 tcc = 0;

      /* Open Y event EDMA channel                                        */
      *edmaHandle = EDMA_open(eventId, EDMA_OPEN_RESET);

      if(*edmaHandle == EDMA_HINV)
            test_exit(FAIL);

      /* allocate TCC for Y event                                         */
      if((tcc = EDMA_intAlloc(-1)) == -1)
            test_exit(FAIL);
```

```
      /* Configure EDMA parameters                                          */
      EDMA_configArgs(
        *edmaHandle,
        EDMA_OPT_RMK(
           EDMA_OPT_PRI_MEDIUM,   /* medium priority                 */
           EDMA_OPT_ESIZE_32BIT,  /* Element size 32 bits            */
           EDMA_OPT_2DS_YES,      /* 2-dimensional source            */
           EDMA_OPT_SUM_INC,      /* source address auto increment   */
           EDMA_OPT_2DD_NO,       /* 1-dimensional destination(FIFO) */
           EDMA_OPT_DUM_NONE,     /* fixed dest address mode(FIFO)   */
           EDMA_OPT_TCINT_YES,    /* Enable transfer complete        */
                                  /* indication                      */
           EDMA_OPT_TCC_OF(tcc & 0xF),
           EDMA_OPT_TCCM_OF(((tcc & 0x30) >> 4)),
           EDMA_OPT_ATCINT_NO,    /* Disable Alternate Transfer      */
                                  /* Complete Interrupt              */
           EDMA_OPT_ATCC_OF(0),
           EDMA_OPT_PDTS_DISABLE, /* disable PDT(peripheral device   */
                                  /* transfer) mode for source       */
           EDMA_OPT_PDTD_DISABLE, /* disable PDT mode for dest       */
           EDMA_OPT_LINK_NO,      /* Disable linking                 */
           EDMA_OPT_FS_NO         /* Array synchronization           */
           ),
        EDMA_SRC_RMK(srcAddr),
        EDMA_CNT_RMK(EDMA_CNT_FRMCNT_OF((frameCount - 1)),
                     EDMA_CNT_ELECNT_OF(elementCount)),
        EDMA_DST_RMK(dstAddr),
        EDMA_IDX_RMK(EDMA_IDX_FRMIDX_OF((elementCount * 4)),
        EDMA_IDX_ELEIDX_OF(0)),   /* note: 32-bit element size       */
        /* no RLD in 2D and no linking                                 */
        EDMA_RLD_RMK(EDMA_RLD_ELERLD_OF(0), EDMA_RLD_LINK_OF(0))
      );

        *tccNum = tcc;
}
```

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

V

Y