ANALOG DEVICES

JPEG2000 VIDEO CODEC

Preliminary Technical Data — ADV202*

March 19, 2004

Previously referred to as ADV-JP2001

1. FEATURES

- Complete single-chip JPEG2000 compression/ decompression solution for video and still images.
- Patented SURF™ (Spatial Ultra-efficient Recursive Filtering) technology enables low power and low cost wavelet based compression.
- Supports both 9/7 and 5/3 wavelet transforms with up to 6 levels of transform.
- Programmable tile/image size with widths up to 2048 pixels in three-component 4:2:2 interleaved mode, and up to 4096 pixels in single-component mode. Maximum tile/image height is 4096 pixels.
- Video interface directly supporting ITU.R-BT656, SMPTE125M PAL/ NTSC, SMPTE274M, SMPTE293M [525p], ITU.R-BT1358[625p] or any video format with a max. input rate of 65 Msamples/ sec for irreversible mode or 40Msamples/sec for reversible mode. Two or more ADV202s can be combined to support full frame SMPTE274M HDTV [1080i] or SMPTE296M [720p].
- Interlace temporally coherent frame based SD video sources for improved performance.
- Flexible asynchronous SRAM style host interface allows glue-less connection to most 16/32-bit micro-controllers and ASICs.
- 2.5-3.3v I/O and 1.5v core supply.
- 12mm x 12mm 121-ball fpBGA, speed grade 115MHZ or 13mmx13mm 144 fpBGA, speed grade 150MHz.

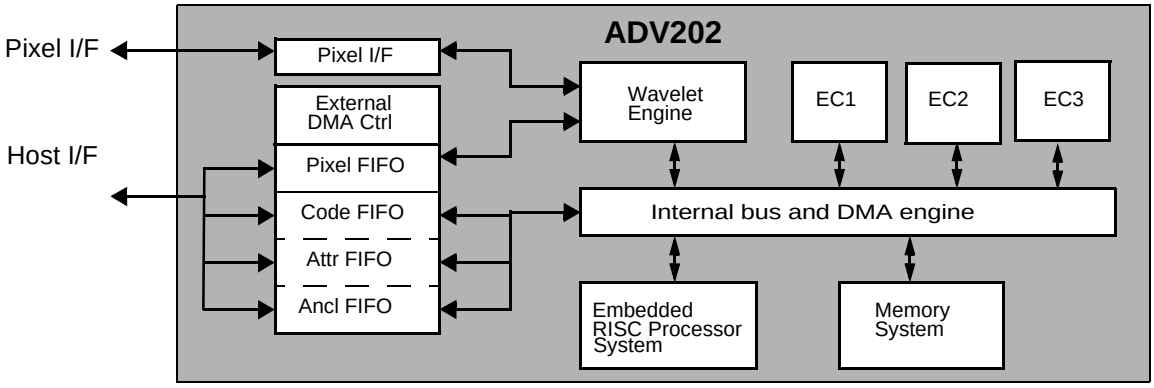
2. APPLICATIONS

- Networked video and image distribution systems
- Wireless video and image distribution
- Image archival/retrieval
- Digital CCTV and surveillance systems
- Digital Cinema Systems
- Professional Video editing and recording
- Digital Still Cameras
- Digital Camcorders

3. GENERAL DESCRIPTION

The ADV202 is a single-chip JPEG2000 CODEC targeted at video and high bandwidth image compression applications that will benefit by the enhanced quality and feature set provided by the JPEG2000 (J2K) - ISO/IEC15444-1 image compression standard. It implements the computationally intensive operations of the JPEG2000 image compression standard as well as providing fully compliant code stream generation for most applications. The ADV202's dedicated video port provides glueless connection to common digital video standards such as ITU.R-BT656, SMPTE125M, SMPTE293M [525p], ITU.R-BT1358 [625p], SMPTE274M[1080i] or SMPTE296M [720p].
A variety of other high speed synchronous pixel and video formats can also be supported using the programmable framing and validation signals.

The ADV202 can process images at a rate of 40M samples/sec in reversible mode, and at higher rates when used in irreversible mode. The ADV202



Rev. PrT

SURF is a trademark of Analog Devices, Inc.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A
Tel: 781.329.4700 **www.analog.com**
Fax: 781.326.8703

contains a dedicated wavelet transform engine, three entropy codecs, on board memory system and an embedded RISC processor which can provide a complete JPEG2000 compression/decompression solution.

The wavelet processor supports the 9/7 irreversible wavelet transform and the 5/3 wavelet transform in reversible and irreversible modes. The entropy codecs support all features in the JPEG2000 Part 1 specification, except Maxshift ROI.

The ADV202 operates on a rectangular array of pixel samples called a tile. A tile may contain a complete image, up to the maximum supported size, or some portion of an image. The maximum horizontal tile size supported depends on the wavelet transform selected and the number of samples in the tile. Images larger than the ADV202's maximum tile size may be broken into individual tiles and then sent sequentially to the chip while still maintaining a single, fully compliant, JPEG2000 code stream for the entire image.

3.1 JPEG2000 Feature Support

The ADV202 supports a broad set of features that are included in Part 1 of the JPEG2000 standard (ISO/IEC 15444). Refer to the "Getting Started with the ADV202" for information as to what JPEG2000 features are presently supported by the ADV202.

Depending on the particular application requirements, the ADV202 can provide varying levels of JPEG2000 compression support.
It can provide raw code-block and attribute data output which allows the host software to have complete control over the generation of the JPEG2000 code stream and other aspects of the compression process such as bit-rate control. Otherwise the ADV202 can create a complete, fully compliant, JPEG2000 code stream (.j2c) and enhanced file formats such as.jp2,.jpx and.mj2 (Motion JPEG2000). Refer to the "Getting Started with the ADV202" for information what formats are presently supported by the ADV202.

4. FUNCTIONAL DESCRIPTION

The input video or pixel data is passed on to the ADV202's pixel interface where samples are de-interleaved and passed on to the Wavelet Engine where each tile or frame is decomposed into subbands using the 5/3 or 9/7 filters. The resultant wavelet coefficients are then written to internal memory. The entropy codecs then code the image data so that it conforms to the JPEG2000 standard. An internal DMA provides high bandwidth memory to memory transfers as well as high performance transfers between functional blocks and memory.

4.1 Wavelet Engine

The ADV202 provides a dedicated wavelet transform processor based on Analog Devices' proven and patented SURF^R technology. This processor can perform up to 6 wavelet decomposition-levels on a tile. In Encode mode, the wavelet transform processor will take in uncompressed samples, perform the wavelet transform and write the wavelet coefficients in all frequency subbands to internal memory. Each of these subbands is then further broken down into code-blocks. The code-block dimensions can be user-defined, and are used by the wavelet transform processor to organize the wavelet coefficients into code-blocks when writing to internal memory. Each completed code-block is then entropy coded by one of the Entropy Codecs.
In Decode mode, wavelet coefficients are read from internal memory and are recomposed into uncompressed samples.

4.2 Entropy Codecs

The entropy codec block performs context modeling and arithmetic coding on a codeblock of the wavelet coefficients. Additionally, this block also performs the distortion metric calculations during compression that are required for optimal rate/distortion performance. Since the entropy coding process is the most computationally intensive operation in the JPEG2000 compression process, three dedicated hardware entropy codecs are provided on the ADV202.

4.3 Embedded Processor System

The ADV202 incorporates an embedded 32-bit RISC processor. This processor is used for configuration, control and management of the dedicated hardware functions as well for parsing/generation

of the JPEG2000 code stream. The processor system includes ROM and RAM for both program and data memory, an interrupt controller, standard bus interfaces and other hardware functions, such as timers, counters, etc.

4.4 Memory System

The memory system's main function is to manage wavelet coefficient data, interim code-block/ attribute data and temporary work space for creation/parsing/storage of JPEG2000 code-stream. The memory system can also be used for program and data memory for the embedded processor.

4.5 Internal DMA engine

The internal DMA engine provides high bandwidth memory to memory transfers as well as high performance transfers between memory and functional blocks. This function is critical for high speed code stream generation and parsing.

4.6 Configurable FIFO block

FIFOs are provided for pixel data, code-stream data, attribute data or ancillary data. The FIFOs can be accessed directly from the host interface using normal addressed read/write cycles or by external host DMA accesses using a DREQ/DACK protocol or by one of the dedicated hardware handshake protocols.
Each FIFO also has a programmable threshold that can be used to generate an interrupt.

Refer to Appendix for more detail on the dataflow and functional description.

5. ADV202 INTERFACE

There are several possible modes to interface to the ADV202 using the VDATA bus and the HDATA bus or the HDATA bus alone.

5.1 Video Interface [VDATA bus]

The video interface can be used in applications where uncompressed pixel data is on a separate bus from compressed data. For example, it is possible to use the VDATA bus to input uncompressed video while using the HDATA bus to output the compressed data. This interface is ideal for applications requiring very high throughput such as live video capture.
Optionally the ADV202 interlaces ITU.R-BT656 resolution video, on the fly prior to wavelet processing, which yields significantly better compression performance for temporally coherent frame based video sources. Additionally, high definition digital video, such as SMPTE-274M (1080i) are supported using two or more ADV202 devices.

The video interface can support video data or still image data input/output. 8, 10, 12-bit single or multiplexed components, dual-lane 8, 10, or 12-bit components as explained on the following pages. The VDATA interface supports digital video in YCbCr format in single input mode or Y and CbCr in dual lane input mode. YCbCr data must be in 4:2:2 format.

Video data can be input/output in several different modes on the VDATA bus. In all of these modes the pixel clock must be input on the VCLK pin.

5.1.1 EAV/SAV mode:
Accepts video with embedded EAV/SAV codes where the YCbCr data is interleaved onto a single bus.
5.1.2 HVF mode:
Accepts video data accompanied with separate H, V, F signals where YCbCr data is interleaved onto a single bus.
5.1.3 Extended mode:
Y and CrCb are on separate buses accompanied by EAV/SAV codes.
5.1.4 Raw Video mode:
This mode is used for still picture data and non-standard video. VFRM, VSTRB, VRDY are used to program the dimension of the image.

5.1.5 High Speed Video mode:
This mode is used for applications were video data is clocked into the part at higher rates than 27MHz.

5.2 Host Interface [HDATA bus]

The ADV202 can connect directly to a wide variety of host processors and ASICs using an asynchronous SRAM-style interface, DMA accesses or JDATA mode interface. The ADV202 supports 16 and 32-bit buses for control and 8/16/ 32-bit buses for data transfers. The control and data channel bus widths can be specified independently which allows the ADV202 to support applications that require control and data buses of different width. The host interface is used for configuration, control and status functions as well as for transferring compressed data streams. It can be used for

uncompressed data transfers in certain modes. The host interface may be shared by as many as four concurrent data streams in addition to control and status communications. The data streams are:

1) uncompressed tile data [for example: still image data].
2) fully encoded JPEG2000 code stream (or unpackaged code blocks),
3) code-block attributes, and
4) ancillary data.

The ADV202 uses big endian byte alignment for 16 and 32-bit transfers. All data is left justified [most significant bit].

5.2.1 Pixel input on Host interface

Pixel input on the host interface supports 8/10/12/14 or 16-bit raw pixel data formats. It can be used for pixel [still image] input/output or compressed video output. Since there are no timing codes or sync signals associated with the input data on the host interface, dimension registers and internal counters are used and must be programmed to indicate start and end of frame. Refer to the Tech-Note “ADV202 in HIPI mode” for detail on how to use the ADV202 in this mode.

5.2.2 Host Bus Configuration

To provide maximum flexibility, the Host interface provides several configurations to meet particular system requirements. The default bus mode uses the same pins to transfer both control/status and data to/from the ADV202. In this mode, the ADV202 can support 16 or 32-bit control transfers and 8/16/32-bit data transfers. The size of the control/status and data busses can be selected independently. This allows for example a 16-bit micro controller to configure and control the ADV202 while still providing 32-bit data transfers to an ASIC or external memory system.

5.2.3 Direct and Indirect registers

In order to minimize pin count and cost, the number of address pins has been limited to four, which yields a total direct address space of 16 locations. These 16 locations are the ones most commonly used by the external controller and are therefore accessible directly. All other registers in the ADV202 can be accessed indirectly through the use of the IADDR and IDATA register.

5.2.4 Control register access

With the exception of the indirect address and data registers (IADDR and IDATA) all control/status registers in the ADV202 are 16 bits wide and are half-word (16-bit) addressable only. When 32-bit host mode is enabled, the upper 16 bits of the HDATA bus are ignored on writes and will return all zeros on reads of 16-bit registers.

5.2.5 Pin Configuration and Bus Modes

The ADV202 provides a wide variety of control and data configurations which allows it to be used in many applications with little or no glue logic. The modes described below are configured using the BUSMODE register. In the following descriptions ‘host’ is used to refer to normal addressed accesses (i.e., CS/RD/WR/ADDR) and ‘data’ refers to external DMA accesses (i.e., DREQ/DACK).

32-Bit host/32-Bit data

In this mode the HDATA<31:0> pins are used to provide full 32-bit wide data access to Pixel, Code, Attr and Ancl FIFOs. The expanded video interface [VDATA] is not available in this mode.

16-Bit host/32-bit data

This mode allows a 16-bit host to configure and communicate with the ADV202 while still allowing 32-bit accesses to the Pixel, Code, Attr and Ancl FIFOs using the external DMA capability. All addressed host accesses are 16-bits and therefore only use the HDATA<15:0> pins. The HDATA<31:16> pins are used to provide the additional 16-bits necessary to support the 32-bit external DMA transfers to/from the FIFOs only. The expanded video interface [VDATA] is not available in this mode.

16-Bit host/16-bit data

This mode uses 16-bit transfers for if used for host or external DMA data transfers. This mode allows for use of the extended pixel interface modes.

16-Bit host/8-bit data JDATA bus mode

This mode provides separate data input/output and host control interface pins. Host control accesses are 16 bits and use HDATA<15:0> while the dedicated data bus uses

JDATA<7:0>. JDATA uses a valid/hold synchronous transfer protocol. The direction of the JDATA bus is determined by the mode of the ADV202. If the ADV202 is encoding (compression) then JDATA<7:0> is an output. If the ADV202 is decoding (de-compressing) then JDATA<7:0> is an input. Host control accesses remain asynchronous. See also 5.4 JDATA mode.

5.3 Stage register

Since the ADV202 contains both 16 and 32-bit registers and its internal memory is mapped as 32-bit data, a mechanism has been provided to allow 16-bit hosts to access these registers and memory locations. This is accomplished with the staging register (STAGE). STAGE is accessed as a 16-bit register using HDATA[15:0]. Prior to writing to the desired register, the STAGE register must be written with the upper [most significant] half-word. When the host subsequently writes the lower half-word to the desired control register, HDATA is combined with the previously staged value to create the required 32-bit value that will be written. When a register is read, the upper [most significant] half-word is returned immediately on HDATA and the lower half-word can be retrieved by reading the STAGE register on a subsequent access. Additional information for using the STAGE register can be found in the register section. NOTE: this does not apply to the four data channels (PIXEL, CODE, ATTR or ANCL). These channels are always accessed at the specified data width and do not require the use of the STAGE register.

5.4 JDATA mode

The JDATA mode is typically only used when the dedicated video interface [VDATA] is also enabled. This mode allows code stream data [compressed data compliant to JPEG 2000] to be input or output on a single dedicated 8-bit bus (JDATA<7:0>). The bus will always be an output during compression operation, and will be an input during decompression. A two pin handshake is used to transfer data over this synchronous interface. VALID is used to indicate that the ADV202 is ready to provide/accept data and is always an output. HOLD is always an input and is asserted by the host if it can not accept/provide data. For example, JDATA mode allows real time applications, where pixel data is input over the VDATA bus while the compressed data steam is output over the JDATA bus.

5.5 External DMA Interface

The External DMA Interface is provided to enable high-bandwidth data I/O between an external DMA controller and the ADV202 data FIFOs. There are two independent DMA channels which can each be assigned to any one of the four data stream FIFOs [Pixel/Code/Attribute/Ancillary].

The controller supports asynchronous DMA using a Data-Request/Data-Acknowledge (DREQ/DACK) protocol in either single or burst access modes. Additional functionality is provided for single address compatibility (Fly-By mode) and Dedicated Chip Select (DCS) modes.

5.6 Serial Communication Ports

This port is used to provide serial communication to/from the ADV202 where the ADV202 always is the SPI master.

6. INPUT FORMATS

The ADV202 supports a wide variety of formats for uncompressed video and still image data [refer to 8.4 Input formats]. The actual interface and bus modes selected for transferring uncompressed data dictates the allowed size of the input data and the number of samples transferred with each access. The host interface can support 8, 10, 12, 14, 16-bit data formats. The video interface can support video data or still image data input/output. Supported formats are 8, 10, 12 or 16-bit single or 2x8-bit, 2x10-bit, 2x12 bit multiplexed formats [refer to 8.4 Input formats]. All possible formats are listed in section 8 of this datasheet. All formats can support less precision than provided by specifying the actual data width/precision in the PMODE register

The maximum allowable data input rate is limited by using irreversible or reversible compression modes and the data width (or precision) of the input samples.

Table 1, 2, 3 should be used for determining maximum data input rate.

Table 1: Maximum pixel data input rates for 144-pin package

INTER FACE	COMPRESSION MODES	INPUT FORMAT	INPUT RATE LIMIT [ACTIVE RESOLUTION]	APPROX. MIN. PEAK OUTPUT RATE [COMPRESSED DATA]*	APPROX. MAX. OUTPUT RATE [COMPRESSED DATA]*
HDATA	irreversible	8-bit data	45 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	irreversible	10-bit data	45 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	irreversible	12-bit data	45 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	irreversible	16-bit data	45 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	8-bit data	40 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	10-bit data	32 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	12-bit data	27 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	14-bit data	23 Msamples/sec	130 Mbits/sec	200 Mbits/sec
VDATA	irreversible	8-bit data	65 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	irreversible	10-bit data	65 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	irreversible	12-bit data	65 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	8-bit data	40 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	10-bit data	32 Msamples/sec	130 Mbits/sec	200 Mbits/sec
	reversible	12-bit data	27 Msamples/sec	130 Mbits/sec	200 Mbits/sec

Table 2: Maximum pixel data input rates for 121-pin package

INTER FACE	COMPRESSION MODES	INPUT FORMAT	INPUT RATE LIMIT [ACTIVE RESOLUTION]	APPROX. MIN PEAK OUTPUT RATE [COMPRESSED DATA]*	APPROX. MAX. OUTPUT [COMPRESSED DATA]*
HDATA	irreversible	8-bit data	34 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	irreversible	10-bit data	34 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	irreversible	12-bit data	34 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	irreversible	16-bit data	34 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	reversible	8-bit data	30 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	reversible	10-bit data	24 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	reversible	12-bit data	20 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	reversible	14-bit data	17 Msamples/sec	98 Mbits/sec	150 Mbits/sec
VDATA	irreversible	8-bit data	48 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	irreversible	10-bit data	48 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	irreversible	12-bit data	48 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	reversible	8-bit data	30 Msamples/sec	98 Mbits/sec	150 Mbits/sec
	reversible	10-bit data	24 Msamples/sec]	98 Mbits/sec	150 Mbits/sec
	reversible	12-bit data	20 Msamples/sec]	98 Mbits/sec	150 Mbits/sec

Table 3: Maximum supported tile width for data input on HDATA and VDATA bus

Compression Mode	Input format	Tile/Precinct max. width
9/7i	Single component	2048
9/7i	Two component	1024 each
9/7i	Three component	1024 [Y]
5/3i	Single component	4096
5/3i	Two component	2048 [each]
5/3i	Three component	2048 [Y]
5/3r	Single component	4096
5/3r	Two component	2048
5/3r	Three component	1024

* Min. peak output rate or guaranteed sustained output rate.
Max. output rate or a output rate above this value is not possible.

7. PIN CONFIGURATIONS

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
MCLK	1	L9	L12	I	System input clock. For more detail refer to PLL register section. Maximum input frequency on MCLK is 74.25MHz.
$\overline{\text{RESET}}$	1	L7	L11	I	Reset. Causes the ADV202 to immediately reset. For more detail refer to Boot mode register section. CS/, RD/, WE/, DACK0/, DACK1/, DREQ0, DREQ1 must be held high when a RESET/ is applied.
HDATA<15:0>	16	D4-D1 C5-C3 B5, B4, C2 B3-B1 A2, A6-A5	F4, E1-E3, D1-D3, C1-C3, B1-B3, A2, A3, A4	I/O	Host data bus. With HDATA<23:16>, <27:24>, <31:28> these pins make up the 32-bit wide host data bus. The async host interface is interfaced together with ADDR<3:0>,CS, WE, RD, ACK. Unused HDATA pins should be pulled-down via a 10K resistor.
ADDR<3:0>	4	H11, K8, H10, J9	J12, J11, J10, H12	I	Address bus for the host interface.
$\overline{\text{CS}}$	1	J8	H11	I	Chip select.This signal is used to qualify addressed read and write access to the ADV202 using the host interface.
$\overline{\text{WE}}$ $\overline{\text{RDFB}}$	1	J7	H10	I	Write Enable used with the host interface. Read Enable when “Fly-by” DMA is enabled. Note: simultaneous assertion of WE/ and DACK/ low will activate the HDATA bus, even if the DMA channels are disabled.
$\overline{\text{RD}}$ $\overline{\text{WEFB}}$	1	H9	G12	I	Read Enable used with the host interface. Write Enable when “Fly-by” DMA is enabled. Note: simultaneous assertion of RD/ and DACK/ low will activate the HDATA bus, even if the DMA channels are disabled.

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
$\overline{\text{ACK}}$	1	H8	G11	O	Acknowledge. Used with the host interface. This signal indicates that the last register access was successful. Note: due to synchronization issues, control and status register accesses may incur an additional delay, so the host software should wait for acknowledgement from the ADV202. Accesses to the FIFOs [External DMA modes], on the other hand, are guaranteed to occur immediately and should not wait for $\overline{\text{ACK}}$ provided that the timing constraints are observed. If ACK/ is shared with more than one device, ACK/ should be connected to a pull-up resistor [10K] and the PLL_HI register, bit 4 must be set to '1'.
$\overline{\text{IRQ}}$	1	G10	G10	O	Interrupt. This pin indicates that the ADV202 requires the attention of the host processor. This pin can be programmed to indicate the status of the internal interrupt conditions within the ADV202. The interrupt sources are enabled via bits in register EIRQIE.

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
$\overline{\text{DREQ0}}$	1	F8	F12	O	Data Request for External DMA interface. Indicates that the ADV202 is ready to send/receive data from/to the FIFO assigned to DMA channel 0. Must be held high at all times if the DMA interface is not used, even if the DMA channels are disabled.
$\overline{\text{FSRQ0}}$				O	Used in DCS-DMA mode. Service request from the FIFO assigned to channel 0 (asynchronous mode).
$\overline{\text{VALID}}$				O	Valid indication for JDATA input/output stream. Polarity of this pin is programmable in the EDMOD0 register. VALID is always an output.
CFG<1>				I	Boot mode configuration. This pin is read on reset to determine the boot configuration of the on-board processor. The pin should be tied to IOVDD or IOGND through a 10 Kohm resistor. Details of boot modes can be found in the BOOT register description below.
$\overline{\text{DACK0}}$	1	F9	F11	I	Data Acknowledge for External DMA interface. Signal from the host CPU that indicates that the data transfer request ($\overline{\text{DREQ0}}$) has been acknowledged and data transfer can proceed. Must be held high at all times if the DMA interface is not used, even if the DMA channels are disabled.
$\overline{\text{HOLD}}$				I	External hold indication for JDATA input/output stream. Polarity is programmable in the EDMOD0 register. This pin is always and input.
$\overline{\text{FCS0}}$				I	Used in DCS-DMA mode. Chip select for the FIFO assigned to channel 0 (asynchronous mode).

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
$\overline{\text{DREQ1}}$	1	F10	F10	O	Data Request for External DMA interface. Indicates that the ADV202 is ready to send/receive data from/to the FIFO assigned to DMA channel 1. Must be held high at all times if the DMA interface is not used, even if the DMA channels are disabled.
$\overline{\text{FSRQ1}}$				O	Used in DCS-DMA mode. Service request from the FIFO assigned to channel 1 (asynchronous mode).
CFG<2>				I	Boot mode configuration. This pin is read on reset to determine the boot configuration of the on-board processor. The pin should be tied to IOVDD or IOGND through a 10 Kohm resistor. Details of boot modes can be found in the BOOT register description below.
$\overline{\text{DACK1}}$	1	G9	F9	I	Data Acknowledge for External DMA interface. Signal from the host CPU that indicates that the data transfer request (DREQ1) has been acknowledged and data transfer can proceed. Must be held high at all times if the DMA interface is not used, even if the DMA channels are disabled.
$\overline{\text{FCS1}}$				I	Used in DCS-DMA mode. Chip select for the FIFO assigned to channel 1 (asynchronous mode).
HDATA<31:28>	4	J2-J4,H1	K3, J1-J3	I/O	Host expansion bus.
JDATA<7:4>				I/O	JData bus (JDATA mode)
HDATA<27:24>	4	H2-H4, G4	J4, H1-H3	I/O	Host expansion bus.
JDATA<3:0>				I/O	JData bus (JDATA mode)
VDATA<23:20>				I/O	Video Data expansion bus

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
HDATA<23:16>	8	G3, G2, F4, F3, F2 E2, E3, E4	H4, G1-G4, F1-F3	I/O	Host expansion bus.
VDATA<19:12>				I/O	Video Data expansion bus. Extended Pixel Interface mode. Used for video formats which use Y and CrCb on separate buses.
SCOMM<7>	8	L2	M2	I/O	When not used should be tied low.
SCOMM<6>		L3	M3	I/O	When not used should be tied low.
SCOMM<5>		L4	M4	I/O	This pin must be used in multiple chip mode to align the outputs of two or more ADV202s. For more detail refer to Applications section and the 'ADV202 multi-chip application' Appnote. When not used should be tied low.
SCOMM<4>		K1	L1	O	LCODE output in encode mode. When LCODE is enabled, the output on this pin indicates on a high transition that the last data word for a field has been read from the FIFO. For an 8-bit interface, i.e JDATA, LCODE will be asserted for 4 consecutive bytes and is enabled by default.
SCOMM<3>		K2	L2	O	SPI interface: S_CSEL. Should be tied low when not used. Only used with boot mode 6.
SCOMM<2>		L5	L3	O	SPI interface: S_MO. Should be tied low when not used. Only used with boot mode 6.
SCOMM<1>		K4	K1	I	SPI interface: S_MI. Should be tied low when not used. Only used with boot mode 6.
SCOMM<0>		K3	K2	O	SPI interface: S_CLK. Should be tied low when not used. Only used with boot mode 6.

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
VCLK	1	E9	E12	I	Video Data clock. Must be supplied if video data is input/output on the VDATA bus.
VDATA<11:0>	12	D11, D10, C7, C9, C10, B7, B8, B9, B11, B10, A7, A10	D10-D12, C10-C12, B10-B12, A9-A11	I/O	Video Data. Unused pins should be pulled down via a 10K resistor.
VSYNC VFRM	1	D8	E10	I/O	Vertical sync for video mode. Raw pixel mode framing signal. Indicates first sample of a tile when asserted high.
HSYNC VRDY	1	D9	E11	I/O O	Horizontal sync for video mode Raw pixel mode ready signal
FIELD VSTRB	1	E10	E9	I/O I	Field sync for video mode Raw pixel mode transfer strobe.
TEST1	1	J6	K12	I	Should be connected to ground via a pull-down resistor.
TEST2	1	K9	K11	I	Should be connected to ground via a pull-down resistor.
TEST3	1	J10	K10	I	Should be connected to ground via a pull-down resistor.
TEST4	1	L6	M9	I	Should be connected to ground via a pull-down resistor.
TEST5	1	K10	L10	O	No connect.
VDD		A3, A8, D7,H7	B6, B7, C6, C7, D6, D7, J6, J7, K6, K7, L6, L7	P	Positive supply for core

Table 4: Pin configurations

NAME	PINS	121-PIN PACKAGE	144-PIN PACKAGE	I/O	DESCRIPTION
DGND		A1, A11,A4, A9, C1, C11, D6, E1, E5-E7, E11, F1, F5-F7, F11, G1, G5-G7, G11,H6, J1, J11, K11, L1, L8, L11	A1, A5-A8, A12, B5, B8, C5, C8,D5, D8, E4-E8, F5-F8, G5-G9, H5-H9, J5, J8-J9, K5,K8,L5, L8,M1, M5-M8, M12, M11	G	Ground for core
PLLVDD	1	L10	M10	P	Positive supply for PLL
IOVDD		B6, C6, C8 D5,E8, G8, H5, J5, K5, K6, K7,	B4, B9, C4,C9, D4,D9, K4,K9, L4,L9,	P	Positive supply for I/O.

8. DIRECT AND INDIRECT REGISTERS

The following section describes the direct and indirect registers of the ADV202.
The “Getting Started...” application note describes how and which registers to access to operate the ADV202. This section contains a functional description of the registers only.

8.1 Direct Register Definition

Direct register access is required regardless of what interface is used [SPI, JDATA mode, Normal Host mode, all DMA modes etc.]. The direct registers are accessed over the ADDR [3-0], CS/, RD/, WR/, ACK/ pins. The host has to initialize these registers before any application specific operation can take place.

Table 5: Direct Registers

DIRECT REGISTER ADDRESS	NAME	DESCRIPTION
0x00	PIXEL	Pixel FIFO Access Register
0x01	CODE	Compressed Code Stream Access Register
0x02	ATTR	Attribute FIFO Access Register
0x03	ANCL	Ancillary FIFO Access Register
0x04	CMDSTA	Command Stream FIFO
0x05	EIRQIE	External Interrupt Enabled
0x06	EIRQFLG	External interrupt Flags
0x07	SWFLAG	Software Flag Register
0x08	BUSMODE	Bus Mode Configuration Register
0x09	MMODE	Miscellaneous Mode Register
0x0A	STAGE	Staging Register
0x0B	IADDR	Indirect Address Register
0x0C	IDATA	Indirect Data Register
0x0D	BOOT	Boot Mode Register
0x0E	PLL_HI	PLL Control Register - High Byte
0X0F	PLL_LO	PLL Control Register - Low Byte

0x0 PIXEL Pixel FIFO access register R/W

This register is used to access the Pixel data FIFO via normal addressed accesses and generally is used for pixel data input on the host interface [HIPI mode - Host Interface Pixel Interface mode]. The actual number of bits accepted or returned on this register depends on the host data mode selected [BUSMODE]. 8-bit data mode will accept/return data on bits <7:0>, 16-bit mode will accept/return data on bits <15:0> and 32-bit mode will use all bits. Unused bits will be ignored on writes and return zeros on reads. The size of the PIXEL FIFO register is limited to 256x32-bits and can not be changed as for the other FIFOs with the settings in the FFMODE register. A read operation when the PIXEL FIFO is empty or a write when it is full,

will cause the PFERR bit in the EIRQFLG to be asserted.

Bits	Name	Description	Reset Value
31:0	PDATA	Pixel data	undef

0x1 CODE Compressed code stream access register R/W

This register is used to access the JPEG2000 compressed code stream FIFO via normal addressed accesses. It is also used for accessing raw code blocks when the ADV202 is in HIPI mode [Host Interface Pixel Interface mode]. The actual number of bits accepted or returned on this register depends on the host data mode selected [BUSMODE]. 8-bit data mode will accept/return data on bits <7:0>, 16-bit mode will accept/return data on bits <15:0> and 32-bit mode will use all bits. Unused bits will be ignored on writes and return zeros on reads. A read operation when the CODE FIFO is empty or a write when it is full, will cause the DFERR bit in the EIRQFLG to be asserted.

Bits	Name	Description	Reset Value
31:0	CDATA	Compressed data	undef

0x2 ATTR Attribute FIFO access register R/W

This register is used to access the attribute FIFO via normal addressed accesses. The actual number of bits accepted or returned on this register depends on the host data mode selected [BUSMODE].
8-bit data mode will accept/return data on bits <7:0>, 16-bit mode will accept/return data on bits <15:0> and 32-bit mode will use all bits. Unused bits will be ignored on writes and return zeros on reads. A read operation when the ATTR FIFO is empty or a write when it is full, will cause the AFERR bit in the EIRQFLG to be asserted.

Bits	Name	Description	Reset Value
31:0	ADATA	Code-block attribute data	undef

0x3 ANCL Ancillary FIFO access register R/W

This register is used to access the Ancillary data FIFO via normal addressed accesses. The actual number of bits accepted or returned on this register depends on the host data mode selected [BUSMODE]. 8-bit data mode will accept/return data on bits <7:0>, 16-bit mode will accept/return data on bits <15:0> and 32-bit mode will use all bits. Unused bits will be ignored on writes and return zeros on reads. A read operation when the ANCL FIFO is empty or a write when it is full, will cause the NFERR bit in the EIRQFLG to be asserted.

Bits	Name	Description	Reset Value
31:0	NDATA	Ancillary data	undef

0x4 CMDSTA Command stream port WO

Bits	Name	Description	Reset Value
31:0	CMDSTA	*Reserved. For internal use only.*	undef

0x5 EIRQIE External interrupt enables R/W

This register is used to enable conditions that will cause an external interrupt to occur on the IRQ/ pin. Refer to the “Getting Started with the ADV202” for details on how to use this register.

Bits	Name	Description	Reset Value
0	PFTH	Pixel FIFO threshold condition exists (Level sensitive)	0
1	DFTH	Data FIFO threshold condition exists (Level sensitive)	0
2	AFTH	Attribute FIFO threshold condition exists (Level sensitive)	0
3	NFTH	Ancillary FIFO threshold condition exists (Level sensitive)	0
4	PFERR	Pixel FIFO has overflowed or underflowed	0
5	DFERR	Data FIFO has overflowed or underflowed.	0
6	AFERR	Attribute FIFO has overflowed or underflowed.	0
7	NFERR	Ancillary FIFO has overflowed or underflowed.	0
8	*Reserved*	Always write 0	0
9	*Reserved*	Always write 0	0
10	SWIRQ0	Software interrupt 0. Refer to “Getting Started with the ADV202” application note for more detail.	0
11	SWIRQ1	Software interrupt 1. Refer to “ADV202 multi-chip application” application note for more detail.	0
12	SWIRQ2	Software interrupt 2. Reserved for future use.	0
13	INDERR	Error occurred on indirect register access	0
14	IHWDIRQ	Internal hardware interrupt	0
15	FERR	The ADV202 has encountered an unexpected fatal error	0

0x6 EIRQFLG External interrupt flags R/W

This register indicates which external interrupt conditions are currently active. The bits in this register correspond directly to the bits in the external interrupt enable register (EIRQIE). Individual interrupts are cleared by writing a '1' in the proper bit position of this register (i.e., "Write 1 to clear".)

Bits	Name	Description	Reset Value
0	PFTH	Pixel FIFO threshold condition exists (Level sensitive)	1
1	DFTH	Data FIFO threshold condition exists (Level sensitive)	1
2	AFTH	Attribute FIFO threshold condition exists (Level sensitive)	1
3	NFTH	Ancillary FIFO threshold condition exists (Level sensitive)	1
4	PFERR	Pixel FIFO has overflowed or underflowed	0
5	DFERR	Data FIFO has overflowed or underflowed.	0
6	AFERR	Attribute FIFO has overflowed or underflowed.	0
7	NFERR	Ancillary FIFO has overflowed or underflowed.	0
8	*Reserved*		0
9	*Reserved*		0
10	SWIRQ0	Software interrupt 0	0
11	SWIRQ1	Software interrupt 1	0
12	SWIRQ2	Software interrupt 2	0
13	INDERR	Error occurred on indirect register access	0
14	IHWDIRQ	Internal hardware interrupt	0
15	FERR	The ADV202 has encountered an unexpected fatal error	0

0x7 SWFLAG Software flag register RO

Bits	Name	Description	Reset Value
15:0	SWFLAGS	Reserved. For internal use only	undef

0x8 BUSMODE Bus mode configuration register R/W

Configures host control and data busses.

Bit(s)	Name	Description	Reset Value
1:0	HWIDTH	Host control data width, 1=half-word [16-bits], 2=word[32-bits]	1
3:2	DWIDTH	DMA data width, 0=byte [8-bits], 1=half-word [16-bits], 2=word [32-bits]	1

Bit(s)	Name	Description	Reset Value
6:4	BCFG	Bus configuration 0 Normal. HDATA [31:16] are available for host or data transfers according to the settings of HWIDTH and DWIDTH. For normal read/write access HWIDTH and DWIDTH are set to 1 or 2. For DMA mode HDWIDTH and DWIDTH can take any settings. 1 JDATA Mode (JDATA). Enables JDATA[7:0]. HWIDTH/DWIDTH can not be set to 2. 2 Extended Video Interface. Enables HDATA [27:16] to be used as a video extension bus where Y and CbCr data is used on separate buses. HWIDTH/DWIDTH can not be set to 2. 3-7 Reserved	0
7	Reserved	Reserved for future use; always write 0.	0
15:8	*N/A*	Reserved for future use; always write 0.	undef

0x9 MMODE Misc mode configuration register R/W

Configures miscellaneous functions for Indirect Access.

Bit(s)	Name	Description	Reset Value
1:0	IWIDTH	Indirect access width 0 Byte [8-bits] 1 Half-word [16-bits] 2 Word [32-bits] 3 Invalid	1
3:2	IAUTOSTP	IADDR step size. IADDR can automatically increment/decrement after each internal access based on the value of IAUTOINC. Increment/decrement is selected using bit 4 in this register. 0 Byte addressing 1 Half-word addressing 2 Word addressing. Note: all internal/indirect registers are on word boundaries. 3 Disable auto increment/decrement	2
4	IAUTOMOD	Indirect address modify mode 0=increment, 1=decrement	0
5	CTLREGAM	Control register address mode. Enable full indirect address map 1 Control register mode. The upper 16-bits of the internal address are forced to the beginning of the control register address map. This eliminates the need to set the upper 16-bits of the indirect address register prior to each access. 0 Full address mode. Provides full access to the ADV202's internal address space.	0
15:6	*Reserved*	Reserved for future use; always write 0.	0

0xA STAGE Staging register R/W

This register is used to access Words (32-bit) when using a 16-bit host control bus. The STAGE register acts as a holding register. When writing to a 32-bit register with a 16-bit host, the host first writes the *upper* (most significant) half-word to the STAGE register prior to writing the *lower* half-word to the desired register address. When reading a 32-bit register with a 16-bit host, the host first reads the desired register to acquire the *upper* half-word, and then reads the STAGE register to acquire the *lower* half-word.

Bits	Name	Description	Reset Value
15:0	STG	Staged value for a write access, or from a read access.	undef

In the following examples, bit <31> is denoted as the most significant bit and bit <0> is the least significant.

Ex 1. Writing to the 32-bit IDATA register with a 16-bit host. The 16-bit host provides the data on the HDATA<15:0> pins.

First write:	IDATA<31:16>	is written to the STAGE register (address 0xA)
Second write:	IDATA<15:0>	is written to the IDATA register (address 0xC)

Ex 2. Reading the 32-bit IDATA register with a 16-bit host. In this mode, half-word(16-bit) data will be returned on the HDATA<15:0> pins.

First read:	IDATA<31:16>	is read from IDATA register (address 0xC)
Second read:	IDATA<15:0>	is read from STAGE register (address 0xA)

0xB IADDR Indirect address register R/W

This register is used to set the address for indirect register accesses. The indirect address may optionally be auto-incremented by setting the IAUTOMOD and IAUTOSTP fields in the MMODE register.

Bits	Name	Description	Reset Value
31:0	ADDR	Indirect address register	undef

0xC IDATA Indirect data register R/W

This register is used to access indirect registers.

Bits	Name	Description	Reset Value
31:0	DATA	Indirect data register	undef

0xD BOOT BOOT mode register R/W

This register is used to read/set the boot mode and initiate a soft or hard reset of the ADV202. Note: A hard reset, via the reset pin or setting the HARDRST bit causes bits 2:0 to be loaded from the configuration pins as specified below, all other bits in this register will be set as specified in the table. A soft reset, via setting the SOFTRST bit will ONLY clear the SOFTRST bit, all other bits will remain unchanged.

Bits	Name	Description	Reset Value
2:0	BOOT-MODE	Boot mode. These bits are used to select various boot modes of the ADV202 after reset to load specific instruction sets into memory. The boot mode can be configured via hardware [over the CFG pins] or via software [via firmware]. In a hardware configuration, after a hard reset the boot mode will be set to the values on the configuration pins CFG<2-1>. These bits do not get reset on a soft reset, see HARDRST and SOFTRST bit definitions below. The first boot mode after power-up is set by the CFG pins. Only boot modes 2, 4 and 6 are available in hardware bootmode. **SOFTWARE BOOTMODE** 0 Reserved. For internal use only. Do not use. 1 Reserved. For internal use only. Do not use. 2 No-Boot Host mode, ADV202 does not boot but all internal registers and memory are accessible through normal host I/O operations. Refer to the “Getting started with the ADV202” application note. 3 Reserved. 4 SoC boot mode. The Embedded Software Framework [ESF] gets control and establishes communications with the host. This mode will be available on the released version of the ADV202. 5 Co-Processor Boot. Is used in conjunction with No-Boot Host mode and starts loaded firmware. Refer to the “Getting started with the ADV202” application note. 6 Boot firmware over SPI from external flash memory. This mode will be available on the released version of the ADV202. 7 Reserved. **HARDWARE BOOTMODE** 0 Reserved. For internal use only. Do not use. 1 Not available. 2 No-Boot host mode, ADV202 does not boot but all internal registers and memory are accessible through normal host I/O operations. Refer to the “Getting started with the ADV202” application note. 3 Not available. 4 SoC boot mode. The Embedded Software Framework [ESF] gets control and establishes communications with the host. This mode will be available on the released version of the ADV202. 5 Not available. 6 Boot firmware over SPI from external flash memory. 7 Not available.	See Text

0xE PLL_HI PLL control register (high byte) R/W

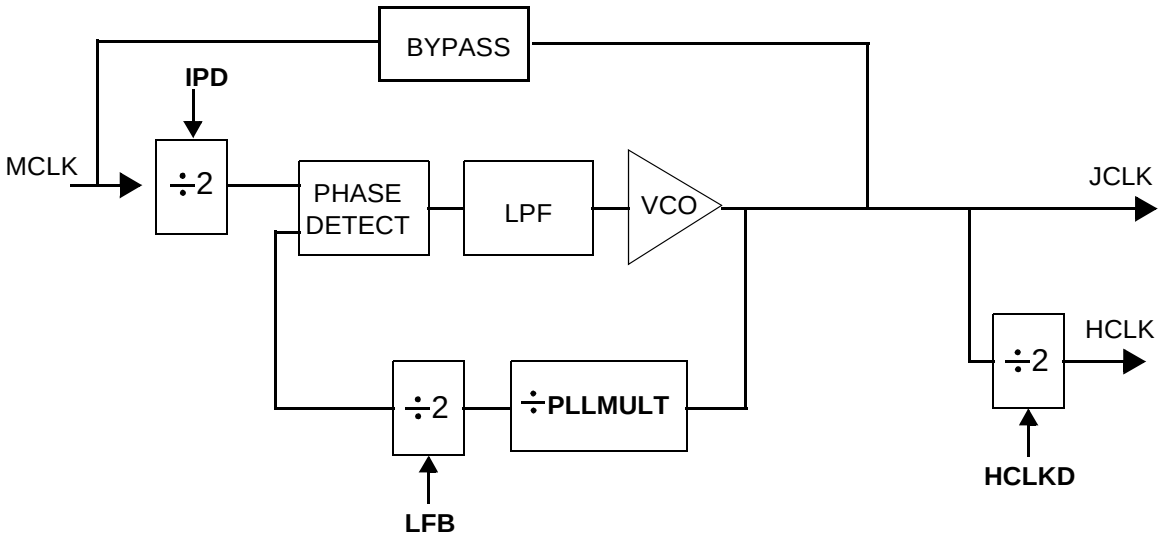
This register is used to configure the on chip PLL. HDATA<7:0> are used to set this register regardless of host mode. Internally, the ADV202 uses two clock domains which are generated by an on chip Phase Locked Loop from the input clock MCLK. JCLK is used to clock all of the JPEG2000 specific hardware blocks and HCLK is used to clock the embedded processor system. A block diagram of the PLL and its control parameters are shown in the figure below.

Bits	Name	Description	Reset Value
0	PLLPDN	Power down enable = 1	0
1	BYPASS	Bypass enabled = 1	0
2	*reserved*	always write 0	0
3	HCLKD	HCLK divider enabled = 1	1
4	ACK_FN	ACK/ is in high-impedance mode = 0 ACK/ is configured as open drain output = 1	
6:4	*reserved*	ADI use only; always write 0	undef
7	SM	ADI use only; always write 0	0
15:8	*N/A*	Not applicable	undef

0xF PLL_LO PLL control register (low byte) R/W

This register is used to configure the on chip PLL. HDATA<7:0> are used to set this register regardless of host mode.

Bits	Name	Description	Reset Value
4:0	PLLMULT	Multiplier [values from 0 to 31]	6
5	LFB	Loop feedback divider enable = 1	0
6	*reserved*	Always write 0	0
7	IPD	Input clock divider enable = 1	0
15:8	*N/A*	Not applicable	undef



Equivalent transfer function: $\frac{JCLK}{MCLK} = \frac{PLLMULT * LFB}{IPD}$

Figure 9. PLL architecture and control functions

Table 1: Recommended Register Settings for PLL registers[1]

IPD	LFB	PLLMULT	HCLKD	HCLK	JCLK
0	0	N	0	N * MCLK	N * MCLK
0	0	N	1	N * MCLK / 2	N * MCLK
0	1	N	0	2 * N * MCLK	2 * N * MCLK
0	1	N	1	N * MCLK	2 * N * MCLK
1	0	N	0	[N * MCLK] / 2	[N * MCLK] / 2
1	0	N	1	[N * MCLK] / 4	N * MCLK / 2
1	1	N	0	N * MCLK	N * MCLK
1	1	N	1	N * MCLK / 2	N * MCLK

Note 1:
The PLL can be programmed to have any possible final multiplier value as long as:
1. JCLK >50MHz and <150MHz.
2. HCLK <115 MHz.
3. JCLK >= 2x VCLK for single component input.
4. JCLK >= 2x VLCK for YCrCb [4:2:2] input for silicon rev 0.0.
5. In JDATA Mode [JDATA] JCLK must be 4x MCLK or higher.
6. The max. burst frequency for external DMA modes is <= 0.44 JCLK
7. If MCLK is larger than 50 MHz, the input clock divider must be enabled, i.e. IPD set to '1'.

If any settings in the PLL registers are changed, a period of 20us should be allowed for the part to re-lock after changing these settings.

Example:

For lowest power consumption an MCLK=27MHz is recommended for a standard definition CCIR656 input. The PLL circuit is recommended to be set to a total multiplier of 3. This will set JCLK and HCLK to 81MHz.

Table 2: Recommended register values for PLL registers

VIDEO STANDARD	CLKIN FREQUENCY ON MCLK	PLL_HI	PLL_LOW
SMPTE125M or ITU-R.BT656 [NTSC or PAL]	27MHz	0x0008	0x0004
SMPTE293M [525p]	27MHz	0x0008	0x0004
ITU-R.BT1358 [625p]	27MHz	0x0008	0x0004
SMPTE274M [1080i]	74.25MHz	0x0008	0x0084

8.2 Indirect Register Definition

These registers are generally accessed by the ESF [Embedded Software Framework] or downloadable firmware only. In this case indirect registers are configured automatically depending on which functions are selected with the ESF or firmware.
In certain modes, for instance custom specific input format or HIPI mode, indirect registers have to be accessed by the user through the use of the IADDR and IDATA registers. The indirect register address space starts at internal address 0xFFFF0000.

If the CTRLREGAM bit in the direct register MMODE is set to FULL ADDRESS MODE, then the complete 32-bit address must be loaded into IADDR before accessing the desired indirect register. Otherwise, the user only needs to load the 16 least significant bits of the desired register address.

Table 3: Indirect Registers

INDIRECT REGISTER ADDRESS	NAME	DESCRIPTION
0xFFFF0400	PMODE1	Pixel/Video Format
0xFFFF0404	COMP_CNT_STATUS	Horizontal count
0xFFFF0408	LINE_CNT_STATUS	Vertical count
0xFFFF040C	XTOT	Total Samples per line
0xFFFF0410	YTOT	Total Lines per frame
0xFFFF0414	F0_START	Start Line of Field 0 [F0]
0xFFFF0418	F1_START	Start Line of Field 1 [F1]
0xFFFF041C	V0_START	Start of active video Field 0 [F0]
0xFFFF0420	V1_START	Start of active video Field 1 [F1]
0xFFFF0424	V0_END	End of active video Field 0 [F0]
0xFFFF0428	V1_END	End of active video Field 1 [F1]
0xFFFF042C	PIXEL_START	Horizontal start of active video
0xFFFF0430	PIXEL_END	Horizontal end of active video
0xFFFF0434	PI_STATUS	Pixel/Video Status Register
0xFFFF0440	MS_CNT_DEL	Master/Slave Delay
0xFFFF0444	LINE_CNT_INTERRUPT	Line count interrupt
0xFFFF0448	PMODE2	Pixel Mode 2
0xFFFF044C	VMODE	Video Mode
0xFFFF1408	EDMOD0	External DMA mode register 0
0xFFFF140C	EDMOD1	External DMA mode register 1
0xFFFF1418	FFMODE	FIFO mode register
0xFFFF1410	FFTHRP	FIFO Threshold for Pixel FIFO
0xFFFF141C	FFTHRC	FIFO Threshold for Code FIFO
0xFFFF1420	FFTHRA	FIFO Threshold for ATTR FIFO
0xFFFF1424	FFTHRN	FIFO Threshold for ANCL FIFO
0xFFFF1414	FFCNTP	FIFO Full/Empty count for PIXEL FIFO
0xFFFF1428	FFCNTC	FIFO Full/ Empty count for CODE FIFO
0xFFFF142C	FFCNTA	FIFO Full/Empty count for ATTR FIFO
0xFFFF1430	FFCNTN	FIFO Full/Empty count for ANCL FIFO
0xFFFF1434 - 0xFFFF14FC	Reserved	Reserved

0xFFFF0400 PMODE1 R/W

PFMT and PREC are used to configure the VDATA or HDATA bus for a specific pixel format.

Bits	Name	Format	Description	Interface	Reset Value
4:0	PFMT	4	12-bit Single Component	VDATA	0x5
		5	12-bit Cb/Y/Cr/Y interleaved	VDATA	
		6	*Reserved*		
		7	12-bit Cb/Cr interleaved	VDATA	
		8-15	*Reserved*		
		16	24-bit 2x12-bit Packed Single Component	VDATA	
		17	24-bit 2x12-bit Packed CbY/CrY interleaved	VDATA	
		18	*Reserved*		
		19	24-bit 2x12-bit Packed CbCr	VDATA	
		20	32-bit 4x8-bit Single Component	HDATA	
		21	32-bit 4x8-bit Packed YCbYCr	HDATA	
		22	32-bit 4x8-bit Packed YYCbCr	HDATA	
		23	32-bit 4x8-bit Packed CbCrCbCr	HDATA	
		24	32-bit 2x16-bit Packed Single Component	HDATA	
		25	32-bit 2x16-bit Packed YCb/YCr	HDATA	
		26	32-bit 2x16-bit Packed YY/CbCr	HDATA	
		27	32-bit 2x16-bit Packed CbCr	HDATA	
		28-31	*Reserved*		
7:5	*Reserved*		Always write 0		0
10:8	PREC	0	8-bit precision		1
		1	10-bit precision		
		2	12-bit precision		
		3	14-bit precision		
		4	16-bit precision		
		5-7	*Reserved*		
15:11	*Reserved*		Always write 0		0

0xFFFF0404 COMP_CNT_STATUS RO

The value in this register indicates the number of columns [samples] presently read into the Pixel interface. It is a horizontal counter.

Bits	Name	Description	Reset Value
15:0	COMP_CNT_STAT	Sample count.	0x0

0xFFFF0408 LINE_CNT_STATUS RO

The value in this register indicates the number of lines presently read into the Pixel interface. It is a vertical counter.

Bits	Name	Description	Reset Value
15:0	LINE_CNT_STAT	Line Count.	0x0

0xFFFF040C XTOT R/W

This register is used to set total number of samples per line.

Bits	Name	Description	Reset Value
15:0	XTOT	Total samples per line	0x64b

0xFFFF0410 YTOT R/W

This register is used to set the total lines per frame for progressive standards or per field for interlaced standard.

Bits	Name	Description	Reset Value
15:0	YTOT	Total lines per frame	0x20d

0xFFFF0414 F0_START R/W

This register is used to indicate the start of Field 0 in units of number of lines. Line count starts with line 1.

Bits	Name	Description	Reset Value
15:0	F0_START	Start of Field 0. Only used in Decode mode to identify at which line number the Field bit will transition from Field 1 to Field 0.	0x20d

0xFFFF0418 F1_START R/W

This register is used to indicate the start of Field 1 in units of number of lines. Line count starts with line 1.

Bits	Name	Description	Reset Value
15:0	F1_START	Start of field 1. Only used in Decode mode to identify at which line number the Field bit will transition from Field 0 to Field 1.	0x10a

0xFFFF041C V0_START R/W

This register is used to set the first active video line in Field 0 that will be captured by the ADV202. Line count starts with line 1.

Bits	Name	Description	Reset Value
15:0	V0_START	Start of active video in field 0	0x14

0xFFFF0420 V1_START R/W

This register is used to set the first active video line in Field 1 that will be captured by the ADV202. Line count starts with line 1.

Bits	Name	Description	Reset Value
15:0	V1_START	Start of active video in field 1	0x11b

0xFFFF0424 V0_END R/W

This register is used to set the vertical end of the active video in Field 0. This register should hold the value of the last active line number. Line count starts with line 1.

Bits	Name	Description	Reset Value
15:0	V0_END	End of active video in field 0	0x107

0xFFFF0428 V1_END R/W

This register is used to set the vertical end of the active video in Field 1. This register should hold the value of the last active line number. Line count starts with line1.

Bits	Name	Description	Reset Value
15:0	V1_END	End of active video in field 1	0x20d

0xFFFF042C PIXEL_START R/W

This register is used to set the horizontal start of the active video in units of samples.

Bits	Name	Description	Reset Value
15:0	PIXEL_START	Start of horizontal active video	0x1

0xFFFF0430 PIXEL_END R/W

This register is used to set the horizontal end of the active video in units of samples.

Bits	Name	Description	Reset Value
15:0	PIXEL_END	End of horizontal active video	0x5A0

8.3 Video timing and dimension registers

8.3.1 Encode mode

In encode mode the part is always in slave configuration. Input data can be accompanied by separate H,V,F signals or embedded timing codes. In both cases, XTOT, YTOT, V0_START, V1_START, V0_END, V1_END, PIXEL_START, PIXEL_END must reflect the video standard of the input. The part synchronizes itself to the incoming sync signals.

Fields are identified by the incoming FIELD signal or the Field bit in the EAV/SAV codes.

Using the value of registers Vx_START, Vx_END, PIXEL_START, PIXEL_END the active video region to be processed is calculated. This does apply to all input modes using the VDATA bus, in HIPI mode only values for XTOT and YTOT need to be programmed, all other dimension register values are ignored in HI PI mode [refer to “ADV202_HIPI_mode” application note].

8.3.2 Decode slave

In decode slave mode input data can be accompanied by separate H,V,F signals or embedded timing codes. In both cases, XTOT, YTOT, V0_START, V1_START, V0_END, V1_END, PIXEL_START, PIXEL_END must reflect the video standard of the input. The part synchronizes itself to the incoming sync signals. Fields are identified by the incoming FIELD signal or the Field bit in the EAV/SAV codes.

8.3.3 Decode master

VSYNC/VFRM, HSYNC/VRDY, FIELD/VSTRB are generated according to the register settings of: XTOT, YTOT, F0_START, F1_START, F0_END, F1_END, V0_START, V1_START, V0_END, V1_END, PIXEL_START, PIXEL_END. To enable the generation of these timing signals in decode mode, VMODE register must be programmed to decode master mode.
VFRM, VRDY, VSTRB are generated when VMODE is set to RAW_MODE.

The polarity of VSYNC/VFRM, HSYNC/VRDY, FIELD/VSTRB is programmed in PMODE2 with VSYNC_POL, HSYNC_POL, FIELD_POL.

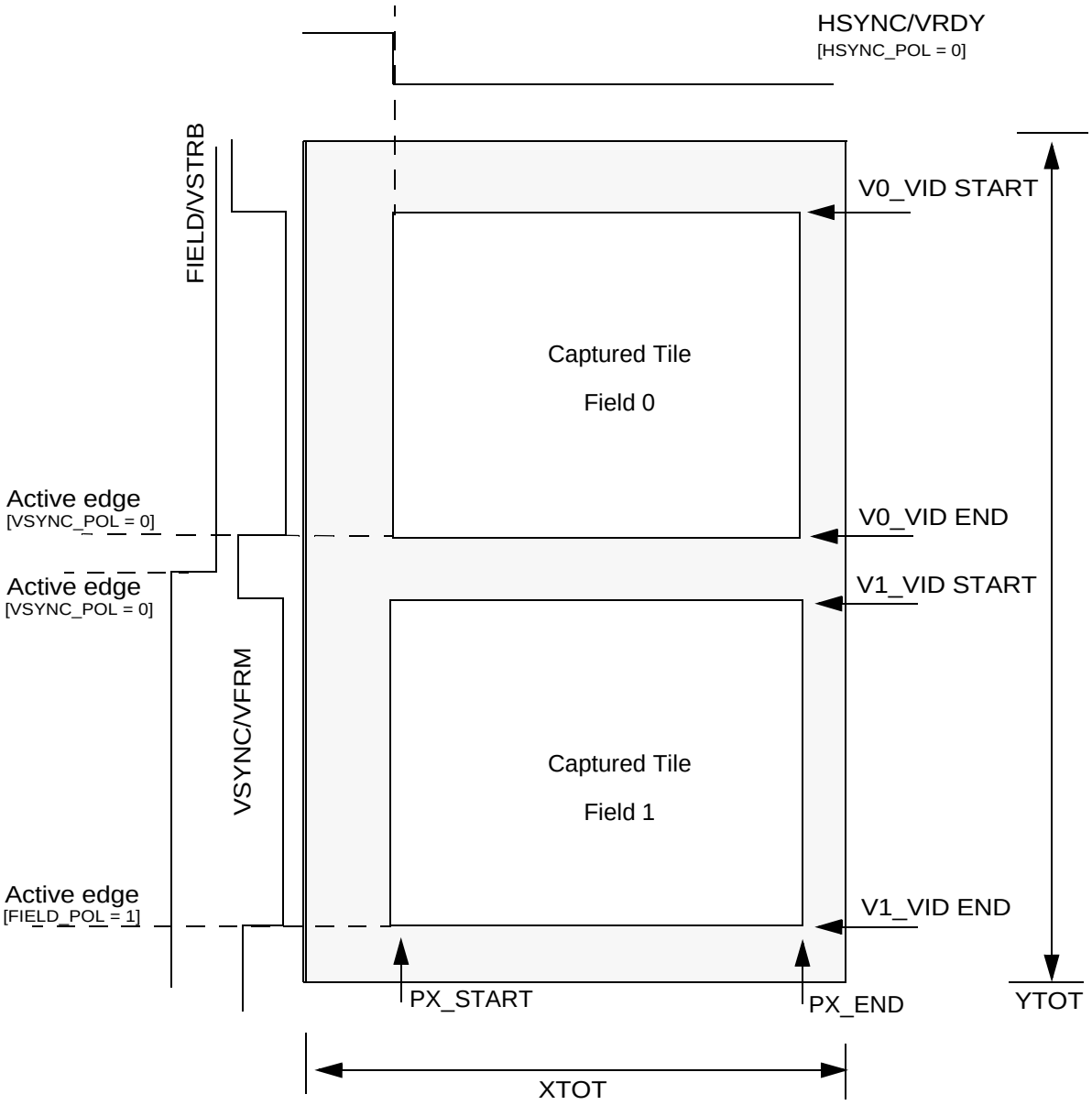


FIGURE 10. Pixel dimension registers [FFFF040C-FFFF0430]

0xFFFF0434 PI_STATUS R

Bits	Name	Description	Reset Value
0	ACTIVE_VIDEO	Active Video Region status. This status bit is activated whenever active video is being processed.	0x0
1	PI_VSTRB	Pixel Interface to Wavelet Transform Strobe status. This status bit is activated when valid data is send from the PI to the WT.	0x0
2	WT_VRDY	Wavelet Transform to Pixel Interface Ready status. This status bit is activated when valid data is send from the WT to the PI.	0x0
3	PI_RDY	Pixel Interface to Host Interface Ready status. This status bit is asserted to signal a valid data transfer between PI and Host interface.	0x0
4	FLD_SYNCED	Input Field status. This status bit is the FSYNC routed from the PI input. It allows the Risc processor visibility to the input field sync signal. If this bit does not match what the Risc processor field status is, then the PI is not synced properly.	0x0
5	FLD_HOLD	Field Hold status. If the PI is not synced properly (mismatch between the input FSYNC and the Risc processors' field status bit), field data will be held off and this status bit asserted. The PI will output a black field instead of the data. Once syncing occurs, the PI will allow data to be output and this bit will be turned off.	0x0
6	F_STATUS	Field Status. In encode mode, this status bit indicates the field status at the PI. In decode mode, this status bit indicates the field status at the Risc processor.	0x0
7	F_IRQ	Pixel Interface Field Interrupt. This interrupt bit is asserted when the PI sends a Field Interrupt to the Risc processor.	0x0
8	L_COUNT_IRQ	Line count interrupt. This interrupt bit is activated when the line number programmed in the LINE_CNT_INTERRUPT register is reached. This interrupt has to be additionally enabled in the PMOD2 register.	0x0
9	F_ST_FIFO_IRQ	Field start reached and FIFO not full - Interrupt. This interrupt bit is set on a field rate basis in decode mode. It is asserted when the PI's internal counters have reached start of active video at the beginning of decoding a field and the PI FIFO is not full.	0x0
10	FIFO_FULL_IRQ	Pixel FIFO full interrupt. This interrupt bit is asserted when the Pixel FIFO has reached full capacity.	0x0

Bits	Name	Description	Reset Value
11	ERR_FLD_END_IRQ	Pixel Interface Video Field Error End - Interrupt. This interrupt bit is asserted if an erroneous field, for example an incomplete field, is passed from the WT to the Pixel Interface in decode mode.	0x0
12	FIFO_EMPTY	Pixel interface FIFO empty status. This status bit is asserted when the Pixel FIFO is empty.	0x0
13	ERR_FLD_BIT_IRQ	Pixel Interface Error Field Bit Interrupt. This interrupt bit is asserted when FLD_IRQ is not cleared by the Risc processor.	0x0
14	ERR_TC_IRQ	Pixel Interface Time Code Interrupt. This interrupt bit is asserted when an erroneous EAV/SAV sequence is being received.	0x0
15	FIFO_FULL	Pixel FIFO full status. This status bit is asserted when the Pixel FIFO has reached its full capacity.	0x0

0xFFFF0440 MS_CNT_DEL RO

Bits	Name	Description	Reset Value
15:0	MS_CNT_DEL	This register is controlled by the ESF or firmware only. Used only in multi-chip sync mode. Refer to AppNote “ADV202 multi-chip application”.	0x0

0xFFFF0444 LINE_CNT_INTERRUPT R/W

Bits	Name	Description	Reset Value
15:0	LINE_CNT_IRQ	Line count interrupt. This register holds the line number value at which a line interrupt will be asserted. L_COUNT_INT_EN must be enabled.	0x0

0xFFFF0448 PMODE2 R/W

Bits	Name	Description	Reset Value
0	VCLK_POL	VCLK active edge 1=pos 0=neg	1
1	VSYNC_POL	VSYNC active edge 1=pos 0=neg	0

Bits	Name	Description	Reset Value
2	HSYNC_POL	HSYNC active edge 1=pos 0=neg	0
3	FIELD_POL	FIELD active edge 1=pos 0=neg	0
4	YUNI	Y unipolar 1=on 0=off Unipolar defines a range from 0 to 255. When this bit is disabled the range is defined to be -127 to +127.	0
5	CUNI	C unipolar 1=on 0=off Unipolar defines a range from 0 to 255. When this bit is disabled the range is defined to be -127 to +127.	0
6	*reserved*	Always write 0	0
7	F_IRQ_EN	Field interrupt enable 1=on 0=off Refer to F_INT in the PI_STATUS register for explanation.	0
8	L_COUNT_IRQ_EN	Line count interrupt enable 1=on 0=off This bit enables the line count interrupt according to the setting in the LINE_CNT_INTERRUPT register.	0
9	F_ST_FIFO_IRQ_EN	Start of field reached and FIFO not full interrupt enable. 1=on 0=off Refer to F_ST_FIFO_IRQ in the PI_STATUS register for explanation.	0
10	FIFO_FULL_IRQ_EN	Enables the FIFO_FULL_IRQ interrupt bit in the PI_STATUS register. 1=on 0=off	0
11	ERR_FLD_END_IRQ _EN	Enables the ERR_FLD_BIT_IRQ in the PI_STATUS register.	0
12	*Reserved*	Always write 0.	0
13	ERR_FLD_BIT_IRQ _EN	Enables the ERR_FLD_BIT_IRQ in the PI_STATUS register.	0
14	ERR_TC_IRQ_EN	Enables the ERR_TC_IRQ in the PI_STATUS register. 1=on 0=off	0
15	*Reserved*	Always write 0 1=on 0=off	0

0xFFFF044C VMODE Video mode R/W

This register sets the video interface basic operating mode.

Bits	Name	Description	Reset Value
0	MAS_SLV	MASTER=1; SLAVE=0 Master or Slave operation is only selectable in Decode mode.	0
1	ENC_DEC	ENCODE=1; DECODE=0	0
2	MP_656	Video input timing control. EAV/SAV mode = 0; for video input with embedded timing codes. HVF mode = 1; for video input with separate H,V,F sync signals.	0
3	DUAL_MODE	Extended mode. Used for any video input data that is more than 12-bit wide on the VDATA bus 1=on 0=off	0
4	HOST_MODE	Video input over HDATA bus 1=on 0=off	0
5	RAW_MODE	Raw video input on the VDATA bus HVF mode is not required to be set in this mode. 1=on 0=off	0
6	*Reserved*	This register should not be written to.	0
7	PRGRSV_SCN	Progressive scan mode; Must be set if video input on the VDATA bus is non-interlaced, i.e 1field/frame. 1=on 0=off	0
8	*Reserved*	Always write 0.	0
9	*Reserved*	Always write 0.	0
10	*Reserved*	Always write 0.	0
11	CNT_RD_EN	Read PI control counters [CO_CNT_STATUS and LINE_CNT_STATUS] 1=on 0=off	0
12:14	*Reserved*	Always write 0	0
15	PI_EN	Pixel interface enable Enables video input over the VDATA bus 1=on 0=off	0

8.4 Input formats programmable in PMODE1 register

8.4.1 Input formats on HDATA bus [8/10 bit data] controlled by PFMT and PREC register settings

8-bit three component YCbYCr 4:2:2

msb(7)	Y_n	lsb(0)
msb(7)	Cb_n	lsb(0)
msb(7)	Y_{n+1}	lsb(0)
msb(7)	Cr_n	lsb(0)
repeat		

8-bit three component YYCbCr 4:2:2

msb(7)	Y_n	lsb(0)
msb(7)	Y_{n+1}	lsb(0)
msb(7)	Cb_n	lsb(0)
msb(7)	Cr_n	lsb(0)
repeat		

8-bit single component

msb(7)	S_n	lsb(0)
repeat		

8- bit two component CbCr

msb(7)	Cb_n	lsb(0)
msb(7)	Cr_n	lsb(0)
repeat		

FIGURE 11. 8-bit input formats on HDATA bus

10-bit three component YCbYCr 4:2:2

msb(9)	Y_n	lsb(0)
msb(9)	Cb_n	lsb(0)
msb(9)	Y_{n+1}	lsb(0)
msb(9)	Cr_n	lsb(0)
repeat		

10-bit three component YYCbCr 4:2:2

msb(9)	Y_n	lsb(0)
msb(9)	Y_{n+1}	lsb(0)
msb(9)	Cb_n	lsb(0)
msb(9)	Cr_n	lsb(0)
repeat		

10-bit two component CbCr

msb(9)	Cb_n	lsb(0)
msb(9)	Cr_n	lsb(0)
repeat		

10-bit single component

msb(9)	S_n	lsb(0)
repeat		

FIGURE 12. 10-bit input formats on HDATA bus

8.4.2 Input formats on HDATA bus [12-/14-bit data] controlled by PFMT and PREC register settings

12-bit three component YCbYCr 4:2:2

msb(11)	Y_n	lsb(0)
msb(11)	Cb_n	lsb(0)
msb(11)	Y_{n+1}	lsb(0)
msb(11)	Cr_n	lsb(0)
repeat		

12-bit three component YYCbCr 4:2:2

msb(11)	Y_n	lsb(0)
msb(11)	Y_{n+1}	lsb(0)
msb(11)	Cb_n	lsb(0)
msb(11)	Cr_n	lsb(0)
repeat		

12-bit single component

msb(11)	S_n	lsb(0)
repeat		

12-bit two component CbCr 4:2:2

msb(11)	Cb_n	lsb(0)
msb(11)	Cr_n	lsb(0)
repeat		

FIGURE 13. 12-bit input formats on HDATA bus

14-bit three component YCbYCr 4:2:2

msb(13)	Y_n	lsb(0)
msb(13)	Cb_n	lsb(0)
msb(13)	Y_{n+1}	lsb(0)
msb(13)	Cr_n	lsb(0)
repeat		

14-bit two component CbCr

msb(13)	Cb_n	lsb(0)
msb(13)	Cr_n	lsb(0)
repeat		

14-bit three component YYCbCr 4:2:2

msb(13)	Y_n	lsb(0)
msb(13)	Y_{n+1}	lsb(0)
msb(13)	Cb_n	lsb(0)
msb(13)	Cr_n	lsb(0)
repeat		

14-bit single component

msb(13)	S_n	lsb(0)
repeat		

FIGURE 14. 14-bit input formats on HDATA bus

8.4.3 Input formats on HDATA bus [16-bit packed] controlled by PFMT and PREC register settings

2x8bit three component YCbYCr 4:2:2

msb(7)	Y_n	lsb(0)	msb(7)	Cb_n	lsb(0)
msb(7)	Y_{n+1}	lsb(0)	msb(7)	Cr_n	lsb(0)
repeat					

2x8bit three component YYCbCr 4:2:2

msb(7)	Y_n	lsb(0)	msb(6)	Y_{n+1}	lsb(0)
msb(7)	Cb_n	lsb(0)	msb(6)	Cr_n	lsb(0)
repeat					

16-bit three component YCbYCr 4:2:2

msb(15)	Y_n	lsb(0)
msb(15)	Cb_n	lsb(0)
msb(15)	Y_{n+1}	lsb(0)
msb(15)	Cr_n	lsb(0)
repeat		

16-bit three component YYCbCr 4:2:2

msb(15)	Y_n	lsb(0)
msb(15)	Y_{n+1}	lsb(0)
msb(15)	Cb_n	lsb(0)
msb(15)	Cr_n	lsb(0)
repeat		

2x8bit two component CbCr

msb(7)	Cb_n	lsb(0)	msb(7)	Cr_n	lsb(0)
repeat					

16-bit two component CbCr

msb(15)	Cb_n	lsb(0)
msb(15)	Cr_n	lsb(0)
repeat		

2x8bit single component

msb(7)	S_n	lsb(0)	msb(7)	S_{n+1}	lsb(0)
repeat					

16-bit single component

msb(15)	S_n	lsb(0)
repeat		

FIGURE 15. 8-bit/16-bit input formats on HDATA bus

8.4.4 Input formats on VDATA bus [8-bit data] controlled by PFMT and PREC register settings

8-bit three component - CbYCrY - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(7)	Cb_n	lsb(0)	[3:0] 0x00
msb(7)	Y_n	lsb(0)	[3:0] 0x00
msb(7)	Cr_n	lsb(0)	[3:0] 0x00
msb(7)	Y_{n+1}	lsb(0)	[3:0] 0x00
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

8-bit two component CbCr - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(7)	Cb_n	lsb(0)	[3:0] 0x00
msb(7)	Cr_n	lsb(0)	[3:0] 0x00
msb(7)	Cb_{n+1}		[3:0] 0x00
msb(7)	Cr_{n+1}	lsb(0)	[3:0] 0x00
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

8-bit single component - HVF Mode

msb(7)	S_n	lsb(0)	[3:0] 0x00
repeat			

8-bit single component - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(7)	S_n	lsb(0)	[3:0] 0x00
msb(7)	S_{n+1}	lsb(0)	[3:0] 0x00
msb(7)	S_{n+2}		[3:0] 0x00
msb(7)	S_{n+3}	lsb(0)	[3:0] 0x00
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

8-bit three component CbYCrY- HVF Mode

msb(7)	Cb_n	lsb(0)	[3:0] 0x00
msb(7)	Y_n	lsb(0)	[3:0] 0x00
msb(7)	Cr_n	lsb(0)	[3:0] 0x00
msb(7)	Y_{n+1}	lsb(0)	[3:0] 0x00
repeat			

8-bit two component CbCr - HVF Mode

msb(7)	Cb_n	lsb(0)	[3:0] 0x00
msb(7)	Cr_n	lsb(0)	[3:0] 0x00
msb(7)	Cb_{n+1}		[3:0] 0x00
msb(7)	Cr_{n+1}	lsb(0)	[3:0] 0x00
repeat			

8-bit single component - Raw Mode

msb(7)	S_n	lsb(0)	[3:0] 0x00
repeat			

FIGURE 16. 8-bit input formats on VDATA bus

8.4.5 Input formats on VDATA bus [10-bit data] controlled by PFMT and PREC register settings

10-bit three component - CbYCrY - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(9)	Cb_n	lsb(0)	[1:0] 0x00
msb(9)	Y_n	lsb(0)	[1:0] 0x00
msb(9)	Cr_n	lsb(0)	[1:0] 0x00
msb(9)	Y_{n+1}	lsb(0)	[1:0] 0x00
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

10-bit single component - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(9)	S_n	lsb(0)	[1:0] 0x00
msb(9)	S_{n+1}	lsb(0)	[1:0] 0x00
msb(9)	S_{n+2}		[1:0] 0x00
msb(9)	S_{n+3}	lsb(0)	[1:0] 0x00
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

10-bit two component CbCr - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(9)	Cb_n	lsb(0)	[1:0] 0x00
msb(9)	Cr_n	lsb(0)	[1:0] 0x00
msb(9)	Cb_{n+1}		[1:0] 0x00
msb(9)	Cr_{n+1}	lsb(0)	1:0] 0x00
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

10-bit three component CbYCrY- HVF Mode

msb(9)	Cb_n	lsb(0)	[1:0] 0x00
msb(9)	Y_n	lsb(0)	[1:0] 0x00
msb(9)	Cr_n	lsb(0)	[1:0] 0x00
msb(9)	Y_{n+1}	lsb(0)	[1:0] 0x00
repeat			

10-bit two component CbCr - HVF Mode

msb(9)	Cb_n	lsb(0)	[1:0] 0x00
msb(9)	Cr_n	lsb(0)	[1:0] 0x00
msb(9)	Cb_{n+1}		[1:0] 0x00
msb(9)	Cr_{n+1}	lsb(0)	[1:0] 0x00
repeat			

10-bit single component - HVF Mode

msb(9)	S_n	lsb(0)	[1:0] 0x00
repeat			

10-bit single component - Raw Mode

msb(9)	S_n	lsb(0)	[1:0] 0x00
repeat			

FIGURE 17. 10-bit input formats on VDATA bus

8.4.6 Input formats on VDATA bus [12/16-bit data] controlled by PFMT and PREC register settings

12-bit three component - CbYCrY - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(11)	Cb_n		lsb(0)
msb(11)	Y_n		lsb(0)
msb(11)	Cr_n		lsb(0)
msb(11)	Y_{n+1}		lsb(0)
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

12-bit single component - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(11)	S_n		lsb(0)
msb(11)	S_{n+1}		lsb(0)
msb(11)	S_{n+2}		lsb(0)
msb(11)	S_{n+3}		lsb(0)
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

12-bit two component CbCr - EAV/SAV Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(11)	Cb_n		lsb(0)
msb(11)	Cr_n		lsb(0)
msb(11)	Cb_{n+1}		lsb(0)
msb(11)	Cr_{n+1}		lsb(0)
repeat			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00

12-bit three component CbYCrY- HVF Mode

msb(11)	Cb_n	lsb(0)
msb(11)	Y_n	lsb(0)
msb(11)	Cr_n	lsb(0)
msb(11)	Y_{n+1}	lsb(0)
repeat		

12-bit two component CbCr - HVF Mode

msb(11)	Cb_n	lsb(0)
msb(11)	Cr_n	lsb(0)
msb(11)	Cb_{n+1}	lsb(0)
msb(11)	Cr_{n+1}	lsb(0)
repeat		

12-bit single component - HVF Mode

msb(11)	S_n	lsb(0)
repeat		

12-bit single component - Raw Mode

msb(11)	S_n	lsb(0)
repeat		

16-bit single component - HVF Mode
set PFMT to 0x4h, PREC to 0x04h

msb(15)	S_n	lsb(0)
repeat		

16-bit single component - Raw Mode
set PFMT to 0x0Ch, PREC to 0x04h

msb(15)	S_n	lsb(0)
repeat		

FIGURE 18. 12/16-bit input formats on VDATA bus

8.4.7 Input formats on VDATA bus [extended mode] controlled by PFMT and PREC register settings

8-bit three component CbYCrY - Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(7)	Cb_n	lsb(0)	[3:0] 0x00	msb(7)	Y_n	lsb(0)	[3:0] 0x00
msb(7)	Cr_n	lsb(0)	[3:0] 0x00	msb(7)	Y_{n+1}	lsb(0)	[3:0] 0x00
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

8-bit two component CbCr - Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(7)	Cb_n	lsb(0)	[3:0] 0x00	msb(7)	Cr_n	lsb(0)	[3:0] 0x00
msb(7)	Cb_{n+1}		[3:0] 0x00	msb(7)	Cr_{n+1}	lsb(0)	[3:0] 0x00
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

8-bit single component - Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(7)	S_n	lsb(0)	[3:0] 0x00	msb(7)	S_{n+1}		[3:0] 0x00
msb(7)	S_{n+2}	lsb(0)	[3:0] 0x00	msb(7)	S_{n+3}	lsb(0)	[3:0] 0x00
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

FIGURE 19. 8-bit extended input formats on VDATA bus

8.4.8 Input formats on VDATA bus [extended mode] controlled by PFMT and PREC register settings

10-bit three component CbYCrY- Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(9)	Cb_n	lsb(0)	[1:0] 0x00	msb(9)	Y_n	lsb(0)	[1:0] 0x00
msb(9)	Cr_n	lsb(0)	[1:0] 0x00	msb(9)	Y_{n+1}	lsb(0)	[1:0] 0x00
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

10-bit two component CbCr- Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(9)	Cb_n	lsb(0)	[1:0] 0x00	msb(9)	Cr_n	lsb(0)	[1:0] 0x00
msb(9)	Cb_{n+1}		[1:0] 0x00	msb(9)	Cr_{n+1}	lsb(0)	[1:0] 0x00
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

10-bit single component - Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(9)	S_n	lsb(0)	[1:0] 0x00	msb(9)	S_{n+1}	lsb(0)	[1:0] 0x00
msb(9)	S_{n+2}	lsb(0)	[1:0] 0x00	msb(9)	S_{n+3}	lsb(0)	[1:0] 0x00
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

FIGURE 20. 10-bit extended input formats on VDATA bus

8.4.9 Input formats on VDATA bus [extended mode] controlled by PFMT and PREC register settings

12-bit three component CbYCrY - Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00				
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(11)	Cb_n		lsb(0)	msb(11)	Y_n		lsb(0)
msb(11)	Cr_n		lsb(0)	msb(11)	Y_{n+1}		lsb(0)
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

12-bit two component CbCr - Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00				
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(11)	Cb_n		lsb(0)	msb(11)	Cr_n		lsb(0)
msb(11)	Cb_{n+1}		lsb(0)	msb(11)	Cr_{n+1}		lsb(0)
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

12-bit one component- Extended Mode

msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00				
msb(7)	***SAV***	lsb(0)	[3:0] 0x00	msb(7)	***SAV***	lsb(0)	[3:0] 0x00
msb(11)	S_n		lsb(0)	msb(11)	S_{n+1}		lsb(0)
msb(11)	S_{n+2}		lsb(0)	msb(11)	S_{n+3}		lsb(0)
repeat				*repeat*			
msb(7)	***0xFF***	lsb(0)	[3:0] 0x00	msb(7)	***0xFF***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***0x00***	lsb(0)	[3:0] 0x00	msb(7)	***0x00***	lsb(0)	[3:0] 0x00
msb(7)	***EAV***	lsb(0)	[3:0] 0x00	msb(7)	***EAV***	lsb(0)	[3:0] 0x00

FIGURE 21. 12-bit extended input formats on VDATA bus

8.4.10 Input Pin Configurations for Pixel/Video input on HDATA or VDATA bus

Table 4: Input pin configuration on VDATA bus

DATA FORMAT	FORMAT	VMODE	PIN CONFIGURATION [MSB]...LSB]	PIN CONFIGURATION [MSB]...[LSB]	I/P FORMAT
Video	**8-bit**	**EAV/SAV**		**VDATA[11]...[4]**	
	10-bit	**EAV/SAV**		**VDATA[11]...[2]**	
	12-bit	**HVF**		**VDATA[11]...[0]**	
	16-bit	**Extended**	**VDATA[23]...[16]**	**VDATA[11]...[4]**	
			Sn,	Sn+1	1 component
			Cbn, Crn...	Yn, Yn+1...	3 component
			Cbn, Cbn+1...	Crn, Crn+1...	2 component
	20-bit	**Extended**	**VDATA[23]...[14]**	**VDATA[11]...[2]**	
			Sn	Sn+1	1 component
			Cbn, Crn...	Yn, Yn+1...	3 component
			Cbn, Cbn+1...	Crn, Crn+1...	2 component
	24-bit	**Extended**	**VDATA[23]...[12]**	**VDATA[11]...[0]**	
			Sn, Sn+2...	Sn+1, Sn+3...	1 component
			Cbn, Crn...	Yn, Yn+1...	3 component
			Cbn, Cbn+1...	Crn, Crn+1...	2 component
Pixel	**8-bit**	**Raw**		**VDATA[15]...[8]**	
	10-bit	**Raw**		**VDATA[15]...[6]**	
	12-bit	**Raw**		**VDATA[15]...[4]**	
	16-bit	**Raw**		**VDATA[15]...[0]**	

Table 5: Input pin configuration on HDATA bus - 8-bit data

DATA FOR -MAT	FORMAT	PIN CONFIG. [MSB]...[LSB]	PIN CONFIG. [MSB]...[LSB]	PIN CONFIG. [MSB]...[LSB]	PIN CONFIG. [MSB...[LSB]	
Pixel	**8-bit**	**HDATA [31]...[24]**	**HDATA [23]...[16]**	**HDATA [15]...[8]**	**HDATA [7]...[0]**	
		Sn	Sn+1.	Sn+2	Sn+3	1 component
		Cbn	Crn	Cbn+1	Crn+1	2 component
		Yn, Yn+2...	Yn+1, Yn+3.	Cbn, Cbn+1.	Crn, Crn+1...	3 component
		Yn, Yn+2...	Cbn, Cbn+1.	Yn+1, Yn+3.	Crn, Crn+1...	3 component

Table 6: Input pin configuration on HDATA bus - 10, 12, 14, 16-bit data

DATA FOR -MAT	FORMAT	PIN CONFIG. [MSB]...[LSB]	PIN CONFIG. [MSB]...[LSB]	
Pixel	**10-bit**	**HDATA [31]...[22]**	**HDATA [15]...[6]**	
		Sn	Sn+1	1 component
		Cbn, Cbn+1	Cr, Crn+1	2 component
		Yn, Yn+1...	Cbn, Crn	3 component
		Yn, Cbn	Yn+1, Crn	3 component
Pixel	**12-bit**	**HDATA [31]...[20]**	**HDATA [15]...[4]**	
		Sn	Sn+1	1 component
		Cbn, Cbn+1	Cr, Crn+1	2 component
		Yn, Yn+1...	Cbn, Crn	3 component
		Yn, Cbn	Yn+1, Crn	3 component
Pixel	**14-bit**	**HDATA [31]...[18]**	**HDATA [15]...[2]**	
		Sn	Sn+1	1 component
		Cbn, Cbn+1	Cr, Crn+1	2 component
		Yn, Yn+1...	Cbn, Crn	3 component
		Yn, Cbn	Yn+1, Crn	3 component
Pixel	**16-bit**	**HDATA [31]...[16]**	**HDATA [15]...[0]**	
		Sn	Sn+1	1 component
		Cbn, Cbn+1	Cr, Crn+1	2 component
		Yn, Yn+1...	Cbn, Crn	3 component
		Yn, Cbn	Yn+1, Crn	3 component

8.5. EXTERNAL DMA ACCESS

0xFFFF1408 EDMOD0 External DMA mode register 0 R/W

This register is used to configure external DMA channel 0 as to what data is assigned to channel 0 and which access pins are to be used.

Bits	Name	Description	Reset Value
0	DMEN0	Enable external DMA channel 0 0 = disabled 1 = enabled	0
2:1	DMSEL0	DMA Channel 0 assignment 0 Pixel data 1 Compressed data/ Code-block data 2 Attribute data 3 Ancillary data.	0
5:3	DMMOD0	DMA Channel 0 mode 0 Dedicated Chip Select DMA mode 1 Single transfer DREQ/DACK DMA mode 2 Burst transfer DREQ/DACK DMA mode 3 JDATA mode 4 *reserved* 5 Single transfer Fly-by DMA mode 6 Burst transfer Fly-by DMA mode 7 *reserved*	0
8:6	DMBL0	DMA Channel 0 burst length [in number of accesses] 0 8 not available for DWIDTH of 8-bit 1 16 for DWIDTH of 8/16 or 32-bits 2 32 for DWIDTH of 8/16 or 32-bits 3 64 for DWIDTH of 8/16 or 32-bits 4 128 for DWIDTH of 8/16 or 32-bits 5 256 for DWIDTH of 8/16 or 32-bits 6 512 for DWIDTH of 8/16-bit, for 32-bits not recommended 7 1024 for DWIDTH of 8-bit, for 16-bit not recommended, for 32-bits not available.	0
9	DR0POL	$\overline{\text{DREQ0}}$ / $\overline{\text{FSRQ0}}$ / $\overline{\text{VALID}}$ polarity; 0=active low, 1=active high	0
10	DA0POL	$\overline{\text{DACK0}}$ / $\overline{\text{FCS0}}$ / $\overline{\text{HOLD}}$ polarity; 0=active low, 1=active high	0
14:11	DR0PULS	DREQ0 pulse width [in Jclkcycles]. If DR0PULS==0, then DREQ0 will remain asserted until DACK0 and RD/ in read mode or WR/ in write mode are asserted.	1
15	Reserved	*Reserved for future use. Always write 0*.	0

0xFFFF140C EDMOD1 External DMA mode register 1 R/W

This register is used to configure external DMA channel 1.

Bits	Name	Description	Reset Value
0	DMEN1	Enable external DMA channel 1 0 = disabled 1 = enabled	0
2:1	DMSEL1	DMA Channel 1 assignment 0 Pixel data 1 Compressed data/ Code-block data 2 Attribute data 3 Ancillary data.	1
5:3	DMMOD1	DMA Channel 1 mode 0 Dedicated Chip Select DMA mode. 1 Single transfer DREQ/DACK DMA mode 2 Burst transfer DREQ/DACK DMA mode 3 *reserved* 4 *reserved* 5 Single transfer Fly-by DMA mode 6 Burst transfer Fly-by DMA mode 7 *reserved*	0
7:6	DMBL1	DMA Channel 1 burst length [in number of accesses]: 0 8 not available for DWIDTH of 8-bit 1 16 for DWIDTH of 8/16 or 32-bits 2 32 for DWIDTH of 8/16 or 32-bits 3 64 for DWIDTH of 8/16 or 32-bits 4 128 for DWIDTH of 8/16 or 32-bits 5 256 for DWIDTH of 8/16 or 32-bits 6 512 for DWIDTH of 8/16-bit, for 32-bits not recommended 7 1024 for DWIDTH of 8-bit, for 16-bit not recommended, for 32-bits not available.	0
9	DR1POL	$\overline{\text{DREQ1}}$ / $\overline{\text{FSRQ1}}$ polarity; 0=active low, 1=active high	0
10	DA1POL	$\overline{\text{DACK1}}$ / $\overline{\text{FCS1}}$ polarity; 0=active low, 1=active high	0
14:11	DR1PULS	DREQ1 pulse width [in Jclkcycles]. If DR1PULS==0, then DREQ1 will remain asserted until DACK1 and RD/ in read mode or WR/ in write mode are asserted.	1
15	Reserved	*Reserved for future use. Always write 0*.	0

8.5.1 EDMOD REGISTERS

EDMOD0 and EDMOD1 are used to configure the ADV202 for external DMA operation. DMA allows transfers from/to memory without the on-board processor being required to perform the transactions as it is the case in Normal Host mode.

The external DMA registers include settings for Single/Burst DMA, Fly-By Mode DMA, and Dedicated Chip Select (DCS) DMA. These registers are used in conjunction with the Bus Mode (BUSMODE) and

FIFO Mode (FFMODE) registers.

8.5.2 EXTERNAL DMA DATA WIDTH

The desired data width when accessing data FIFOs is set in the BUSMODE register. The DWIDTH field in this register (direct address 0x8) can be programmed to byte, half-word (16-bit), or word (32-bit) data widths. The default data-width is half-word (16-bit).

8.5.3 FIFO MODE CONFIGURATION

The FIFO mode configuration register (FFMODE) must be programmed before using External DMA. This register sets the FIFO depths and data direction for the Code, Attribute, and Ancillary FIFOs. The FFCFG field is used to configure the FIFO depths. The CINP/AINP/NINP bits are used to set the Code/Attribute/Ancillary FIFOs to data input. The register encodings are listed in the FFMODE description under the CONFIGURABLE FIFO BLOCK section. Writing to the FFMODE register automatically resets the CODE, ATTR, ANC FIFOs.

For details on the Pixel FIFO refer to the CONFIGURABLE PARAMETERS FOR THE PIXEL INTERFACE section.The depth of the Pixel FIFO is fixed at 256 words.

8.5.4 DMA MODES

8.5.4.1 DREQ/DACK DMA Mode and FLY-BY DMA Mode

These registers are used to enable the External DMA interface, assign the desired FIFO toeach channel, and select modes of operation. All EDMA modes make use of the Data Request (DREQ0/DREQ1) and Data Acknowledge (DACK0/DACK1) pins.
DREQ/DACK DMA mode and Fly-By DMA mode can be used in single transfer or burst transfer modes.

8.5.4.2 Dedicated Chip Select DMA Mode

This mode is available for devices which monitor the $\overline{\text{FSRQx}}$ ($\overline{\text{DREQx}}$) pins statically (as opposed to pulse detection). The $\overline{\text{FSRQx}}$ ($\overline{\text{DREQx}}$) pins will be asserted when data/space is available in the selected FIFO and will be de-asserted when the FIFO is empty/full. The $\overline{\text{FCSx}}$ ($\overline{\text{DACKx}}$) pins are used as FIFO Chip Select signals and should be asserted in conjunction with the $\overline{\text{RD}}/\overline{\text{WE}}$ pins.
This mode is enabled via the EDMODE0/EDMOD1 registers.
The FSRQx/ polarity can be programmed to be active low or active high in the EDMOD0 register.[1]

8.5.5 JDATA JDATA MODE

This mode allows input/output of compressed data using the VALID/HOLD protocol and the JDATA[7:0] pins. Please note that 32-bit data/host modes and the Extended Pixel Interface are not available when JDATA mode is enabled.
To assign HDATA [31:24] pins to JDATA[7:0], the BUSMODE register must first be programmed to JDATA. The polarity of the VALID (DREQ0 pin) and HOLD (DACK0 pin) is set in the EDMOD0 register through bits DR0POL/DA0POL. The DMSEL0 field in EDMOD0 must be set to “1” (Compressed Data). The DMMOD0 field must be set to “3” (JDATA Mode). It is recommended that the user set up all mode bits in EDMOD0 without enabling the interface (DMEN0). Then, the HOLD pin should be asserted (depending on the polarity selected), and then de asserted after the JDATA interface has been enabled by either the Host or the ADV202. Data transfers occur when the ADV202 asserts VALID and the external device de-asserts HOLD.

The VALID signal will always function as an output from the ADV202 and the HOLD signal will always function as an input to the ADV202. Data transfers occur when VALID is asserted and HOLD is de-asserted at the rising edge of MCLK.

JDATA is referenced with MCLK and requires a PLL_MULT setting of >= 4.

8.5.6. SINGLE TRANSFER/ BURST TRANSFER AND BURST LENGTH

Single Transfer DREQ/DACK DMA
In this mode, the ADV202 will assert DREQx when there is at least one data transfer available for the programmed FIFO. If the FIFO is set to output, the DREQx assertion indicates that output data is present in the FIFO. When set to input, this indicates that space is available in the FIFO for at least one data transfer. The external device must respond with DACKx, in conjunction with RD/WE/HDATA activity.
In Single Transfer Fly-By Mode, the functionality of the RD and WE pins is reversed.

Burst Transfer DREQ/DACK DMA
This mode is similar to Single Transfer EDMA, but each DREQx assertion indicates that the programmed number of accesses of the width selected are available for transfer. The number of accesses per burst is set in the Burst Length (DMBL0/DMBL1) fields in the EDMOD0/EDMOD1 registers. The access data width is set by the BUSMODE/DWIDTH field, as previously described.
In Burst Transfer Fly-By Mode, the functionality of the RD and WE pins is reversed.

Burst Length
The Burst Length settings must be programmed in conjunction with the FIFO depths setting in the FFMODE/FFCFG field. Note that the FIFO depths refer to 32-bit words, but the Burst Lengths correspond to the number of accesses (which may be 8/16/32 bits wide). A given FIFO depth can support the same number of word transfers, twice the number of half-word transfers, and four times the number of byte transfers.

For example, if the Attribute FIFO is set to a depth of 64, Burst Length (EDMODx/DMBLx) can be programmed to up to 64 word transfers, 128 half-word transfers, or 256 byte transfers. For highest throughput, it is recommended that the Burst Length is set to a value which is less than the FIFO depth.
This will allow the ADV202 to transfer data internally while the external device simultaneously accesses thesame FIFO through the EDMA interface.

If the CODE FIFO does not contain the programmed number of accesses [i.e. words] for the last portion of the compressed field, the FIFO is packed with 0s to ensure that the field ends on a burst boundary.

8.6. CONFIGURABLE FIFO BLOCK

For details on how to use FIFO access refer to the External DMA section in this data sheet.

0xFFFF1418 FFMODE Configurable FIFO mode register R/W

This register is used to configure the FIFO for the CODE, ATTR and ANCIL data channels. Writing to this register will reset all data FIFOs. The PIXEL FIFO has always a size of 256 words.

Bits	Name	Description	Reset Value
3:0	FFCFG	Enables and sets the sizes of the FIFOs associated with the CODE, ATTR and ANCL data channels. The maximum data width of these FI FOs is 32-bits. FFCFG CODE ATTR ANCL 0 256 256 N/A 1 256 128 128 2 384 128 N/A 3 384 64 64 4 512 N/A N/A 15-5 Reserved	4
4	CINP	Sets the CODE data FIFO as an input channel.	0
5	AINP	Sets the ATTR data FIFO as an input channel.	0
6	NINP	Sets the ANCL data FIFO as an input channel.	0
15:7	*reserved*	Reserved for future use, always write 0	undef

0xFFFF1410 FFTHRP FIFO Threshold for PIXEL FIFO R/W

Sets the threshold detect level for the PIXEL data channel. This value is compared with the empty/full count of the FIFO and is used to set interrupt conditions. If the ADV202 is used in Dedicated Chip Select mode the setting of this register will set the interrupt conditions and will drive the FSRQx/ signal.[1]
If the FIFO is in input mode then the threshold condition will be true whenever the empty count is greater than, or equal to, the threshold value. If the
FIFO is in output mode then the threshold condition will be true whenever the full count is greater than, or equal to, the threshold value. The threshold count is in units of 32-bit words.

Bits	Name	Description	Reset Value
15:0	THRP	Set the threshold level for PIXEL data FIFO.	0

0xFFFF141C FFTHRC FIFO Threshold for CODE FIFO R/W

Sets the threshold detect level for the CODE data channel. This value is compared with the empty/full count of the FIFO and is used to set interrupt conditions. If the ADV202 is used in Dedicated Chip Select mode the setting of this register will set the interrupt conditions and will drive the FSRQx/ signal.[1]

If the FIFO is in input mode then the threshold condition will be true whenever the empty count is greater than, or equal to, the threshold value. If the FIFO is in output mode then the threshold condition will be true whenever the full count is greater than, or equal to, the threshold value. The threshold count is in units of

32-bit words. Writing to this register will reset all data FIFOs.

Bits	Name	Description	Reset Value
15:0	THRC	Set the threshold level for CODE data FIFO.	0

0xFFFF1420 FFTHRA FIFO Threshold for ATTR FIFO R/W

Sets the threshold detect level for the ATTR data channel. This value is compared with the empty/full count of the FIFO and is used to set interrupt conditions. If the ADV202 is used in Dedicated Chip Select mode the setting of this register will set the interrupt conditions and will drive the FSRQx/ signal.[1]
If the FIFO is in input mode then the threshold condition will be true whenever the empty count is greater than, or equal to, the threshold value. If the FIFO is in output mode then the threshold condition will be true whenever the full count is greater than, or equal to, the threshold value.The threshold count is in units of 32-bit words. Writing to this register will reset all data FIFOs.

Bits	Name	Description	Reset Value
15:0	THRA	Set the threshold level for ATTR data FIFO.	0

0xFFFF1424 FFTHRN FIFO Threshold for ANCL FIFO R/W

Sets the threshold detect level for the ANCL data channel. This value is compared with the empty/full count of the FIFO and is used to set interrupt conditions. If the ADV202 is used in Dedicated Chip Select mode the setting of this register will set the interrupt conditions and will drive the FSRQx/ signal.[1]
If the FIFO is in input mode then the threshold condition will be true whenever the empty count is greater than, or equal to, the threshold value. If the FIFO is in output mode then the threshold condition will be true whenever the full count is greater than, or equal to, the threshold value.The threshold count is in units of 32-bit words. Writing to this register will reset all data FIFOs.

Bits	Name	Description	Reset Value
15:0	THRN	Set the threshold level for ANCL data FIFO	0

0xFFFF1414 FFCNTP FIFO Full/Empty count for PIXEL FIFO RO

Returns the full/empty count of the PIXEL FIFO. If the FIFO is in input mode then this register will return the number of empty locations in the FIFO. If the FIFO is in output mode then this register will return the number of locations in the FIFO that contain valid data.The threshold count is in units of 32-bit words

Bits	Name	Description	Reset Value
8:0	CNTP	Full/Empty count for the PIXEL FIFO. **Reset value depends on mode	**
15:9	*reserved*	Reserved for future use.	undef

0xFFFF1428 FFCNTC FIFO Full/Empty count for CODE FIFO RO

Returns the full/empty count of the CODE FIFO. If the FIFO is in input mode then this register will return the number of empty locations in the FIFO. If the FIFO is in output mode then this register will return the number of locations in the FIFO that contain valid data.The threshold count is in units of 32-bit words

Bits	Name	Description	Reset Value
9:0	CNTC	Full/Empty count for the CODE FIFO. **Reset value depends on mode	**
15:10	*reserved*	Reserved for future use.	undef

0xFFFF142C FFCNTA FIFO Full/Empty count for ATTR FIFO RO

Returns the full/empty count of the ATTR FIFO. If the FIFO is in input mode then this register will return the number of empty locations in the FIFO. If the FIFO is in output mode then this register will return the number of locations in the FIFO that contain valid data.The threshold count is in units of 32-bit words

Bits	Name	Description	Reset Value
8:0	CNTA	Full/Empty count for the ATTR FIFO. **Reset value depends on mode	**
15:9	*reserved*	Reserved for future use,	undef

0xFFFF1430 FFCNTN FIFO Full/Empty count for ANCL FIFO RO

Returns the empty/full count of the ANCL FIFO. If the FIFO is in input mode then this register will return the number of empty locations in the FIFO. If the FIFO is in output mode then this register will return the number of locations in the FIFO that contain valid data.The threshold count is in units of 32-bit words

Bits	Name	Description	Reset Value
7:0	CNTN	Full/Empty count for the ANCL FIFO. **Reset value depends on mode	**
15:8	*reserved*	Reserved for future use,	undef

9. START-UP CONFIGURATION

The “Getting Started...” Application Note contains detailed instructions how to program the ADV202 for specific applications, such as encode or decode for NTSC,1080i or custom specific format, and to run certain functionality tests.

The following section contains a short overview only.
Before configuring the ADV202 for specific applications it is recommended to verify the functionality of the ADV202 design in performing some basic functionality tests.

9.1. Verify Direct Register access functionality

Direct Register access is achieved with Normal Host accesses, interfacing the ADV202 over:
- ADDR<3:0>
- CS/
- WE/
- RD/
- ACK/
- HDATA <31:0> or <15:0>

All direct registers should be accessible regardless of which boot mode is use [except boot modes 0 and 1].
A hard reset [BOOT mode register] over several cycles is required before Direct Register access.
A hard reset is applied in holding the RESET/ pin low.

Until the ADV202 asserts ACK/, the host must hold the state of the ADDR<3:0>, CS/, WE/, RD/ and HDATA pins. Direct register access is not completed until after ACK/ is asserted.
This only applies to Normal Host access read operations and is not required for any DMA modes.

9.2. Indirect Register access functionality

The general procedure is to:

1. Configure the PLL registers PLL_HI and PLL_LO and wait for the PLL to lock.
2. Initiate a re-boot [No-Boot-Host mode] by writing to the BOOT mode register.
3. Configure the BUSMODE register to configure the HDATA bus.
4. Configure the MMODE register for indirect register access.
5. For 16-bit hosts, configure the STAGE register.
6. Set start address for the data values in writing to IADDR.
7. Load test data to memory in writing the IDATA.
8. Read test values back from their indirect register location.

9.3. Load and run a specific application for 32-bit host application

1. Configure the PLL registers PLL_HI and PLL_LO and wait for the PLL to lock.
2. Initiate a re-boot [No-Boot-Host mode] in writing to the BOOT mode register.
3. Configure the BUSMODE register to configure the HDATA bus.
4. Configure the MMODE register for indirect register access.
5. For 16-bit hosts, configure the STAGE register.
6. Set start address for the program in writing to IADDR.

7. Load program into memory in writing the program data to IDATA.
8. Initiate a re-boot [Co-Processor mode] to start the program.
9. Configure the BUSMODE register to configure the HDATA bus again.
10. Configure the MMODE register for indirect register access.
11. Load program specific parameters in writing to IADDR and IDATA.
12. Unmask the SWIRQ0 bit in the EIRQIE register and wait for the ADV202 to assert an interrupt.
13. Configure DMA channels 0 and 1 in the EDMOD0 and EDMOD1 registers using indirect register access.
14. Clear the SWIRQ0 bit in the EIRQFLG register and start the data flow process.

10. ESF and ADV202 firmware

The ADV202 is a System-on-Chip (SOC) implementation, which means that it incorporates dedicated hardware functions, a processor and on-board firmware/software. It also means that configuring and managing the operations of the chip can be handled by the on-chip processor and its software. Other than basic bus and I/O configuration which the user has to set up [see previous chapter], most of the configuration and control of the ADV202 is handled by the firmware. The firmware is responsible of setting other registers in different functional blocks of the ADV202, scheduling events etc. The user has no access to these functional blocks, they are entirely controlled by the firmware or ESF.

There are two methods to operate the ADV202:

1. Load specific firmware to the ADV202 as described in the previous chapter.

2. Embedded Software Framework.

The Embedded Software Framework resides in the on-chip ROM and contains all necessary support for all functions which might be performed by the software interface. It is controlled with a high-level command protocol, which simplifies processing. It supports plug-in updates loading which allows to keep up with up-to-date software features without the requirement to replace the ADV202 with a newer ROM-version ADV202. ESF will be available on the released version of the ADV202.

11. APPLICATIONS

11.1. Encode/decode in multi-chip applications

Due to the data input rate limitation [ref. to tables 1 and 2] an 1080i application will require at least 2 ADV202s to encode or decode full resolution 1080i video.
In encode mode the ADV202 accepts Y and CbCr data on separate buses. The input data must be in EAV/ SAV format.
In decode mode a master/slave configuration, as shown in the figure below, or a slave/slave configuration can be used in order to synchronize the outputs of the two ADV202s. Refer to the technical note “ADV202 multi-chip application” for more detail on how to configure the ADV202s in a multi-chip application.
This data sheet contains a brief outline of this procedure.
Applications that have the two separate VDATA outputs send to an FPGA or buffer before they are send to an encoder, for example, do not require synchronization at the ADV202 outputs.

Encode mode

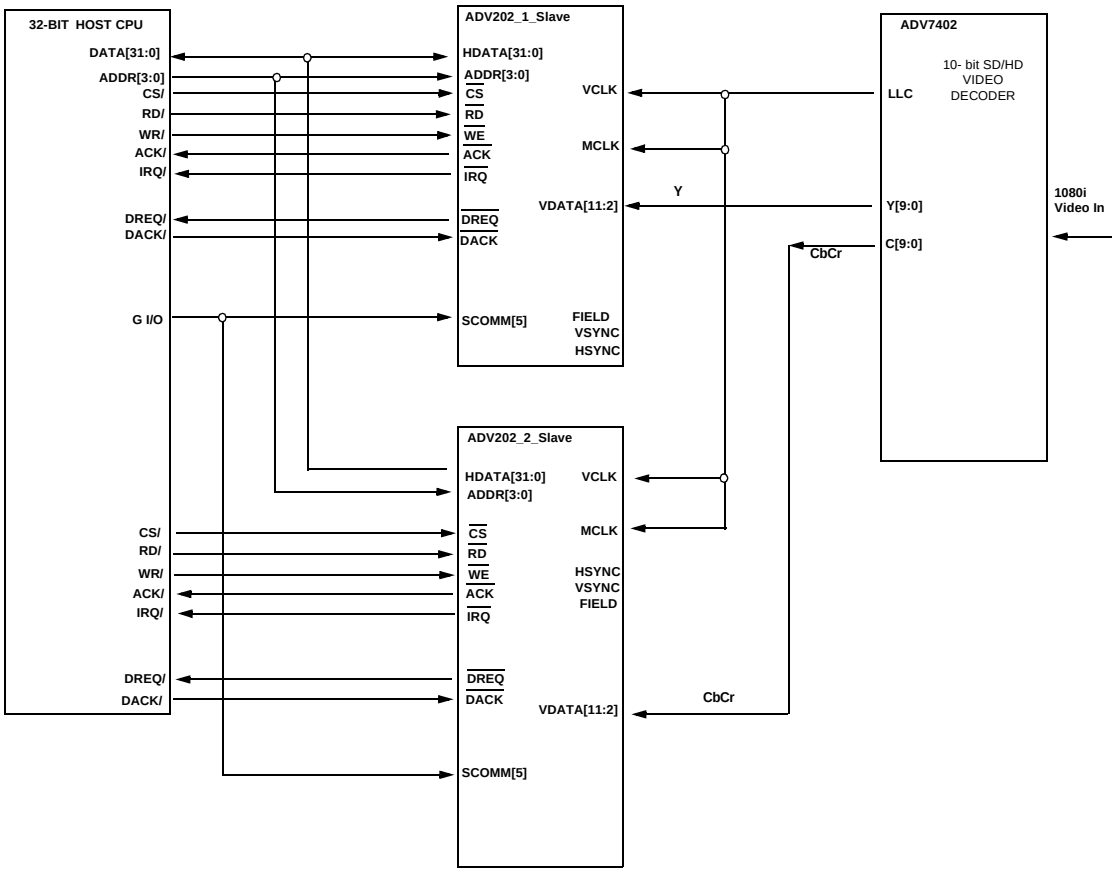


Figure 22. ADV202 multi-chip application - Encode

In a master/slave configuration it is expected that the master HVF outputs are connected to the slave HVF inputs and that each SCOMM[5] pin is connected to the same GPIO on the host.
In a slave /slave configuration the common HVF for both ADV202s is generated by an external house sync

Decode Master/Slave configuration

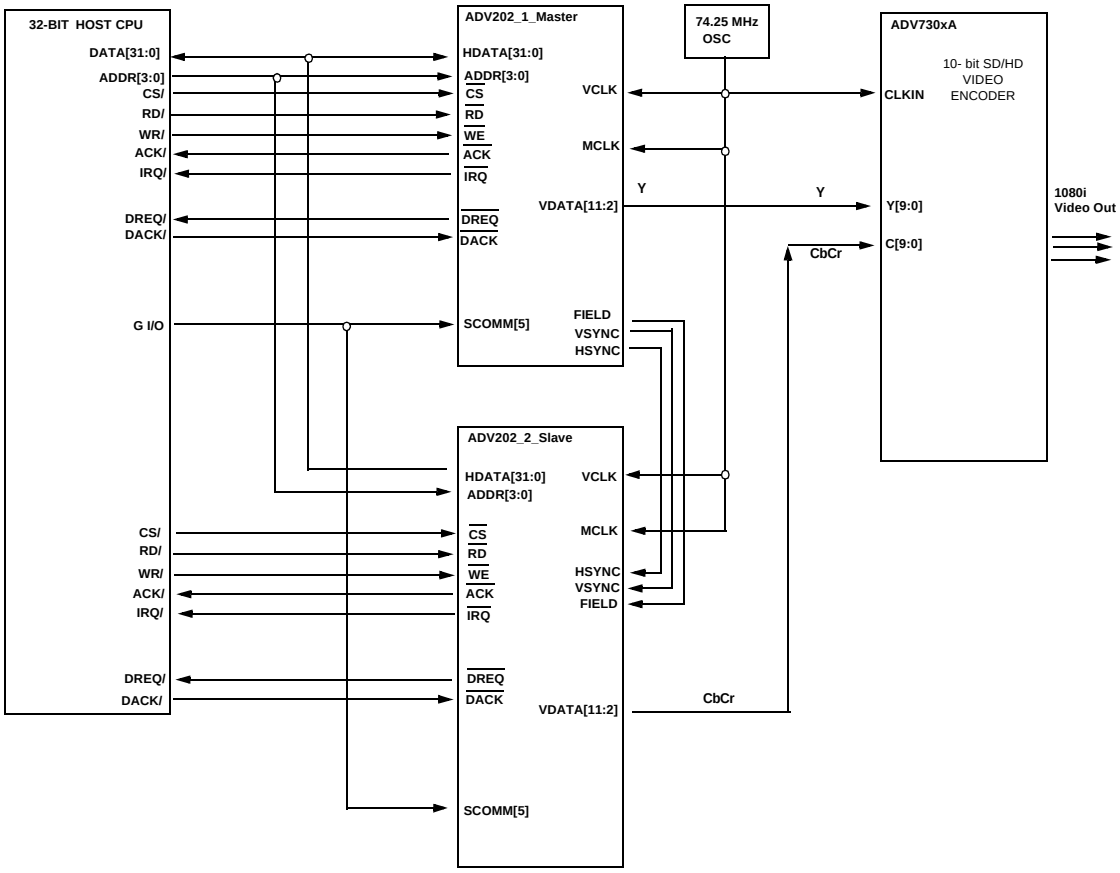


Figure 23. ADV202 multi-chip application in a decode master/slave configuration

and each SCOMM[5] is connected to the same GPIO output on the host.

SWIRQ1, Software Interrupt 1 in the EIRQIE register must be unmasked on both devices to enable multi-chip mode.

In a master/slave configuration the following must be considered also:
Every slave ADV202 has a fixed timing delay form HSYNC/ active input to video data out. The necessary register to compensate for this delay is set via the firmware on the master ADV202. The value of this register is programmed into the 0xFFFF0440 register of the master device by the firmware when the part is configured in multi-sync mode [i.e. when SWIRQ1 is enabled].

11. 2. Digital Still Camera and Camcorder Applications

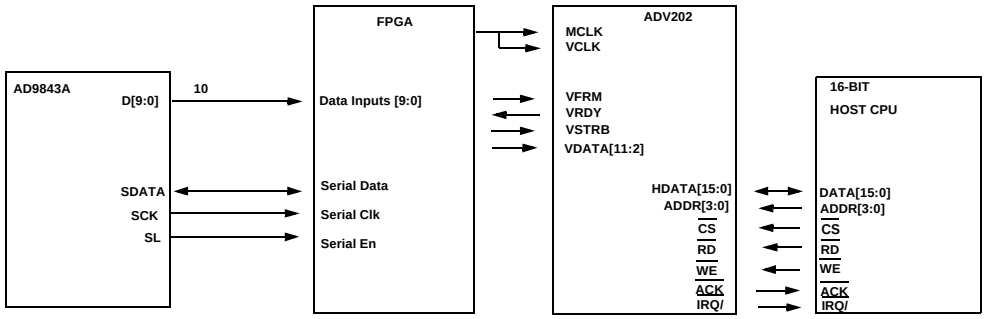


Figure 24. ADV202 with pixel input for Digital Still Camera and Camcorder Applications

11.3 Encode/decode SDTV video

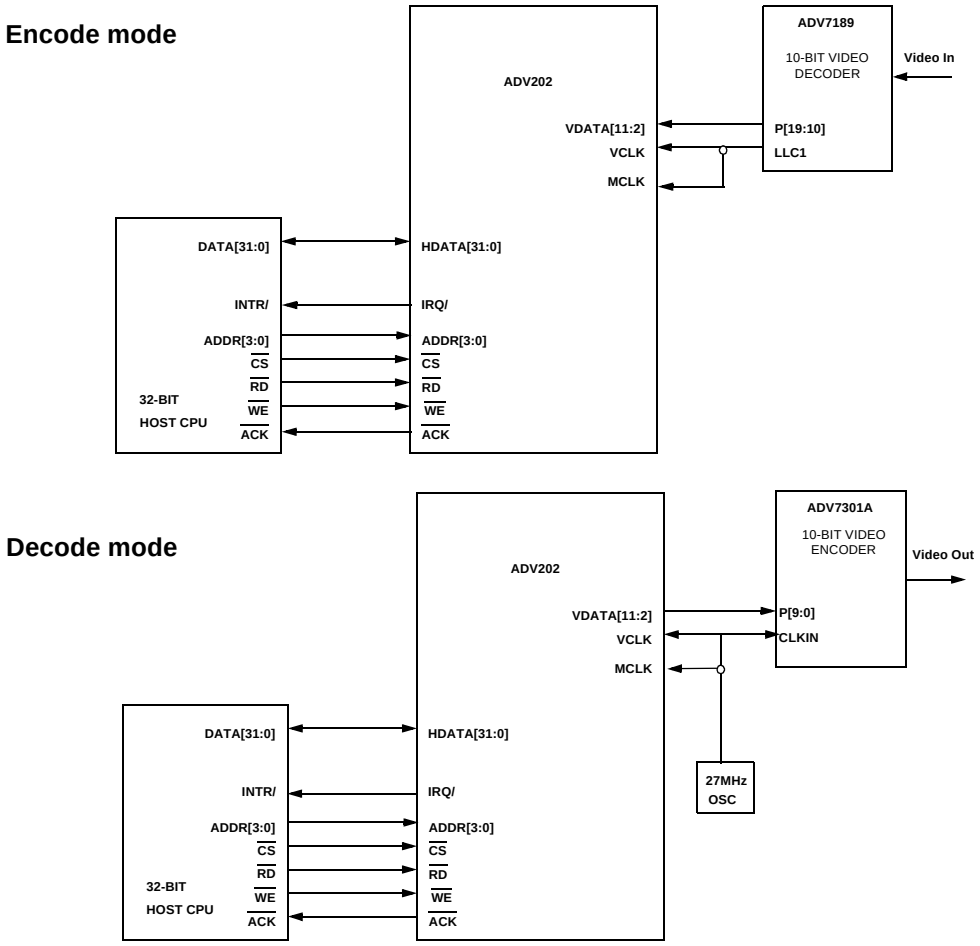


Figure 25. ADV202 with 32-bit host interfaced and 10-bit CCIR656 video application [Normal host mode]

11.4 32-bit Host/ 32-bit ASIC application

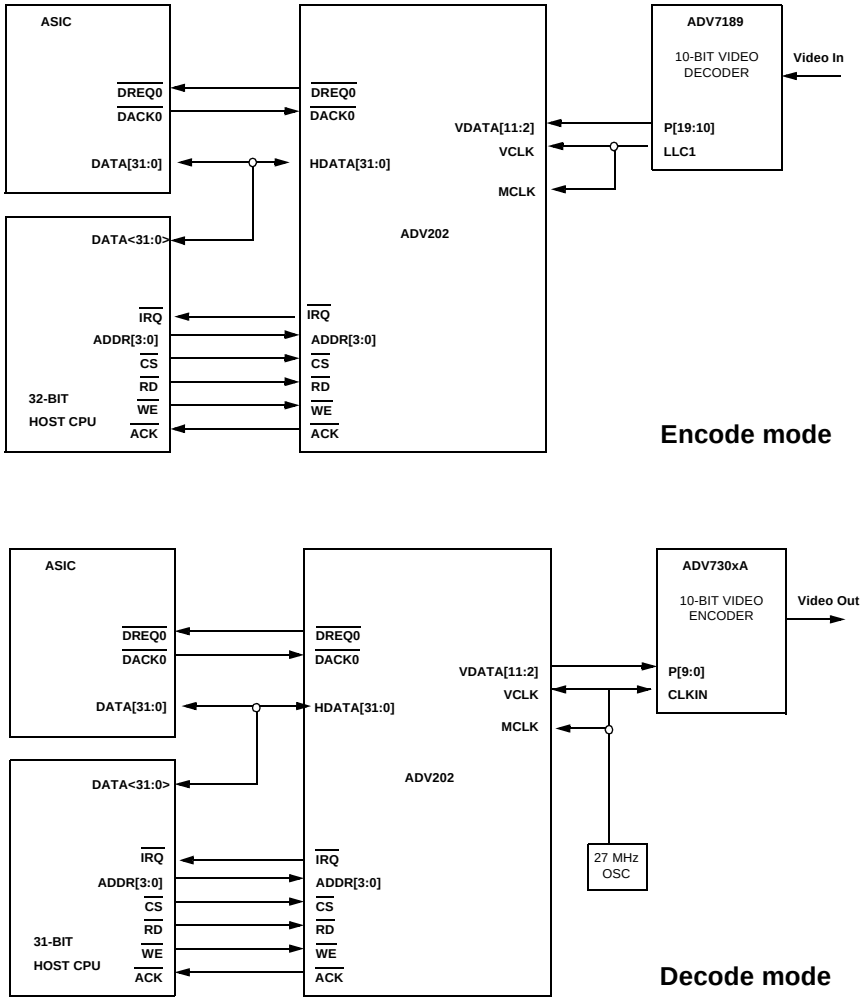


Figure 26. ADV202 with 32-bit host interfaced and 10-bit CCIR656 video application

11.5 ADV202 in HIPI configuration [Host Interface Pixel Interface]

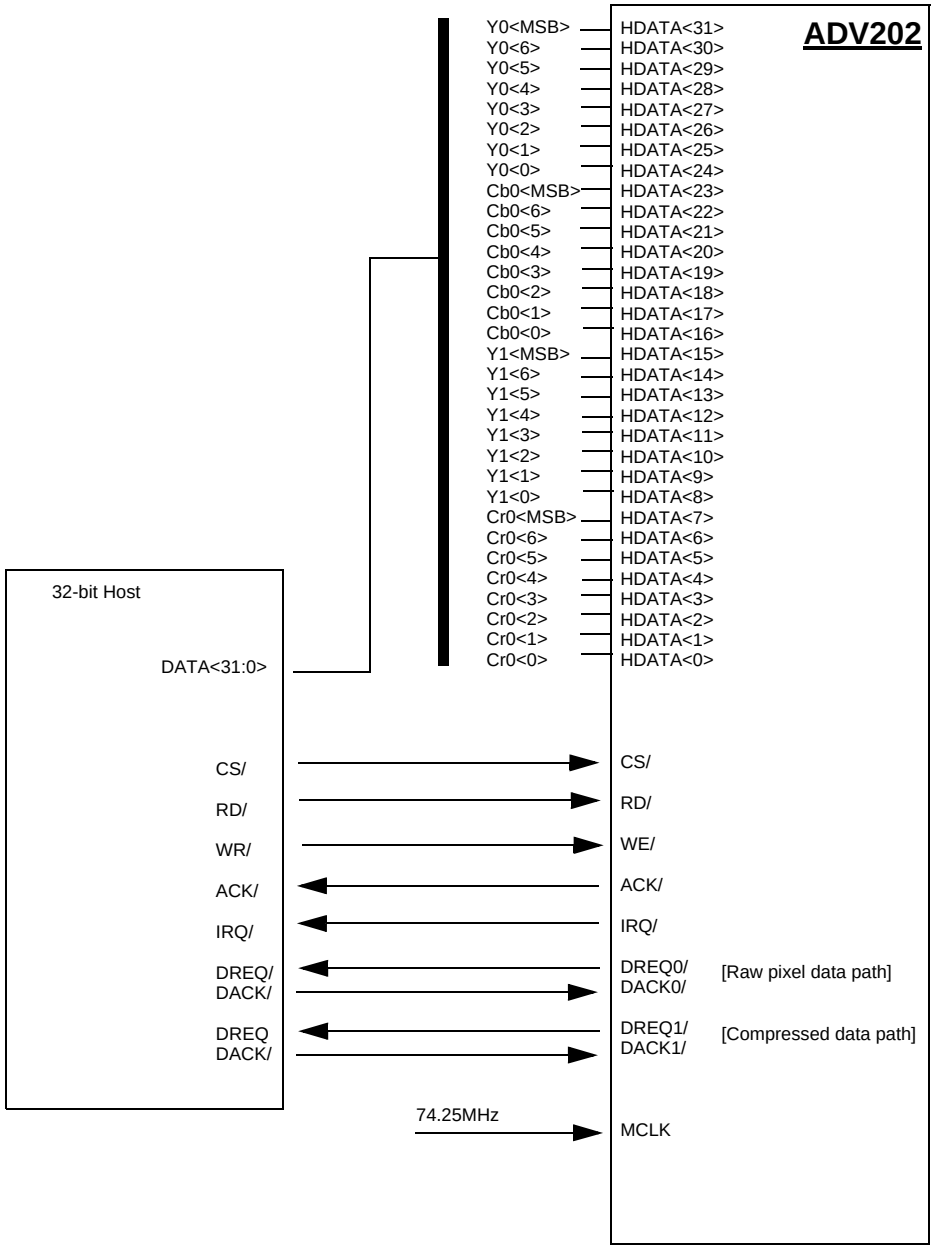


Figure 27. ADV202 in HIPI - Encode mode

11.6 ADV202 with JDATA interface

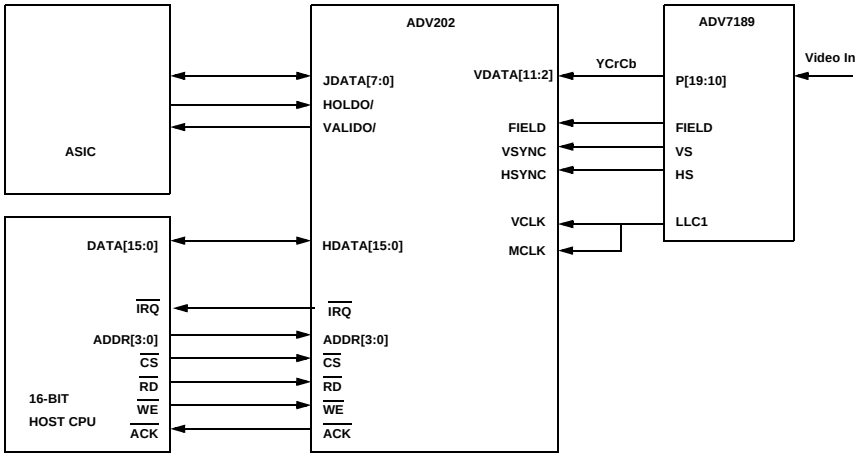


Figure 28. ADV202 with 16-bit host, dedicated JDATA output and 10-bit CCIR656 input

12. SPECIFICATONS

Specifications subject to change without notice.

Table 14: Supply Voltage

Symbol	Parameter	Min	Typ	Max	Unit
VDD	DC supply voltage, core	1.425	1.5	1.575	V
IOVdd	DC supply voltage, I/O	2.375	3.3	3.465	V
PLLVDD	DC supply voltage, PLL	1.425	1.5	1.575	V
V_{Input}	Input range	-0.3		$Vdd_{I/O}$ + 0.3	V
Temp	Operating ambient temperature range in free air	-40		+85	^oC

Table 15: Input/Output Specifications

Parameter	Description	Test Conditions	Min	Typ	Max	Unit
V_{IH}	Hi-Level Input Voltage	VDD = max	2.0	-	-	V
V_{IL}	Lo-Level Input Voltage	VDD = min	-	-	0.8	V
V_{OH}	Hi-Level Output Voltage	VDD = min, I_{OH} = -0.5mA	2.4	-	-	V
V_{OL}	Lo-Level Output Voltage	VDD = min, I_{OL} = 2mA	-	-	0.4	V
I_{IH}	Hi-Level Input Current	VDD = max, V_{IN} = VDD	-	6	-	uA
I_{IL}	Lo-Level Input Current	VDD = max, V_{IN} = 0v	-	10	-	nA
I_{OZH}	Hi-Level Three-State Leakage Current	VDD = max, V_{IN} = VDD	-	6	-	uA
I_{OZL}	Lo-Level Three-State Leakage Current	VDD = max, V_{IN} = 0v	-	10	-	nA
I_{DD}	Supply Current (Power Down)	VDD = max	-	-	100	uA
I_{DD}	Supply Current (Active)	VDD = max	-	-	100	mA
C_I	Input Pin Capacitance		-	-	8	pF
C_O	Output Pin Capacitance		-	-	8	pF

Table 16: Input clocks and RESET/

Parameter	Comments	Min	Typ	Max	Unit
tMCLK	MCLK period	13.5	-	100.5	nS
tMCLKL	MCLK Width Low	6.5	-	-	nS
tMCLKH	MCLK Width High	6.5	-	-	nS
tVCLK	VCLK period	13.9	-	50.5	nS
tVCLKL	VCLK Width Low	5.5	-	-	nS

Parameter	Comments	Min	Typ	Max	Unit
tVCLKH	VCLK Width High	5	-	-	nS
tRST	$\overline{\text{RESET}}$ Width Low	5	-	-	Mclk cycles*

* Refer to PLL section and figure 9 for definition of Mclk.

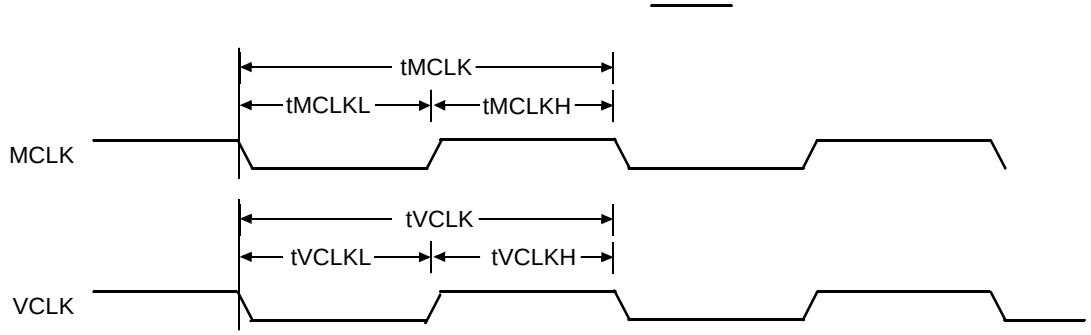


FIGURE 29. Input Clocks

Table 17. Normal Host Mode - Read Operation
[internal registers only, does not apply to FIFO accesses]

Parameter	Comments	Min	Typ	Max	Unit
tACK	RD to ACK [direct registers]	5.5ns	-	1.5xJclk + 5.5ns	
tACK	RD to ACK [indirect registers]	10.5xJclk + 5.5ns		15.5xJclk + 5.5ns	
tDRD	Read access time [direct registers]	5.5ns	-	1.5xJclk + 5.5ns	
tDRD	Read access time [indirect registers]	10.5xJclk + 5.5ns		15.5xJclk +5.5ns	
tHZRD	Data hold	2	-	8.5	nS
tSC	$\overline{\text{CS}}$ to $\overline{\text{RD}}$ setup	0	-	-	nS
tSA	Address setup	2.5	-	-	nS
tHC	$\overline{\text{CS}}$ hold	0	-	-	nS
tHA	Address hold	2.5	-	-	nS
tRH	Read inactive pulse width	15.5	-	-	nS
tRL	Read active pulse width	15.5	-	-	nS
tRCyc	Read Cycle time [direct registers]	30.5	-	-	nS

* Refer to PLL section and figure 9 for definition of Jclk.

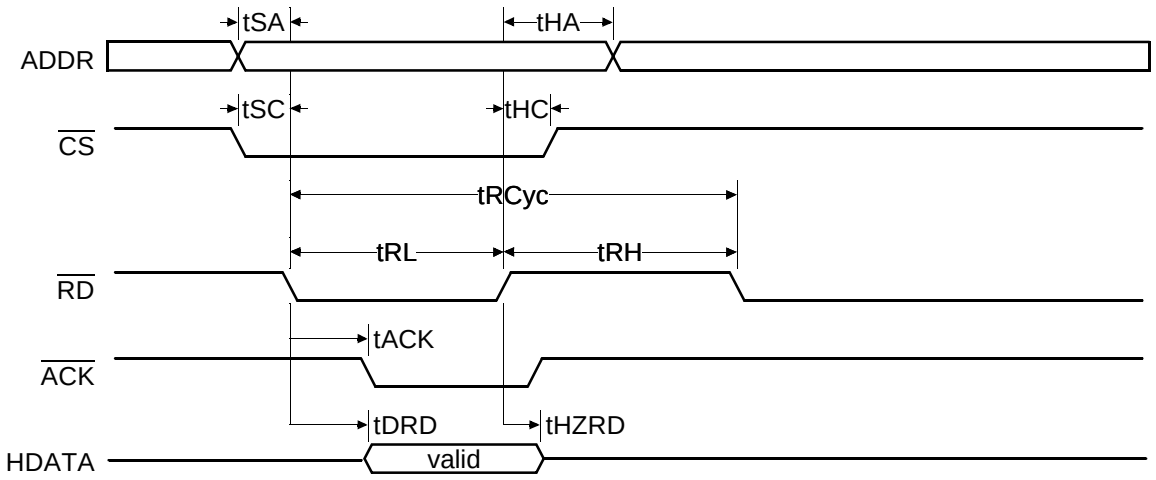


FIGURE 30. Normal host mode - Read access

Table 18: Normal Host mode - write operation
[internal registers only, does not apply to FIFO accesses]

Parameter	Comments	Min	Typ	Max	Unit
tACK[dir]	WE to ACK [direct registers]	5.5ns	-	1.5xJclk + 5.5ns	
tACK[ind]	WE to ACK [indirect registers]	5.5ns		2.5xJclk + 5.5ns	
tSD	Data setup	2.5	-	-	nS
tHD	Data hold	1.5	-	-	nS
tSC	$\overline{CS}$ to $\overline{WE}$ setup	0	-	-	nS
tSA	Address setup	2.5	-	-	nS
tHC	$\overline{CS}$ hold	0	-	-	nS
tHA	Address hold	2.5	-	-	nS
tWH	Write inactive pulse width [min. time until next WE/ pulse]	15.5*	-	-	nS
tWL	Write active pulse width	15.5*	-	-	nS
tWCyc	Write Cycle time	30.5*	-	-	nS

* Specified for Jclk = 150MHz. Refer to PLL section for definition of Jclk.

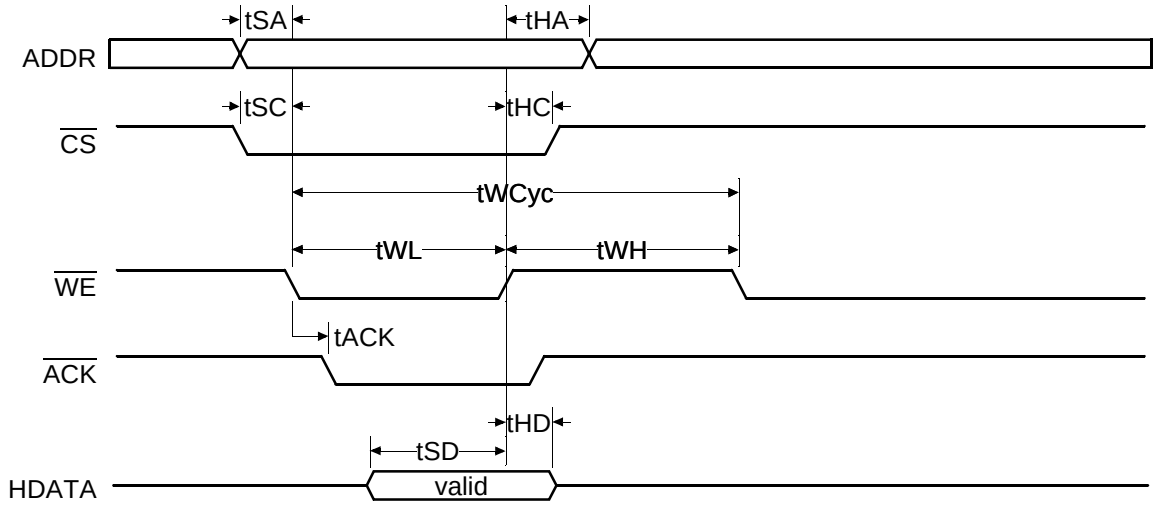


FIGURE 31. Normal host mode - Write Access

Table 19: DREQ/DACK DMA mode - single FIFO write

Parameter	Comments	Min	Typ	Max	Unit
DREQpulse	DREQ pulse width[a]	1	-	15	Jclkcycles[d]
tDREQ	DACK assert to subsequent DREQ delay	2.5	-	3.5xJclk + 5.5ns	Jclkcycles[d]
tWEsu	WE to DACK setup	0	-	-	ns
tSU	Data to DACK de-assert setup	2.5	-	-	ns
tHD	Data to DACK de-assert hold	2.5	-	-	ns
DACKlo	DACK assert pulse width	8[b]	-	-	ns
DACKhi	DACK de-assert pulse width	8[c]	-	-	ns
tWEhd	WE hold after DACK de-assert	0	-	-	ns
WFSRQ	WE de-assert to FSRQ de-assert [FIFO full]	1.5		2.5xJclk + 5.5ns	Jclkcycles[d]
TDREQrtn	WE to DREQ de-assert [DRxPULS=0]	2.5		3.5xJclk + 5.5ns	Jclkcycles[d]

a. Applies to assigned DMA channel if EDMOD0 or EDMOD1<14:11> is programmed to a value of 0.
b. Assumes Jclk = 150MHz. Refer to PLL section for definition of Jclk.
c. Assumes Jclk = 150MHz. Refer to PLL section for definition of Jclk.
d. Refer to PLL section for definition of Jclk.

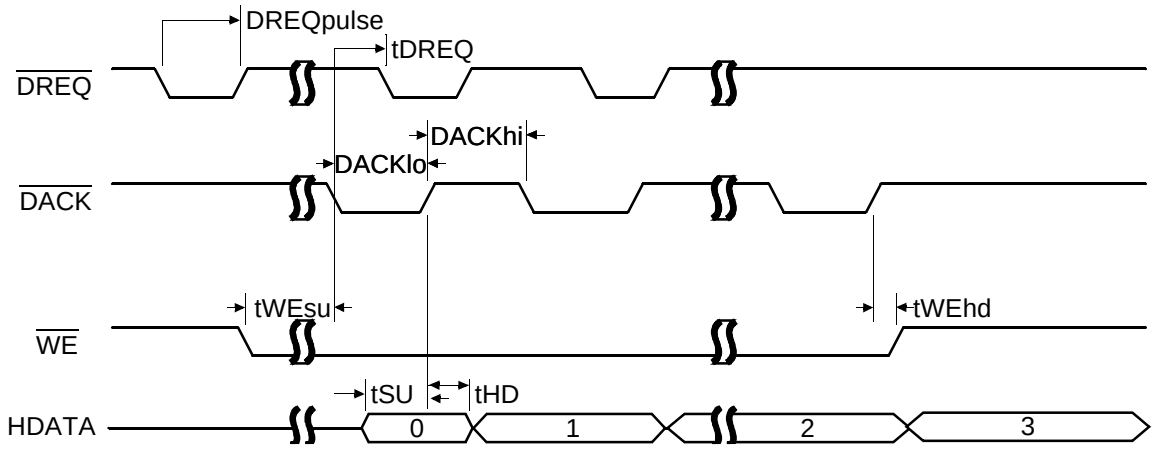


FIGURE 32. Single Write Cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11> NOT programmed to a value of 0.

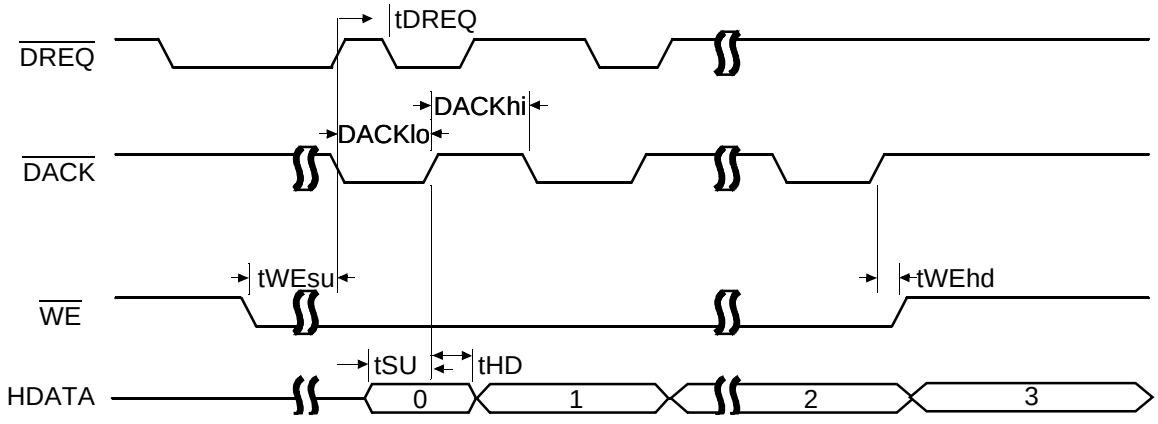


FIGURE 33. Single Write Cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11> programmed to a value of 0.

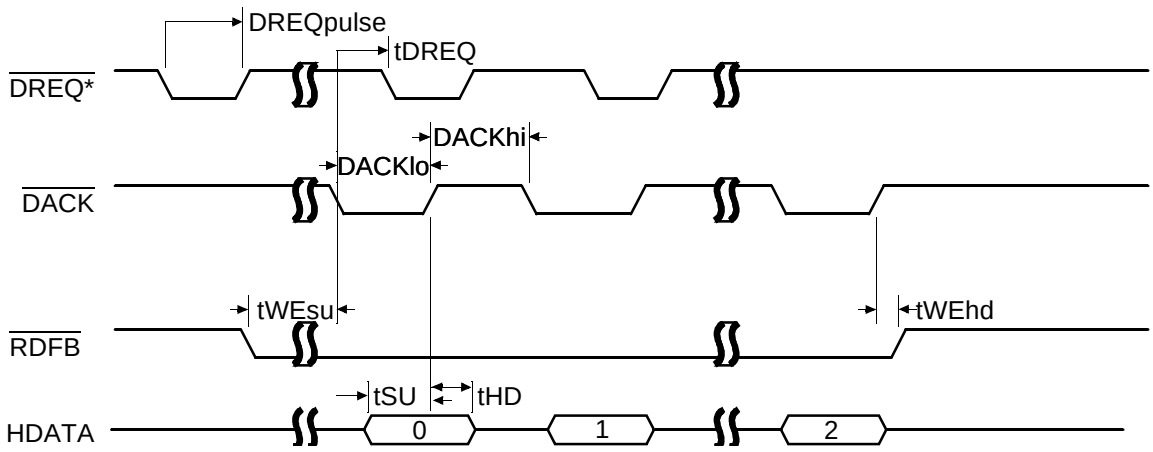


FIGURE 34. Single Write Cycle in Fly_By DMA mode

* DREQ/ pulse width is programmable. Refer to Table 19 and figures 32 and 33.

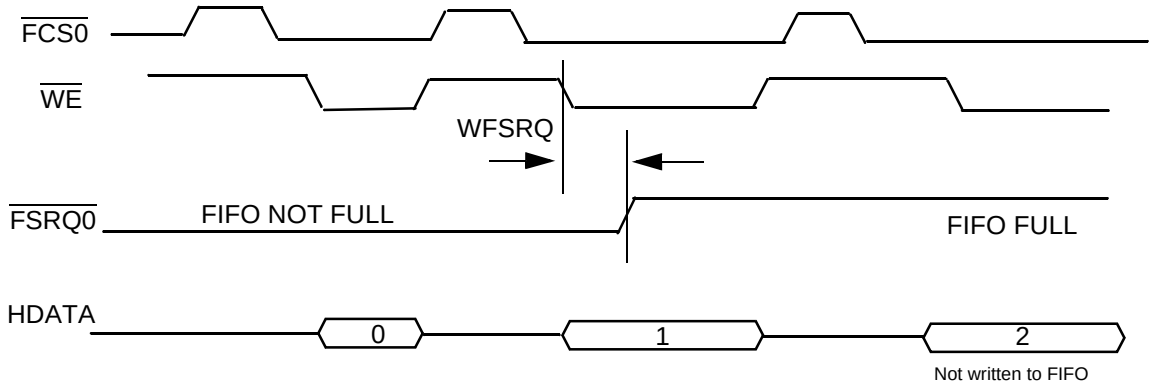


FIGURE 35. Single Write Cycle for DCS DMA mode
This figure applies to ADV202 REV0.1 and upwards

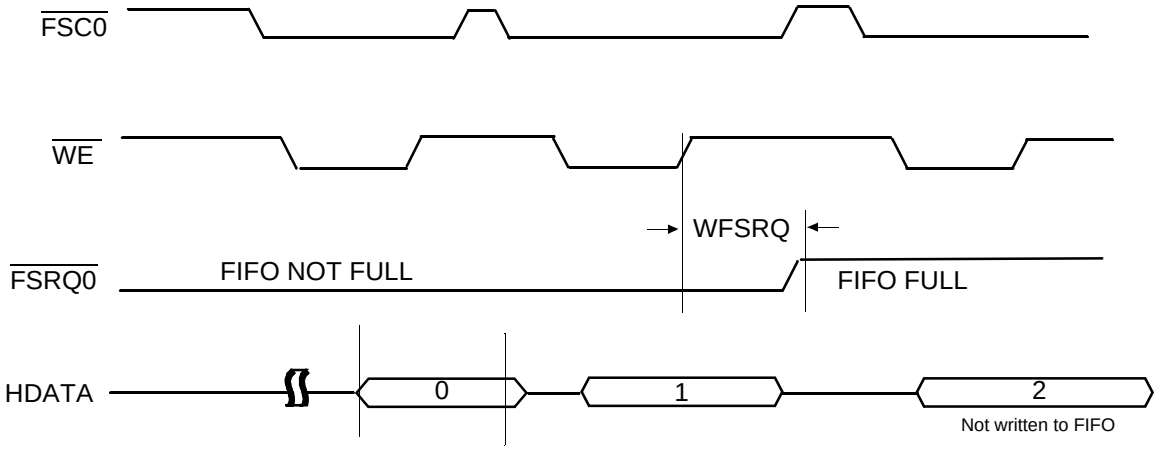


FIGURE 36. Single Write Cycle for DCS DMA mode
This figure applies to REV0.0 ADV202s

Table 20 : DREQ/DACK DMA mode DMA - single FIFO read

Parameter	Comments	Min	Typ	Max	Unit
DREQpulse	DREQ pulse width	1	-	15	Jclkcycles[a]
tDREQ	DACK assert to subsequent DREQ delay	2.5	-	3.5xJclk + 5.5ns	Jclkcycles[a]
tRDsu	RD to DACK setup	0	-	-	ns
tRD	DACK to data valid	2.5	-	-	ns
tHD	Data hold	1.5	-	-	ns
DACKlo	DACK assert pulse width	8.0	-	-	ns
DACKhi	DACK de-assert pulse width	8.0	-	-	ns
tRDhd	RD hold after DACK de-assert	0	-	-	ns
RDFSRQ	RD assert to FSRQ de-assert [FIFO empty]	2.0		2.5xJclk + 5.5ns	Jclkcycles
tDREQrtn	RD to DREQ de-assert [DRxPULS=0]	3.0		3.5xJclk + 5.5ns	Jclkcycles

a. Applies to assigned DMA channel if EDMOD0 or EDMOD1<14:11> is programmed to a value of 0. Refer to PLL section for definition of Jclk.

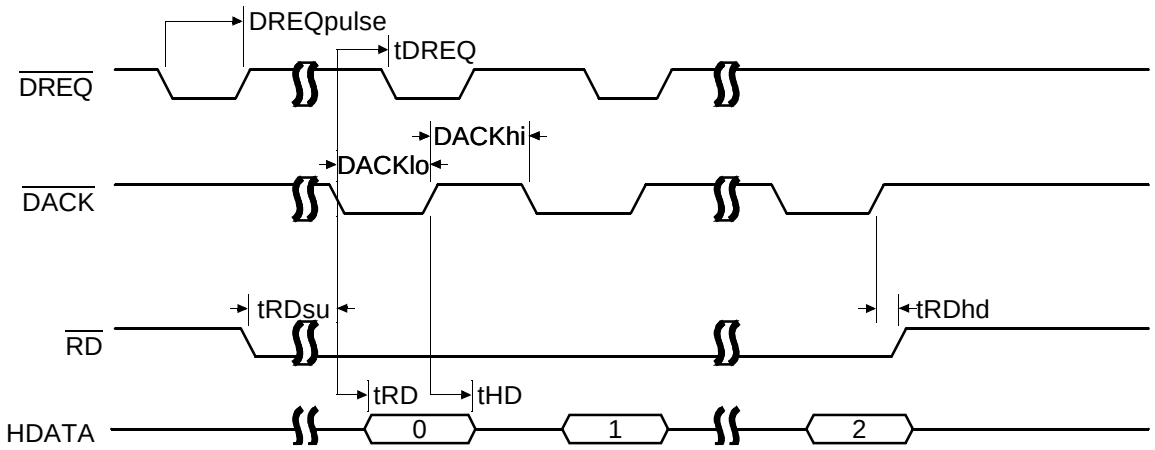


FIGURE 37. Single Read Cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11> NOT programmed to a value of 0.

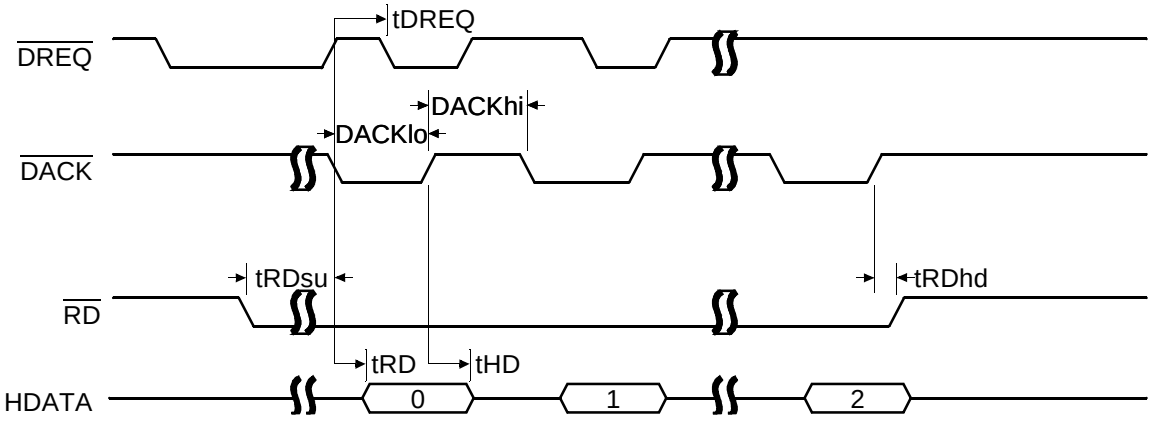
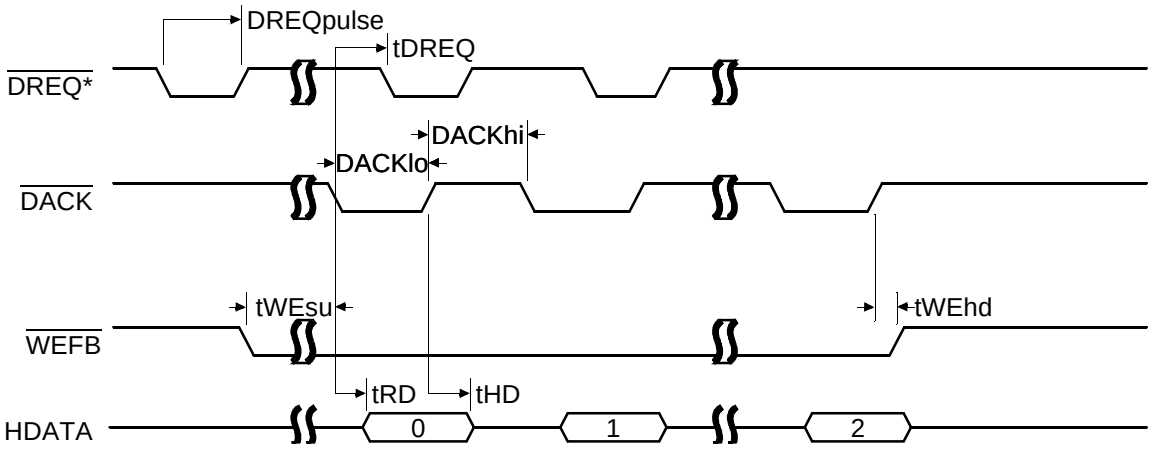


FIGURE 38. Single Read Cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11>programmed to a value of 0.



* DREQ/ pulse width is programmable. Refer to table 20 and figures 37 and 38.

FIGURE 39. Single Read cycle in Fly-By DMA mode

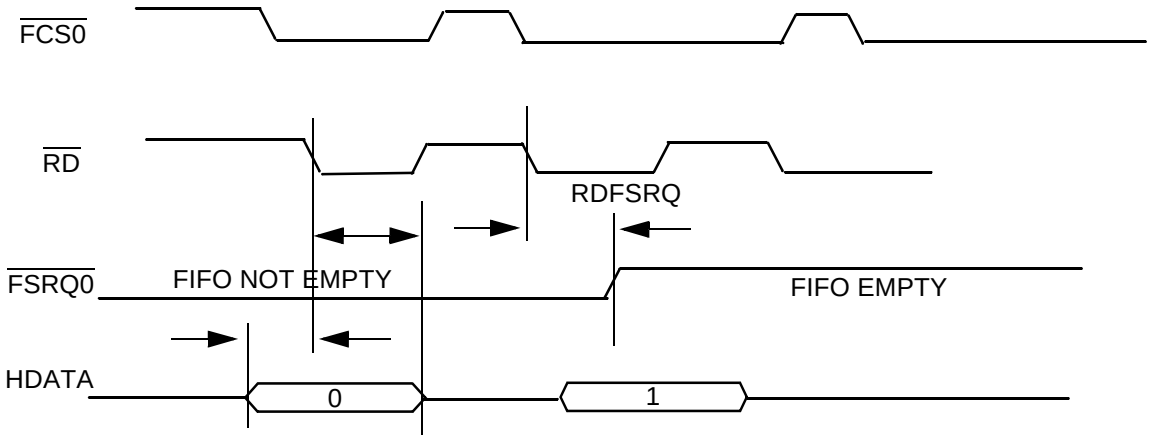


FIGURE 40. Single Read cycle in DCS- DMA mode

Table 21: External DMA - FIFO write - Burst mode

Parameter	Comments	Min	Typ	Max	Unit
DREQpulse	DREQ pulse width[a]	1		15	Jclkcycles[b]
tDREQrtn	DACK to DREQ de-assert [DRxpulse=0]	2.5	-	3.5	Jclkcycles[b]
tDACKsu	DACK to WE setup	0	-	-	ns
tSU	Data setup	2.5	-	-	ns
tHD	Data hold	2.5	-	-	ns
WElo	WE assert pulse width	8.0	-	-	ns
WEhi	WE de-assert pulse width	8.0	-	-	ns
tDREQwait	DACK de-assert to next DREQ	4.0	-	4.5[c]	Jclkcycles[b]

a. Applies to assigned DMA channel if EDMOD0 or EDMOD1<14:11> is programmed to a value of 0.
b. Assumes Jclk = 150MHz. Refer to PLL section for definition of Jclk.
c. If sufficient space is available in FIFO.

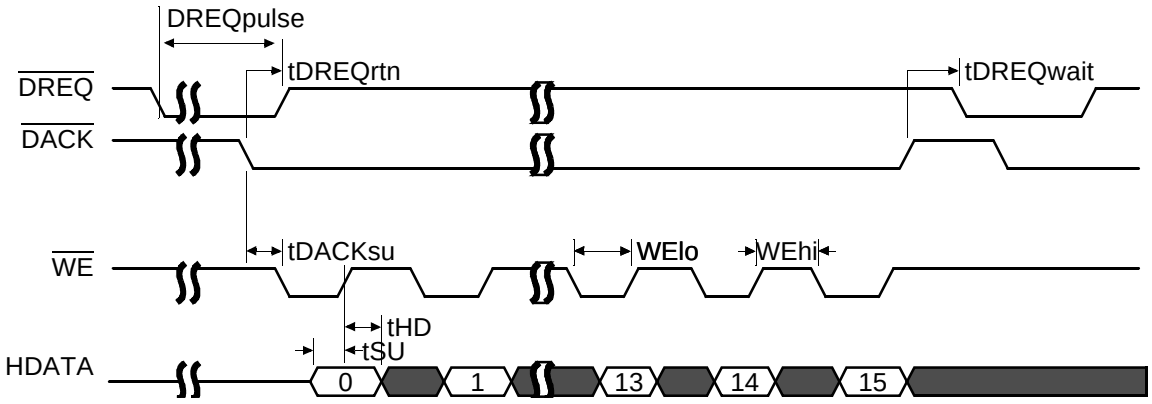


FIGURE 41. Burst Write Cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11> NOT programmed to a value of 0.

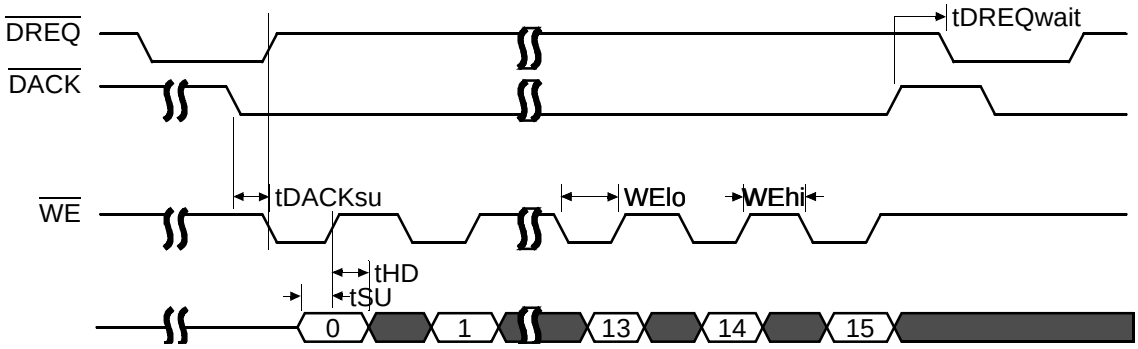
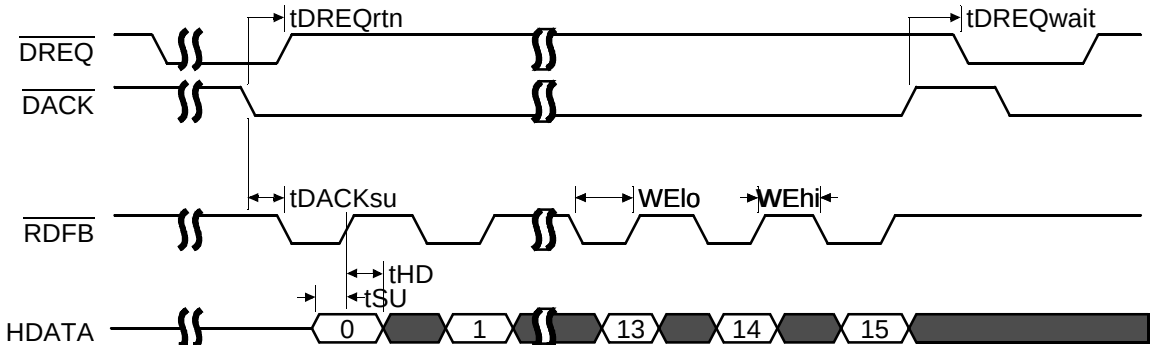


FIGURE 42. Burst write cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx <14:11> programmed to a value of 0.



* DREQ/ pulse width is programmable. Refer to table 21 and figures 41 and 42.

FIGURE 43. Burst write cycles for Fly-by DMA mode

Table 22 : DREQ/DACK DMA mode - FIFO read - Burst mode

Parameter	Comments	Min	Typ	Max	Unit*
DREQpulse	DREQ pulse width[a]	1		15	Jclkcycles[b]
tDREQrtn	DACK to DREQ deassert [DRxPULS=0]	2.5	-	3.5	Jclkcycles[b]
tDACKsu	DACK to WE setup	0	-	-	ns
tRD	DACK to Data valid	2.5	-		ns
tHD	Data hold	2.5	-	-	ns
RDlo	RD assert pulse width	8.0	-	-	ns
RDhi	RD de-assert pulse width	8.0	-	-	ns
tDREQwait	DACK de-assert to next DREQ	2.5	-	3.5[c]	Jclkcycles

a. Applies to assigned DMA channel if EDMOD0 or EDMOD1<14:11> is programmed to a value of 0.
b. Assumes Jclk = 150MHz. Refer to PLL section for definition of Jclk.
c. If sufficient data is available in FIFO.

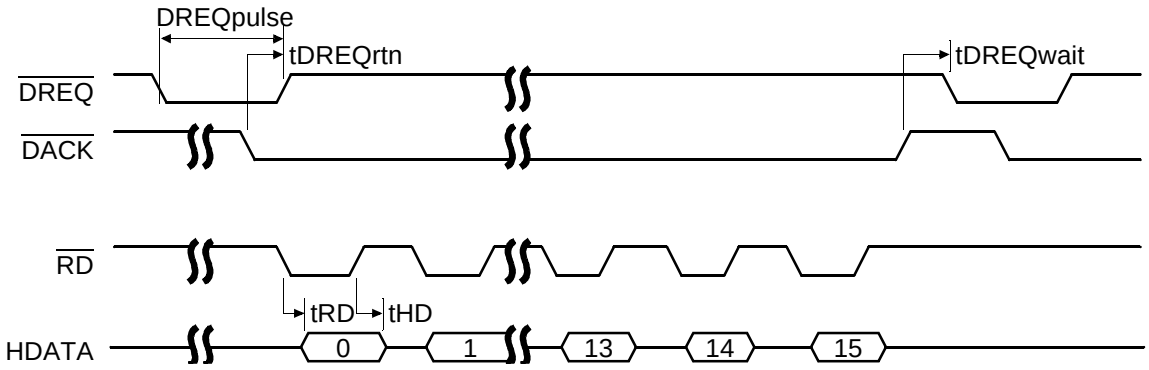


FIGURE 44. Burst Read Cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11> NOT programmed to a value of 0.

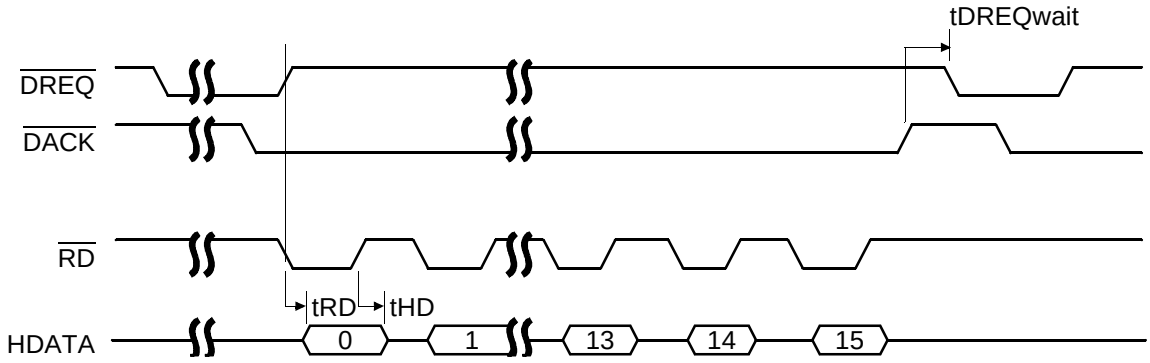
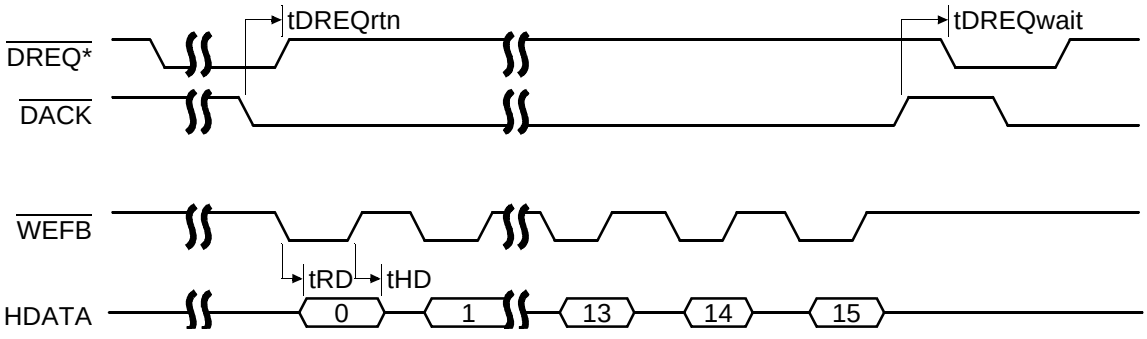


FIGURE 45. Burst read cycle for DREQ/DACK DMA mode for assigned DMA channel EDMODx<14:11> programmed to a value of 0.



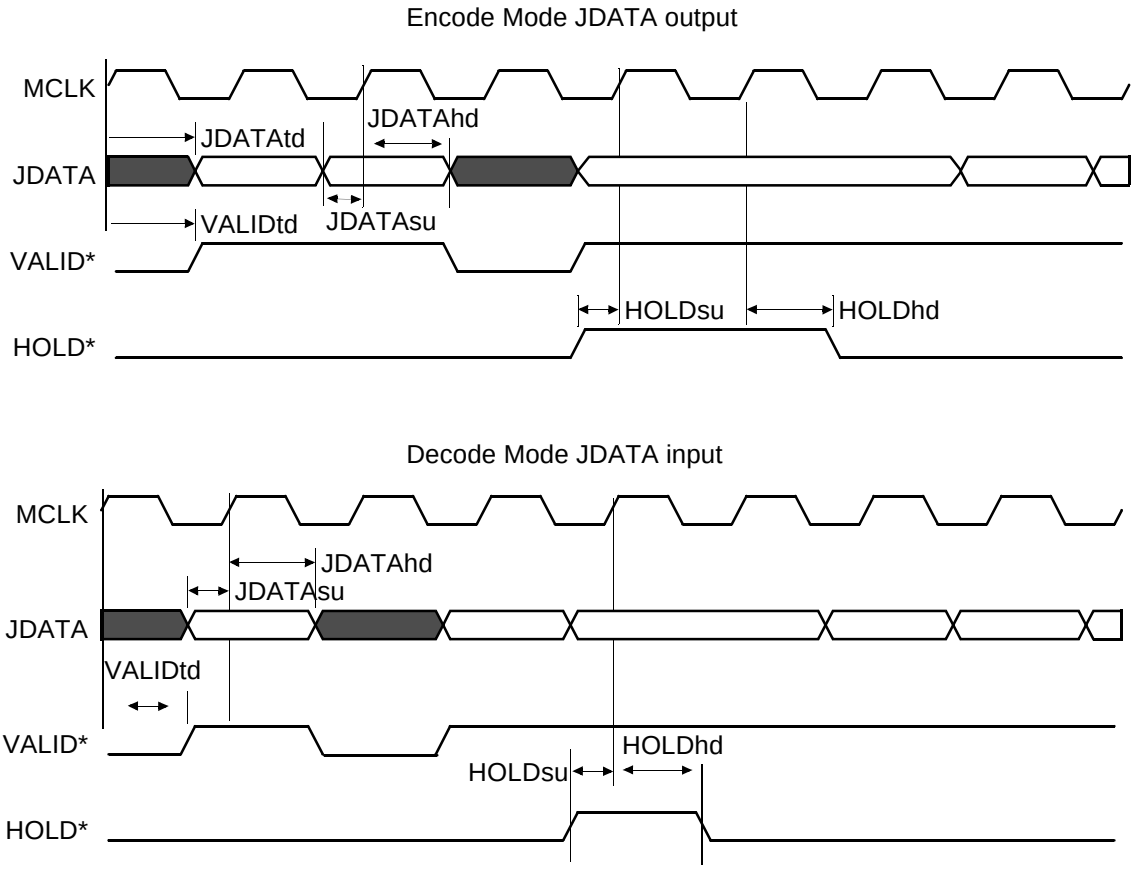
* DREQ/ pulse width is programmable. Refer to table 22 and figures 41 and 42.

FIGURE 46. Burst read cycle for Fly-by DMA mode

Table 23: Streaming Mode [JDATA] - FIFO read/write

Parameter	Comments	Min	Typ	Max	Unit
JDATAtd	MCLK to JDATA valid	1.5	-	3.0	Jclkcycles[a]
VALIDtd	MCLK to VALID assert/ de-assert	1.5	-	3.0	Jclkcycles[a]
HOLDsu	HOLD setup to rising MCLK	3.5	-	-	ns
HOLDhd	HOLD hold from rising MCLK	3.5	-	-	ns
JDATAsu	JDATA setup to rising MCLK	3.5	-	-	ns
JDATAhd	JDATA hold from rising MCLK	3.5	-	-	ns

a. Refer to PLL section for definition of Jclk.



* HOLD and VALID polarity is programmable in register EDMOD0. The figure above shows VALID and HOLD to be programmed to active high polarity.

FIGURE 47 Streaming Mode timing

Table 24: Video Mode Timing

Parameter	Comments	Min	Typ	Max	Unit
VDATAtd	VCLK to VDATA valid delay (VDATA output)	-	-	-	VCLKcycles
VDATAsu	VDATA setup to rising VCLK (VDATA input)	2	-	4	VCLKcycles
VDATAhd	VDATA hold from rising VCLK (VDATA input)	2	-	4	VCLKcycles
HSYNCsu	HSYNC setup to rising VCLK	1	-	3	
HSYNChd	HSYNC hold from rising VCLK	2	-	4	
VSYNCsu	VSYNC setup to rising VCLK	1	-	3	VCLKcycles
VSYNChd	VSYNC hold from rising VCLK	2	-	4	VCLKcycles
VSYNCtd	VCLK to VFRM valid delay		-	-	VCLKcycles
FIELDsu	VDATA setup to rising VCLK	2	-	4	VCLKcycles
FIELDhd	VDATA hold from rising VCLK	1	-	3	VCLKcycles
SYNC DELAY	Decode data sync delay for HD input with EAV/SAV codes		7		VCLKcycles
SYNC DELAY	Decode data sync delay for SD input with EAV/SAV codes		9		VCLKcycles
SYNC DELAY	Decode data sync delay for DUAL_LANE [Extended] input		7		VCLKcycles
SYNC DELAY	Decode data sync delay for HVF input [from first rising VCLK after HSYNC low to first data sample]		10		VCLKcycles

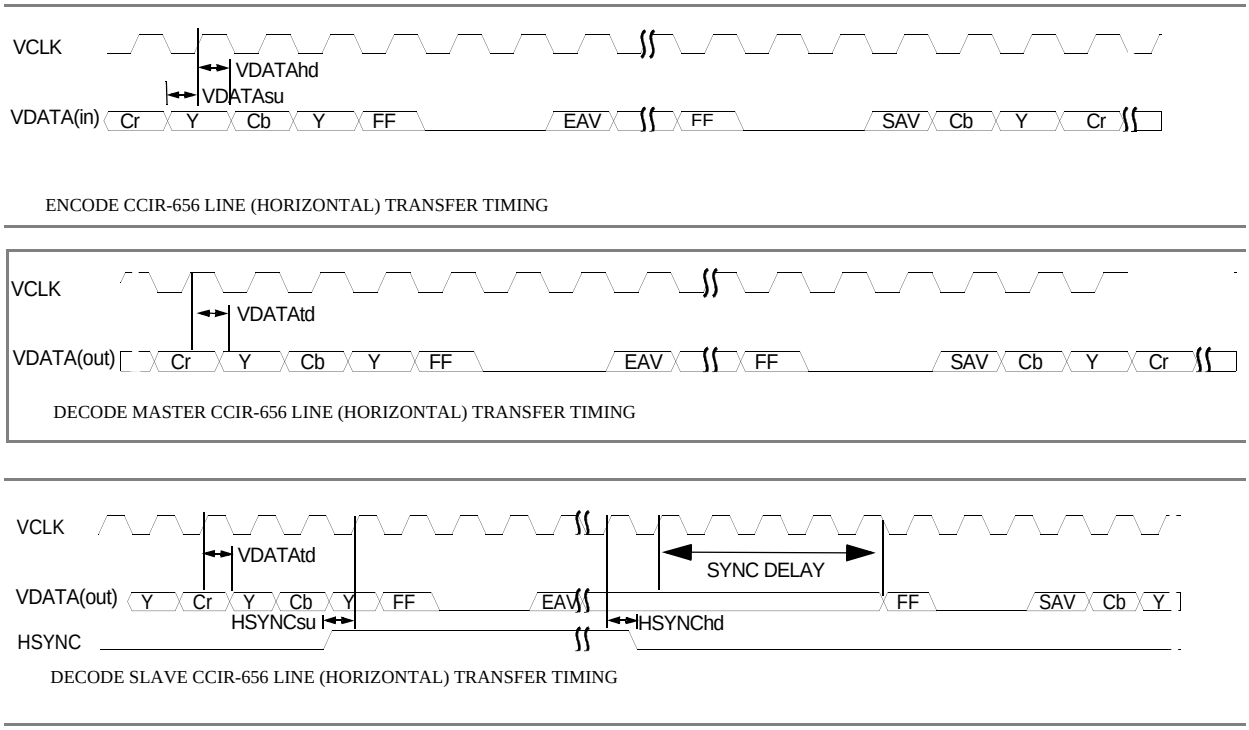


FIGURE 48 Video Mode -Horizontal Timing

Table 25: RAW Pixel Mode Timing

Parameter	Comments	Min	Typ	Max	Unit*
VDATAtd	VCLK to PIXELDATA valid delay (PIXELDATA output)	-	-	-	VCLKcycles
VDATAsu	PIXELDATA setup to rising VCLK (PIXEL-DATA input)	2	-	4	VCLKcycles
VDATAhd	PIXELDATA hold from rising VCLK (PIXEL-DATA input)	2	-	4	VCLKcycles
VRDYtd	VCLK to VRDY valid delay	-	-	-	VCLKcycles
VFRMsu	VFRM setup to rising VCLK (VRAME input)	1	-	3	VCLKcycles
VFRMhd	VFRM hold from rising VCLK (VRAME input)	2	-	4	VCLKcycles
VFRMtd	VCLK to VFRM valid delay (VFRAME output)	-	-	-	VCLKcycles
VSTRBsu	VSTRB setup to rising VCLK	2	-	4	VCLKcycles
VSTRBhd	VSTRB hold from rising VCLK	1	-	3	VCLKcycles

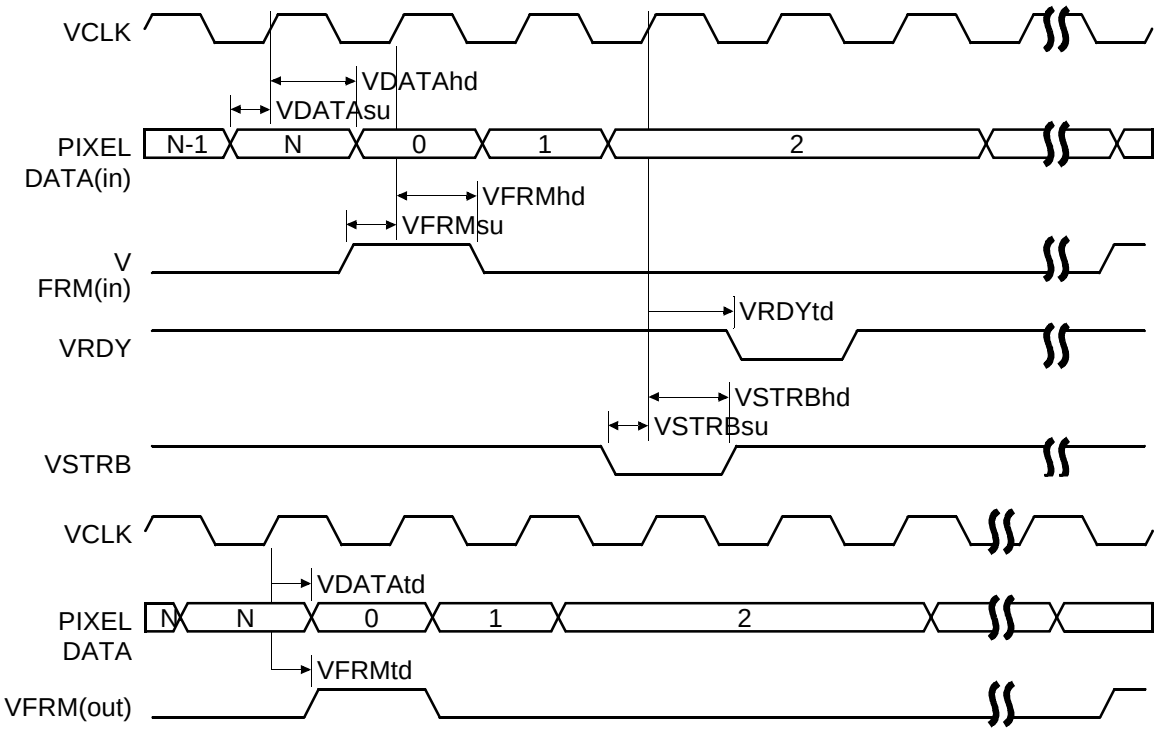


FIGURE 49. Raw pixel mode timing

Table 26: SPI Timing

Parameter	Comments	Min	Typ	Max	Unit
SCLK_fall	S_CLK fall time	-	-	-	ns
SCLK_ris	S_CLK rise time	-	-	-	ns
SPIhd	Data input hold time	-	-	-	ns
SPIsu	Data input setup time	-	-	-	ns
NCSELhd	Not active hold time	-	-	-	ns
NCSELsu	Not active setup time	-	-	-	ns
CSELsu	Active setup time	-	-	-	ns
CSELhd	Active hold time	-	-	-	ns
DStime	Deselect time	-	-	-	ns
Th	Clock high time	-	-	-	ns
Tl	Clock low time	-	-	-	ns
OPh	Output hold time	-	-	-	ns
CLKov	Clock low to Output Valid	-	-	-	ns

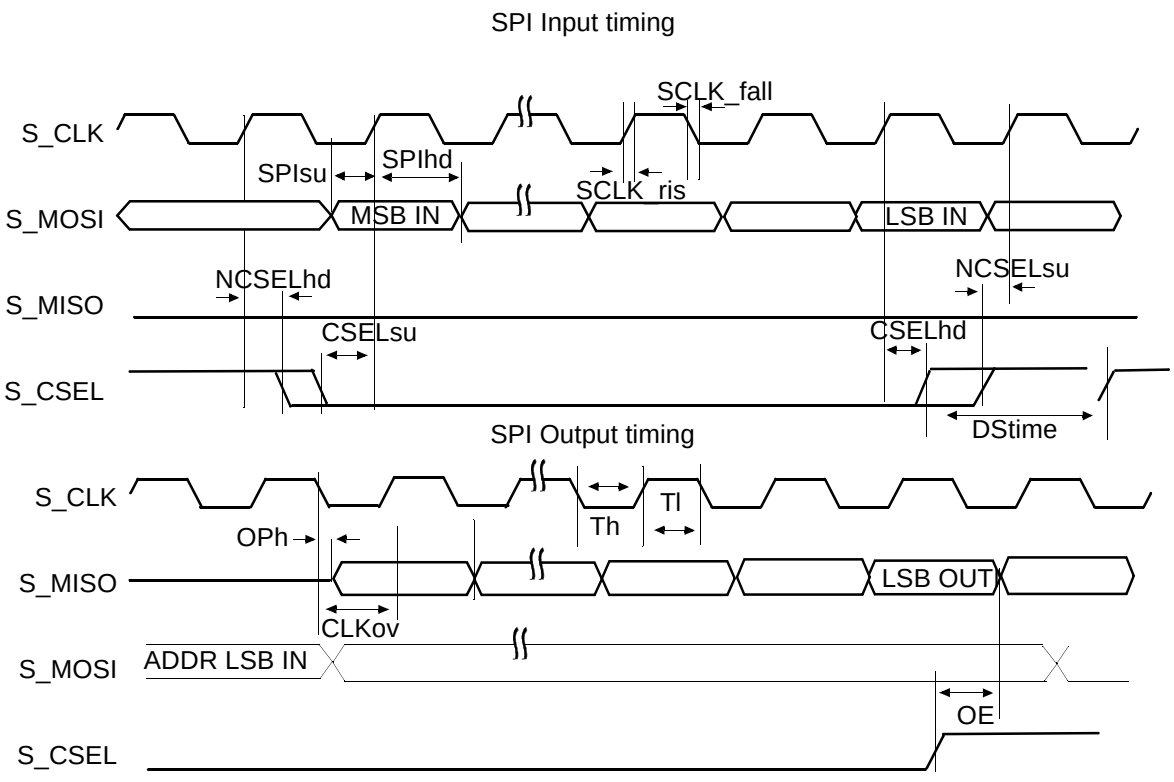


FIGURE 50. SPI timing

13. PIN ASSIGNMENT

Table 27. Pin assignment for ADV202 - 121-pin package

	PIN NUMBER	PIN DESCRIPTION
1	A1	DGND
2	A2	HDATA[2]
3	A3	VDD
4	A4	DGND
5	A5	HDATA[0]
6	A6	HDATA[1]
7	A7	VDATA[1]
8	A8	VDD
9	A9	DGND
10	A10	VDATA[0]
11	A11	DGND
12	B1	HDATA[3]
13	B2	HDATA[4]
14	B3	HDATA[5]
15	B4	HDATA[7]
16	B5	HDATA[8]
17	B6	IOVDD
18	B7	VDATA[6]
19	B8	VDATA[5]
20	B9	VDATA[4]
21	B10	VDATA[2]
22	B11	VDATA[3]
23	C1	DGND
24	C2	HDATA[6]
25	C3	HDATA[9]
26	C4	HDATA[10]
27	C5	HDATA[11]
28	C6	IOVDD
29	C7	VDATA[9]
30	C8	IOVDD
31	C9	VDATA[8]
32	C10	VDATA[7]
33	C11	DGND
34	D1	HDATA[12]
35	D2	HDATA[13]

	PIN NUMBER	PIN DESCRIPTION
36	D3	HDATA[14]
37	D4	HDATA[15]
38	D5	IOVDD
39	D6	DGND
40	D7	VDD
41	D8	VSYNC
42	D9	HSYNC
43	D10	VDATA[10]
44	D11	VDATA[11]
45	E1	DGND
46	E2	HDATA[18]_VDATA[14]
47	E3	HDATA[17]_VDATA[13]
48	E4	HDATA[16]_VDATA[12]
49	E5	DGND
50	E6	DGND
51	E7	DGND
52	E8	IOVDD
53	E9	VCLK
54	E10	FIELD
55	E11	DGND
56	F1	DGND
57	F2	HDATA[19]_VDATA[15]
58	F3	HDATA[20]_VDATA[16]
59	F4	HDATA[21]_VDATA[17]
60	F5	DGND
61	F6	DGND
62	F7	DGND
63	F8	DREQ0/
64	F9	DACK0/
65	F10	DREQ1/
66	F11	DGND
67	G1	DGND
68	G2	HDATA[22]_VDATA[18]
69	G3	HDATA[23]_VDATA[19]
70	G4	HDATA[24]_VDATA[20]_JDATA[0]
71	G5	DGND
72	G6	DGND
73	G7	DGND
74	G8	IOVDD

	PIN NUMBER	PIN DESCRIPTION
75	G9	DACK1/
76	G10	IRQ/
77	G11	DGND
78	H1	HDATA[28]_JDATA[4]
79	H2	HDATA[27]_VDATA[23]_JDATA[3]
80	H3	HDATA[26]_VDATA[22]_JDATA[2]
81	H4	HDATA[25]_VDATA[21]_JDATA[1]
82	H5	IOVDD
83	H6	DGND
84	H7	VDD
85	H8	ACK/
86	H9	RD/
87	H10	ADDR[1]
88	H11	ADDR[3]
89	J1	DGND
90	J2	HDATA[31]_JDATA[7]
91	J3	HDATA[30]_JDATA[6]
92	J4	HDATA[29]_JDATA[5]
93	J5	IOVDD
94	J6	TEST1
95	J7	WE/
96	J8	CS/
97	J9	ADDR[0]
98	J10	TEST3
99	J11	DGND
100	K1	SCOMM[4]
101	K2	SCOMM[3]
102	K3	SCOMM[0]
103	K4	SCOMM[1]
104	K5	IOVDD
105	K6	IOVDD
106	K7	IOVDD
107	K8	ADDR[2]
108	K9	TEST2
109	K10	TEST5
110	K11	DGND
111	L1	DGND
112	L2	SCOMM[7]
113	L3	SCOMM[6]

	PIN NUMBER	PIN DESCRIPTION
114	L4	SCOMM[5]
115	L5	SCOMM[2]
116	L6	TEST4
117	L7	RESET/
118	L8	DGND
119	L9	MCLK
120	L10	PLLVDD
121	L11	DGND

Table 28: Pin assignment for 144-pin package

1	A1	DGND
2	A2	HDATA[2]
3	A3	HDATA[1]
4	A4	HDATA[0]
5	A5	DGND
6	A6	DGND
7	A7	DGND
8	A8	DGND
9	A9	VDATA[2]
10	A10	VDATA[1]
11	A11	VDATA[0]
12	A12	DGND
13	B1	HDATA[5]
14	B2	HDATA[4]
15	B3	HDATA[3]
16	B4	IOVDD
17	B5	DGND
18	B6	VDD
19	B7	VDD
20	B8	DGND
21	B9	IOVDD
22	B10	VDATA[5]
23	B11	VDATA[4]
24	B12	VDATA[3]
25	C1	HDATA[8]
26	C2	HDATA[7]
27	C3	HDATA[6]
28	C4	IOVDD

29	C5	DGND
30	C6	VDD
31	C7	VDD
32	C8	DGND
33	C9	IOVDD
34	C10	VDATA[8]
35	C11	VDATA[7]
36	C12	VDATA[6]
37	D1	HDATA[11]
38	D2	HDATA[10]
39	D3	HDATA[9]
40	D4	IOVDD
41	D5	DGND
42	D6	VDD
43	D7	VDD
44	D8	DGND
45	D9	IOVDD
46	D10	VDATA[11]
47	D11	VDATA[10]
48	D12	VDATA[9]
49	E1	HDATA[14]
50	E2	HDATA[13]
51	E3	HDATA[12]
52	E4	DGND
53	E5	DGND
54	E6	DGND
55	E7	DGND
56	E8	DGND
57	E9	FIELD
58	E10	VSYNC
59	E11	HSYNC
60	E12	VCLK
61	F1	HDATA[18]_VDATA[14]
62	F2	HDATA[17]_VDATA[13]
63	F3	HDATA[16]_VDATA[12]
64	F4	HDATA[15]
65	F5	DGND
66	F6	DGND
67	F7	DGND
68	F8	DGND

69	F9	DACK1/
70	F10	DREQ1/
71	F11	DACK0/
72	F12	DREQ0/
73	G1	HDATA[22]_VDATA[18]
74	G2	HDATA[21]_VDATA[17]
75	G3	HDATA[20]_VDATA[16]
76	G4	HDATA[19]_VDATA[15]
77	G5	DGND
78	G6	DGND
79	G7	DGND
80	G8	DGND
81	G9	DGND
82	G10	IRQ/
83	G11	ACK/
84	G12	RD/
85	H1	HDATA[26]_VDATA[22]_JDATA[2]
86	H2	HDATA[25]_VDATA[21]_JDATA[1]
87	H3	HDATA[24]_VDATA[20]_JDATA[0]
88	H4	HDATA[23]_VDATA[19]
89	H5	DGND
90	H6	DGND
91	H7	DGND
92	H8	DGND
93	H9	DGND
94	H10	WR/
95	H11	CS/
96	H12	ADDR[0]
97	J1	HDATA[30]_JDATA[6]
98	J2	HDATA[29]_JDATA[5]
99	J3	HDATA[28]_JDATA[4]
100	J4	HDATA[27]_VDATA[23]_JDATA[3]
101	J5	DGND
102	J6	VDD
103	J7	VDD
104	J8	DGND
105	J9	DGND
106	J10	ADDR[1]
107	J11	ADDR[2]
108	J12	ADDR[3]

109	K1	SCOMM[1]
110	K2	SCOMM[0]
111	K3	HDATA[31]_JDATA[7]
112	K4	IOVDD
113	K5	DGND
114	K6	VDD
115	K7	VDD
116	K8	DGND
117	K9	IOVDD
118	K10	TEST3
119	K11	TEST2
120	K12	TEST1
121	L1	SCOMM[4]
122	L2	SCOMM[3]
123	L3	SCOMM[2]
124	L4	IOVDD
125	L5	DGND
126	L6	VDD
127	L7	VDD
128	L8	DGND
129	L9	IOVDD
130	L10	TEST5
131	L11	RESET/
132	L12	MCLK
133	M1	DGND
134	M2	SCOMM[7]
135	M3	SCOMM[6]
136	M4	SCOMM[5]
137	M5	DGND
138	M6	DGND
139	M7	DGND
140	M8	DGND
141	M9	TEST4
142	M10	PLLVDD
143	M11	DGND
144	M12	DGND

14. PACKAGE DIMENSIONS

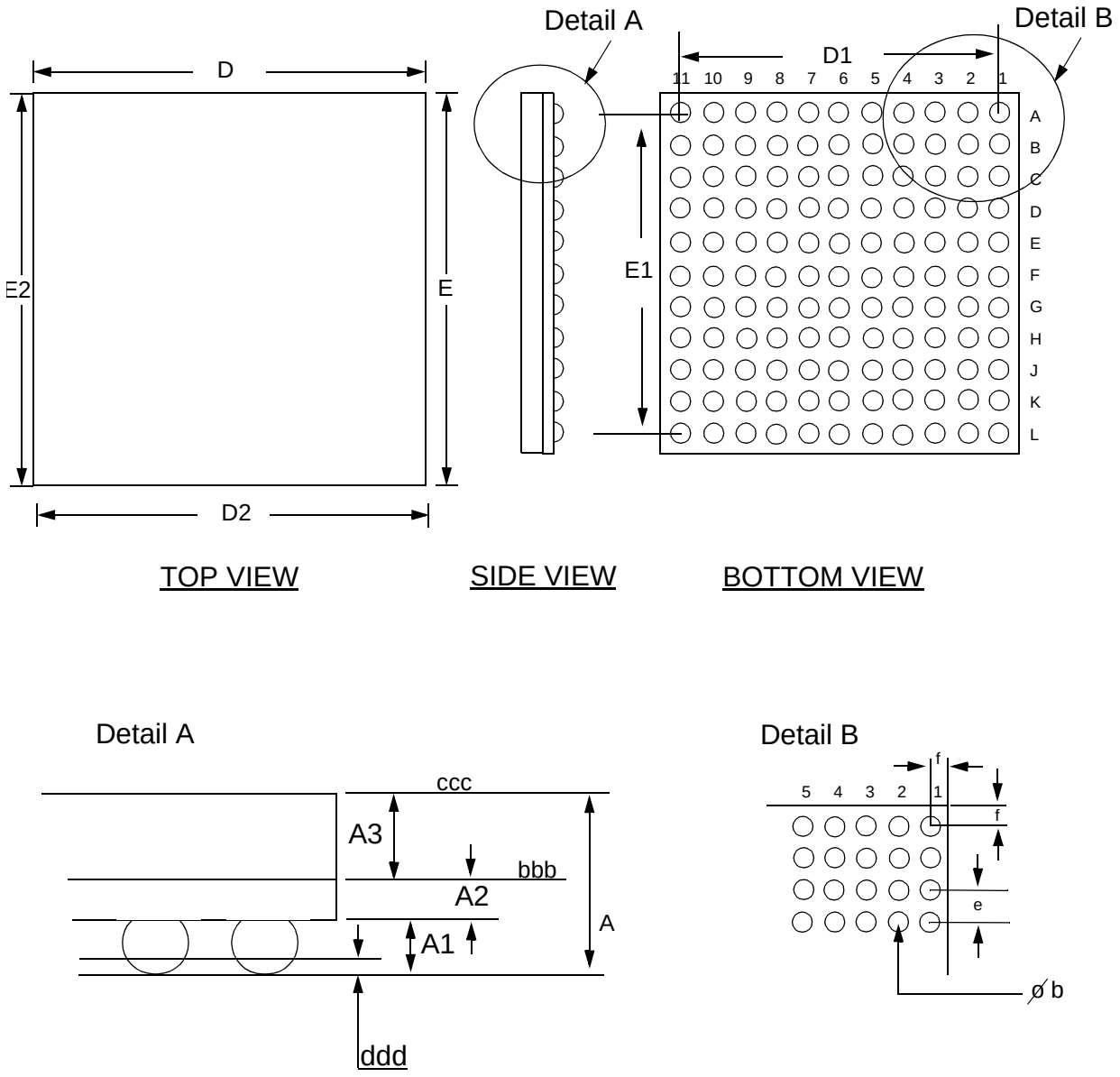


FIGURE 51. Package dimensions for 121-pin package

Table 29: Dimensions for 121-pin package

Reference	Minimum	Nominal	Maximum
A	1.54	1.69	1.84
A1	0.47	0.52	0.57
A2	0.42	0.47	0.52
A3	0.65	0.70	0.75
D	11.8	12.0	12.2
D1		10.0 BSC	
D2	11.8	12.0	12.2
E	11.8	12.0	12.2
E1		10.0 BSC	
E2	11.8	12.0	12.2
b	0.6	0.65	0.7
bbb			0.25
ccc			0.35
ddd			0.2
e		1.0 BSC	
f	0.9	1.0	1.1
M		11	
N		121	

a. All dimensions are in mm.

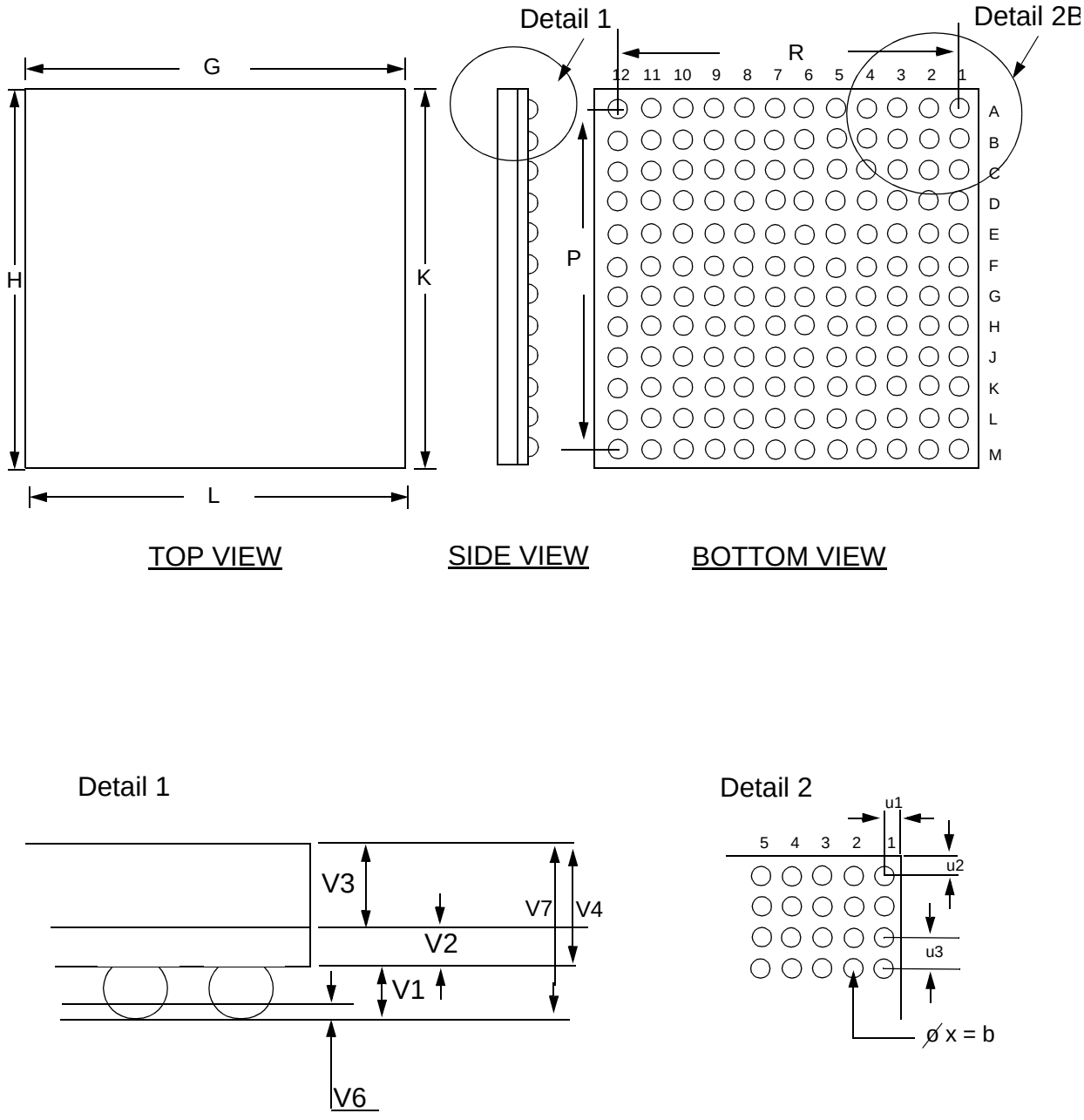


FIGURE 52. Package dimensions for 144-pin package

Table 30: Package dimensions for 144-pin

Reference	Minimum	Nominal	Maximum
V7	1.36	1.71	1.85
V1	0.25	0.5	-
V2			
V3			
G		13	
R		11	
L		13	
K		13	
P		11	
H		13	
x			
V5			
V4	1.11	1.21	1.31
V6		0.2	
u3		1.0	
u1			
u2			
b Reflow ball diameter	0.5	0.6	0.7

package

a. all dimensions in mm.