

## Using the ADV202 in a Multichip Application

by Christine Bako

### INTRODUCTION

This application note provides instructions on how to interface two ADV202s when used in HDTV, 1080i mode.

SMPTE274M 1080i video at 60 fields/sec translates to a total data rate of 1.485 Gbps.

This translates to an input data rate of active video of approximately 124 Mbytes/sec for 10-bit data. The ADV202 limits data input rates to 65 MSPS in irreversible mode or 32 MSPS in reversible mode using the VDATA interface. This requires at least two ADV202s to manage the data input rates for 1080i.

Using two ADV202s in HDTV mode, the ADV202 expects Y and CbCr to be on separate buses: ADV202\_1 is processing the 1080i luma data and the ADV202\_2 is processing the chroma data.

Both data streams have to contain EAV/SAV codes in order to synchronize the two outputs in this mode of application. Multichip applications can be used in decode in a master/slave or slave/slave configuration (see Figure 1 and Figure 2). In encode the ADV202 is always slave.

Limiting an HDTV application (1080i) to two ADV202s requires the following:

Maximum data input rate: 65 MSPS per ADV202 (ADV202-150).

Interface: VDATA bus (for uncompressed video data input or output)

Compression mode: irreversible

If a higher performance is required, for example 1080i lossless compression, it is recommended to use three or more ADV202s. The principles described in this application note also apply to an application that is using three or more ADV202s.

This document applies to applications where the video outputs of the ADV202s are directly connected to a device that requires synchronization on the chroma and luma data inputs (see Recommended Interface section) of a receiving device, such as an encoder or serializer. In cases where the ADV202 video outputs are sent to a buffer or FPGA, synchronization is not important.

## RECOMMENDED INTERFACE

### Decode Master/Slave Configuration

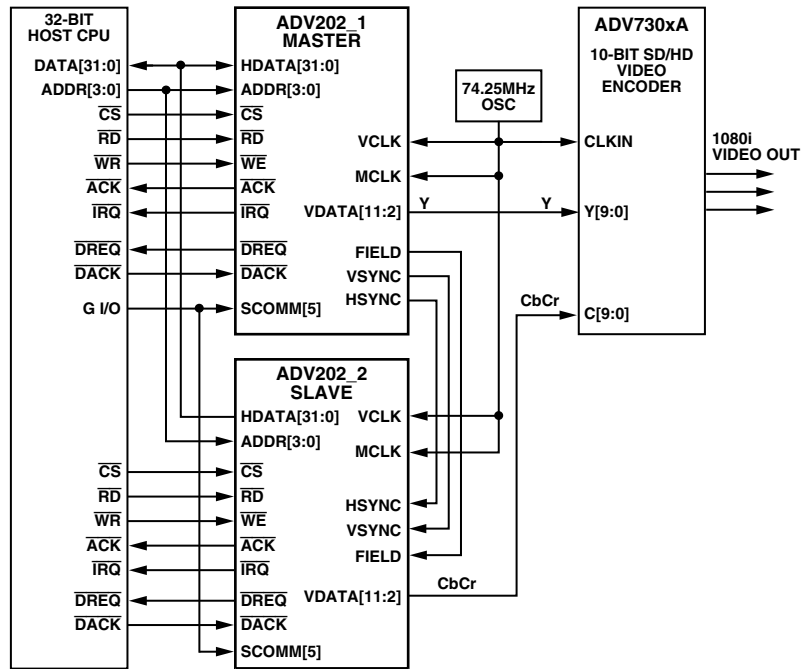


Figure 1. ADV202 in Multichip Application in Decode Master/Slave Configuration

### Decode Slave/Slave Configuration

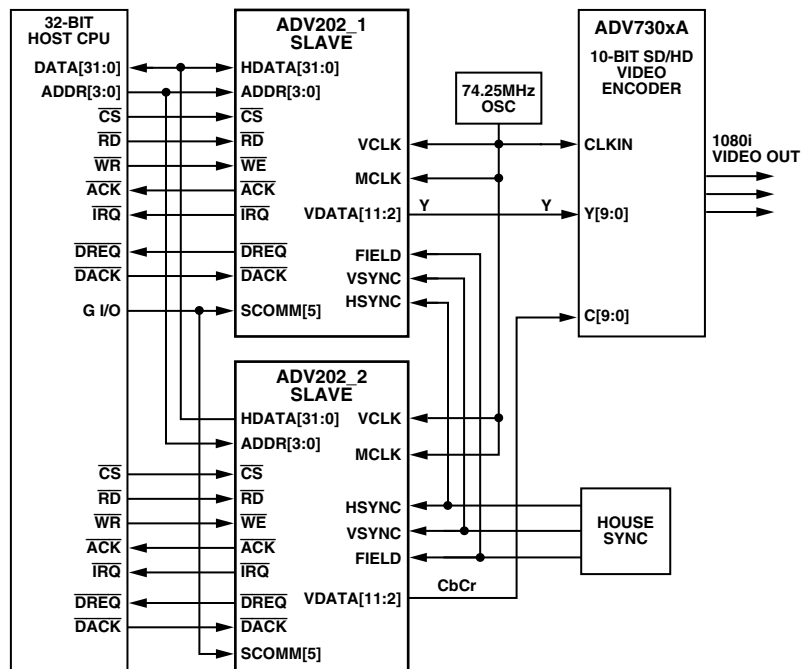


Figure 2. ADV202 in Multichip Application in Decode Slave/Slave Configuration

## Encode Configuration

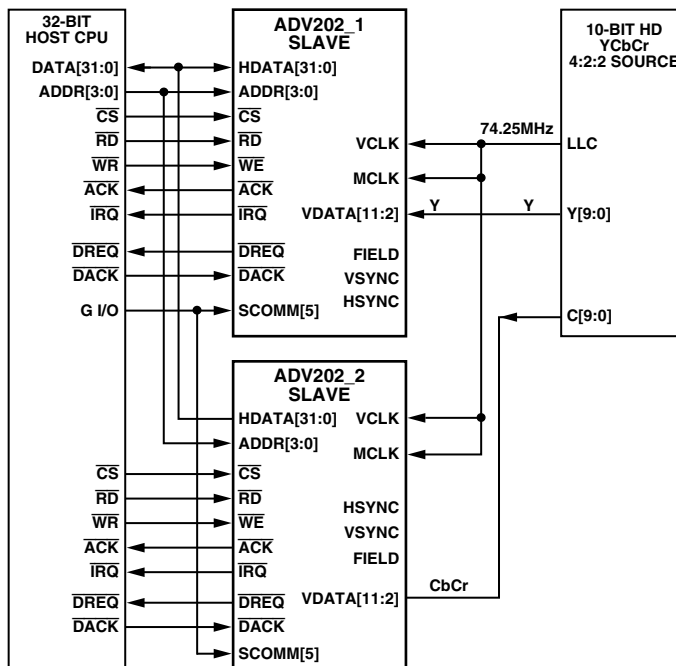


Figure 3. ADV202 in Multichip Application in Encode

### Transfer Rates on HDATA Bus

The highest data rate is achieved over the HDATA bus using the external DMA DREQ/DACK mode in burst transfer configuration.

### Maximum Transfer Rate for 32-Bit Data

A 1080i application requires a VCLK of 74.25 MHz. According to the ADV202 data sheet, JCLK must be at least  $2 \times \text{VCLK}$  since the maximum burst frequency is recommended to be  $0.35 \times \text{JCLK}$ , which is approximately 50 MHz. This is the maximum frequency of read/write pulses.

The CODE FIFO in which the compressed data is to be stored is programmed to a size of no less than 256 32-bit words. For 32-bit wide data, the maximum number of accesses is limited to 256; 512 for 16-bit data.

Therefore, the maximum throughput rate for 32-bit data on the HDATA bus is

$$4 \text{ bytes} \times 50 \text{ MHz} = 200 \text{ MBytes/sec}$$

Assuming a burst length of 128 accesses, if the CODE FIFO does not contain 128 words for the last portion of the compressed field, the FIFO is packed with zeroes to ensure that the field ends on a burst boundary.

The maximum transfer rate for 16-bit data on the HDATA bus is

$$2 \text{ bytes} \times 50 \text{ MHz} = 100 \text{ MBytes/sec}$$

Assuming a burst length of 256 accesses, if the CODE FIFO does not contain 256 words for the last portion of the compressed field, the FIFO is packed with zeros to ensure that the field ends on a burst boundary. It is important that the data throughput rate of the interface used in the

application (for example, a PCI interface to transfer/store compressed data on a PC system) supports these maximum data rates.

### Video Input—Encode Mode

YCbCr data must be in 4:2:2 format and must be accompanied by EAV/SAV timing codes.

The following is an outline on how to program two ADV202s in multichip sync mode using a 32-bit host interface where the 32-bit bus is shared between two ADV202s.

For more information, refer to the current ADV202 data sheet and the Tech Note titled, "Getting Started with the ADV202."

### Programming the ADV202\_1

1. Write 0x0008h to the PLL\_HI register, 0x0084 to the PLL\_LO register (for VCLK = 74.25 MHz).
2. Wait for 20  $\mu\text{s}$  to allow the PLL to settle.
3. Write 0x008A to the BOOT register. This boot mode is used for applications where the firmware has to be loaded into the part.
4. Write 0x000A to BUSMODE. This sets the host control data width to 32 bits and the DMA data width to 32 bits.
5. Write 0x000A to MMODE. This sets the indirect data access width and indirect address step size to 32 bits.
6. Set the start of the program memory in writing 0x00050000 to IADDR.
7. Load the program into the memory by writing every 32-bit value of the firmware to IDATA.

8. Initiate a reboot by writing 0x008D to the BOOT register. This initiates program execution.
9. Write 0x000A to BUSMODE.
10. Write 0x000A to MMODE.

## Pre-Initialization Routine for ADV202\_1

1. Write 0x00057F00 to IADDR. This sets the start address for the encode parameters which will be loaded into the ADV202.

2. Write 0x02010503 to IDATA. These are the actual encode parameters.

In this example

02 = 1080i luminance

01 = 10-bit precision

05 = 5 levels of wavelet transforms

03 = Y, C unipolar

3. Write 0x03000000 to IDATA. Encode parameters continued.

03 = code block size 128 × 32

00 = irreversible 9 × 7 using fixed table

00 = skip no fields

00 = no attribute data output

4. Encoder parameters continued.

Write 0x01019500 to IDATA

01 = target size per video field/frame

019500 = target size value (103,680 bytes/field or 10:1 compression rate for luma).

5. Encoder parameters continued.

Write 0x00000000 to IDATA

00 = LRCP progression style

00 = use EAV/SAV codes, all syncs negative polarity

00 = Qfactor is 1x

01 = code to .j2c

6. Write 0x00000000 to IDATA for all remaining parameter locations.

## Initialization Routine for ADV202\_1

1. Write to 0x0C00 to EIRQIE to unmask SWIRQ0 and SWIRQ1 (address 0x5h). Unmasking SWIRQ1 enables the multichip sync feature.

2. Wait for IRQ to be asserted (going low, SWIRQ0 is set at address 0x6h, Bit 10).

3. Read application ID to ensure the program has correctly initialized. Here: 0xFF82

## Post-Initialization Routine to Configure the DMA Channels for ADV202\_1

1. Write 0xFFFF1408 to IADDR.
2. Write 0x00120000 to IDATA. This configures DMA channel 0 to eight bursts of 32-bit words, assigned to the Compressed/Code block data FIFO.
3. Write 0xFFFF1408 to IADDR.
4. Write 0x00130000 to IDATA.

## Programming the ADV202\_2

1. Write 0x0008h to the PLL\_HI register, 0x0084 to the PLL\_LO register.

2. Wait for 20 μs to allow the PLL to settle.

3. Write 0x008A to the BOOT register. This boot mode is used for applications where the firmware has to be loaded into the part.

4. Write 0x000A to BUSMODE. This sets the host control data width to 32 bits and the DMA data width to 32 bits.

5. Write 0x000A to MMODE. This sets the indirect data access width and indirect address step size to 32 bits.

6. Set the start of the program memory in writing 0x00050000 to IADDR.

7. Load the program into the memory by writing every 32-bit value of the firmware to IDATA.

8. Initiate a reboot by writing 0x008D to the BOOT register.

9. Write 0x000A to BUSMODE.

10. Write 0x000A to MMODE.

## Pre-Initialization Routine of the ADV202\_2

1. Write 0x00057F00 to IADDR. This sets the start address for the encode parameters which will be loaded into the ADV202.

2. Write 0x03010503 to IDATA. These are the actual encode parameters.

In this example

03 = 1080i chrominance

01 = 10-bit precision

05 = 5 levels of wavelet transforms

03 = Y, C unipolar

3. Write 0x03000000 to IDATA. Encode parameters continued.

03 = code block size 128 × 32

00 = irreversible 9 × 7 using fixed table

00 = skip no fields

00 = no attribute data output

4. Write 0x01008700 to IDATA. Encoder parameters continued.

01 = target size per video field/frame

008700 = target size value (34,560 bytes/field or 30:1 compression rate for chroma)

5. Write 0x00000001 to IDATA. Encoder parameters continued.

00 = LRCP progression style

00 = use EAV/SAV codes, all syncs negative polarity

00 = Qfactor is 1x

01 = code to .j2c

6. Write 0x00000000 to IDATA for all remaining parameter locations.

**Initialization Routine for the ADV202\_2**

1. Write to 0x0C00 to EIRQIE to unmask SWIRQ0 and SWIRQ1. Unmasking SWIRQ1 enables the multichip sync feature.
2. Wait for IRQ to be asserted (going low, SWIRQ0 is set at address 0x6h, Bit 10).
3. Read application ID to ensure the program has correctly initialized. Here: 0xFF82

**Post-Initialization Routine to Configure the DMA Channels for the ADV202\_2**

1. Write 0xFFFF1408 to IADDR.
2. Write 0x00120000 to IDATA. This configures DMA Channel 0 to eight bursts of 32-bit words, assigned to the compressed/code block data FIFO.
3. Write 0xFFFF1408 to IADDR.
4. Write 0x00130000 to IDATA.

**Start Program for ADV202\_1**

Write 0x0400 to EIRQFLG (address 0x6h) on the ADV202\_1 to clear the software interrupt (SWIRQ0) and start the program.

**Start Program for ADV202\_2**

Write 0x0400 to EIRQFLG (address 0x6h) on the ADV202\_2 to clear the software interrupt (SWIRQ0) and start the program.

**Data Transfer**

On DREQ0 going active the ADV202 is ready to transmit data from the CODE FIFO.

The host should then initiate a data transfer according to the timing specifications as described in the ADV202 data sheet [Rev 0, Page 13, External DMA Mode—FIFO Read, Burst Mode].

**Encode Mode—Timing**

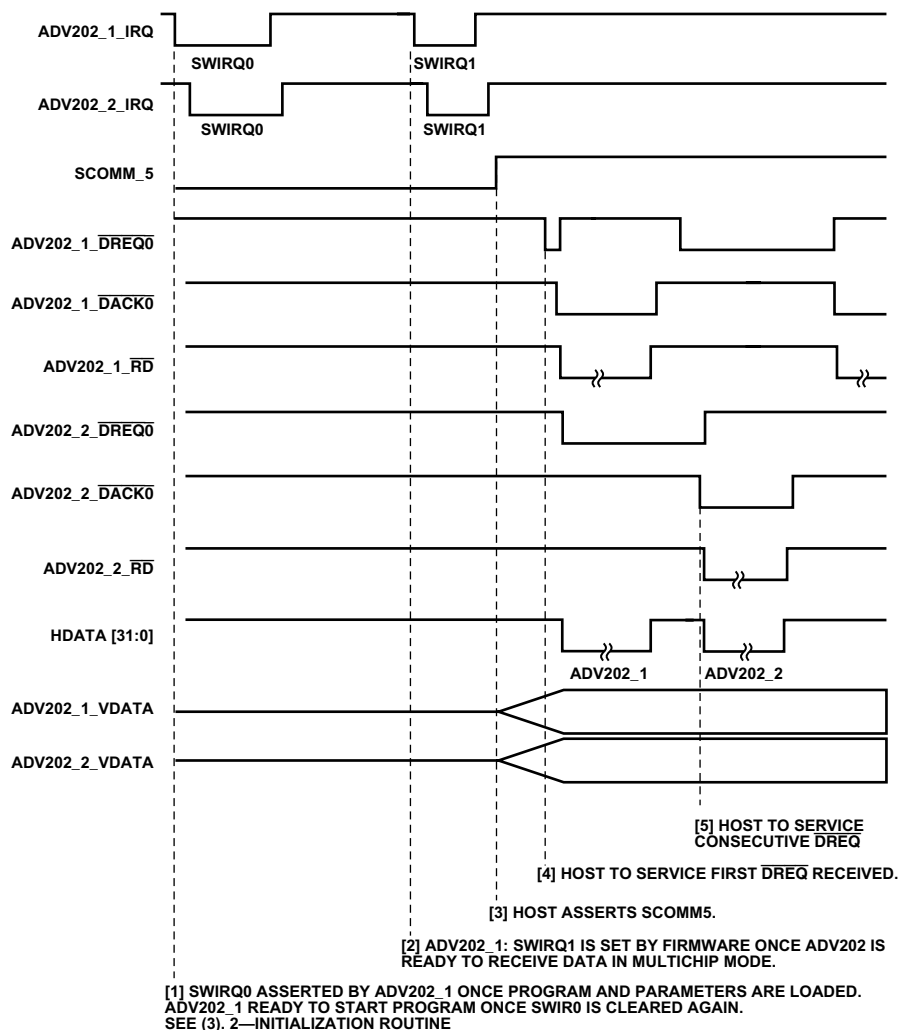


Figure 4. Encode Mode Timing Diagram

In encode mode the data path is VDATA bus—wavelet transform engine/entropy codecs—internal memory—CODE FIFO—HDATA bus. For more detail, see the AN-790 application note.

$\overline{\text{SWIRQ0}}$  is asserted by each ADV202 as soon as the ADV202 is ready to start the program [1] and must be cleared to start the program in writing 0x0400 to address 0x6h. SWIRQ1 is asserted shortly after this [2], indicating that the ADV202 is ready to receive data in multichip mode. The host should poll this bit at address 0x6h. When both SWIRQ1s are set and then cleared, the host should assert SCOMM[5] and keep this pin asserted. On asserting SCOMM[5] the ADV202s will start clocking in video input data [3].

The timing diagram shows the ADV202s configured in  $\overline{\text{DREQ}}/\overline{\text{DACK}}$  DMA burst mode, two ADV202s sharing the 32-bit HDATA bus. Each ADV202s DMA Channel 0 is set to eight accesses ( $8 \times 32\text{-bit words}$ ) and  $\overline{\text{DREQ}}$  is configured to remain asserted until  $\overline{\text{RD}}$  and  $\overline{\text{DACK}}$  are asserted (EDMOD registers).

It can be assumed that both ADV202s will be ready to output compressed data from the CODE FIFO at close to the same time. The host should service the first  $\overline{\text{DREQ0}}$  received and assign the data from each ADV202 to a separate memory location. While  $\overline{\text{DREQ0}}$  from ADV202\_1 is serviced, ADV202\_2 should already have  $\overline{\text{DREQ0}}$  asserted.  $\overline{\text{DREQ0}}$  from the ADV202\_2 will remain asserted until the host asserts  $\overline{\text{DACK}}$  and  $\overline{\text{RD}}$  to the ADV202\_2.

The maximum number of accesses in  $\overline{\text{DREQ}}/\overline{\text{DACK}}$  DMA burst mode using 32-bit accesses is stated to be 256 (see User's Guide, EDMOD registers). In 1080i mode and sharing the 32-bit HDATA bus between two ADV202s, it is recommended to set the number of accesses to  $8 \times 32\text{-bit accesses}$ . This should allow constant data flow from the CODE FIFOs in cases where high compression ratios are used.

## Video Output—Decode Mode

The compressed video data is loaded over the HDATA bus to the ADV202s. The following is an outline on how to program two ADV202s in multichip sync mode using a 32-bit host interface. This procedure is identical for a master/slave or slave/slave application unless otherwise stated. For more information, refer to the current ADV202 data sheet and the Tech Note titled, "Getting Started with the ADV202."

### Programming the ADV202\_1

1. Write 0x0008h to the PLL\_HI register, 0x0084 to the PLL\_LO register.
2. Wait for 20  $\mu\text{s}$  to allow the PLL to settle.
3. Write 0x008A to the BOOT register. This boot mode is used for applications where the firmware has to be loaded into the part.
4. Write 0x000A to BUSMODE to enable. This sets the host control data width to 32 bits and the DMA data width to 32 bits.

5. Write 0x000A to MMODE. This sets the indirect data access width and indirect address step size to 32 bits.
6. Set the start of the program memory in writing 0x00050000 to IADDR.
7. Load the program into the memory by writing every 32-bit value of the firmware to IDATA.
8. Initiate a reboot by writing 0x008D to the BOOT register. This initiates program execution.
9. Write 0x000A to BUSMODE.
10. Write 0x000A to MMODE.

### Pre-Initialization Routine for ADV202\_1

1. Write 0x00057F00 to IADDR. This sets the start address for the decode parameters which will be loaded into the ADV202.
2. Write 0x0201XX03 to IDATA. These are actual decode parameters.

In this example

02 = 1080i luminance

01 = 10-bit precision

XX = this information is derived from the code stream

03 = Y, C unipolar

These values must reflect the values that were used in encode mode.

3. Write 0XXXXXXXX to IDATA. Decode parameters continued. This information is derived from the code stream.
4. Decoder parameters continued.  
Write 0XX0000XX to IDATA for a slave/slave configuration.  
XX = this information is derived from the code stream  
00 = decode slave mode  
00 = decode resolution settings  
XX = this information is derived from the code stream  
  
Write 0XX0010XX to IDATA for a master/slave configuration  
XX = this information is derived from the code stream  
10 = decode master mode  
00 = reserved  
XX = this information is derived from the code stream

### Initialization Routine for ADV202\_1

1. Write 0x0C00 to EIRQIE to unmask SWIRQ0 and SWIRQ1. Unmasking SWIRQ1 enables the multichip sync feature.
2. Wait for IRQ to be asserted (going low).
3. Read application ID to ensure the program has correctly initialized. Here: 0xFFA2.

### Post-Initialization Routine to Configure the DMA Channels for ADV202\_1

1. Write 0xFFFF1408 to IADDR.
2. Write 0x00120000 to IDATA. This configures DMA Channel 0 to eight bursts of 32-bit words, assigned to the compressed/code block data FIFO.
3. Write 0xFFFF1408 to IADDR.
4. Write 0x00130000 to IDATA.

### Programming the ADV202\_2

1. Write 0x0008h to the PLL\_HI register, 0x0084 to the PLL\_LO register.
2. Wait for 20  $\mu$ s to allow the PLL to settle.
3. Write 0x008A to the BOOT register. This boot mode is used for applications where the firmware has to be loaded into the part.
4. Write 0x000A to BUSMODE. This sets the host control data width to 32 bits and the DMA data width to 32 bits.
5. Write 0x000A to MMODE. This sets the indirect data access width and indirect address step size to 32 bits.
6. Set the start of the program memory in writing 0x00050000 to IADDR.
7. Load the program into the memory by writing every 32-bit value of the firmware to IDATA.
8. Initiate a reboot by writing 0x008D to the BOOT register.
9. Write 0x000A to BUSMODE.
10. Write 0x000A to MMODE.

### Pre-Initialization Routine of the ADV202\_2

1. Write 0x00057F00 to IADDR. This sets the start address for the decode parameters which will be loaded into the ADV202.
2. Write 0x0301XX03 to IDATA. These are actual decode parameters.
  - In this example
  - 03 = 1080i chrominance
  - 01 = 10-bit precision
  - XX = this information is derived from the code stream
  - 03 = Y, C unipolar
3. Write 0XXXXXXXX to IDATA. Decode parameters continued. This information is derived from the code stream.

4. Decode parameters continued.
  - Write 0xXX0000XX to IDATA for a slave/slave configuration
  - XX = this information is derived from the code stream
  - 00 = decode slave mode
  - 00 = decode resolution settings
  - XX = this information is derived from the code stream
- Write 0xXX0010XX to IDATA for a master/slave configuration
- XX = this information is derived from the code stream
- 10 = decode master mode
- 00 = reserved
- XX = this information is derived from the code stream

### Initialization Routine for the ADV202\_2

1. Write to 0x0C00 to EIRQIE to unmask SWIRQ0 and SWIRQ1. Unmasking SWIRQ1 enables the multichip sync feature.
2. Wait for IRQ to be asserted (going low).
3. Read application ID to ensure the program has correctly initialized. Here: 0xFFA2

### Post-Initialization Routine to Configure the DMA Channels for the ADV202\_2

1. Write 0xFFFF1408 to IADDR.
2. Write 0x00120000 to IDATA. This configures DMA Channel 0 to eight bursts of 32-bit words, assigned to the compressed/code block data FIFO.
3. Write 0xFFFF1408 to IADDR.
4. Write 0x00130000 to IDATA.

### Start Program for ADV202\_1

Write 0x0400 to EIRQFLG (address 0x6h) on the ADV202\_1 to clear the software interrupt (SWIRQ0) and start the program.

### Start Program for ADV202\_2

Write 0x0400 to EIRQFLG (address 0x6h) on the ADV202\_2 to clear the software interrupt (SWIRQ0) and start the program.

### Data Transfer

On  $\overline{\text{DREQ0}}$  going active the ADV202 is ready to receive data. The host should then initiate a data transfer according to the timing specifications as described in the ADV202 data sheet [Rev 0, Page 12, External DMA Mode—FIFO Write, Burst Mode].

## Decode Mode—Timing

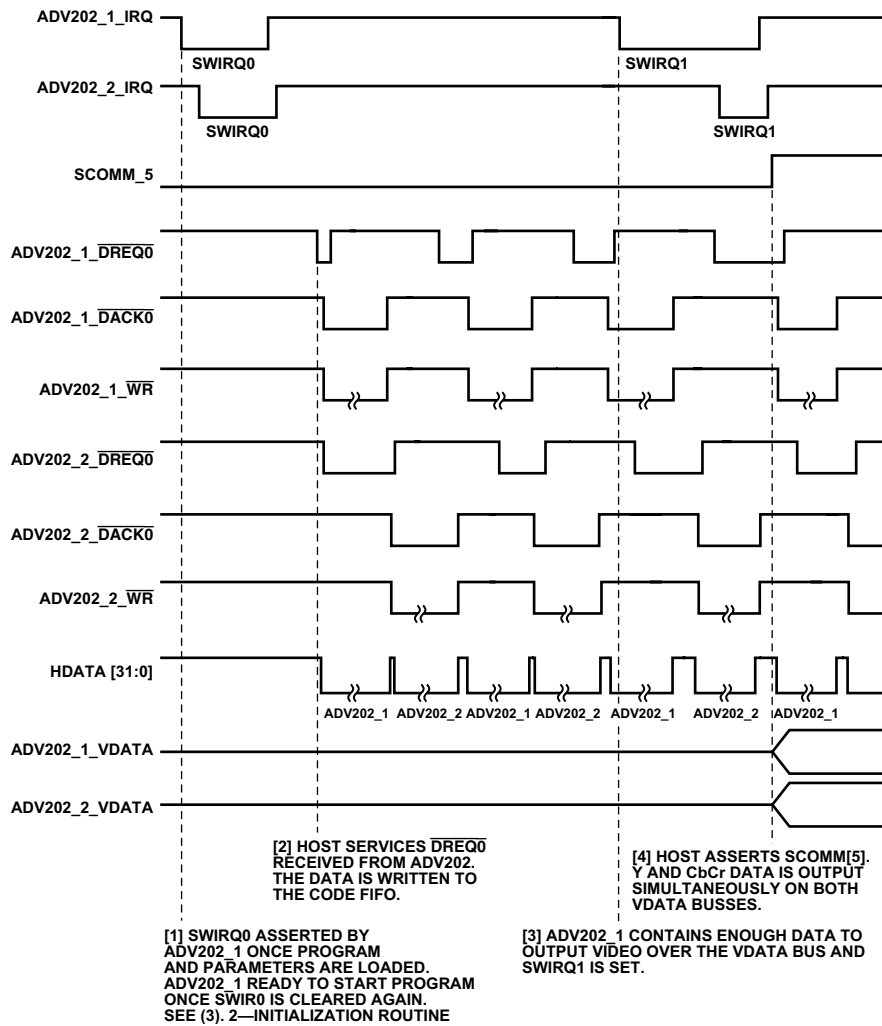


Figure 5. ADV202 Timing Diagram-Decode Mode

In decode mode the data path is HDATA bus—CODE FIFO—wavelet transform engine/entropy codecs—internal memory—pixel interface VDATA bus. See the AN-790 application note.

SWIRQ0 is asserted by the ADV202 as soon as the ADV202 is ready to start the program [1] and must be cleared to start the program in writing 0x0400 to address 0x6h. Both ADV202s will assert DREQ0 at close to the same time. The host should service the first DREQ0 received [2], and alternate servicing DREQ0 from the second ADV202.

The timing diagram shows an example where the ADV202s are configured in DREQ/DACK D12MA burst mode using a shared 32-bit HDATA bus. DREQ is configured to stay asserted until DACK and WR are asserted.

As the CODE FIFOs are filled up and code stream data is passed on to the entropy codecs and wavelet transform engines back to the pixel interface, the ADV202 is ready to output uncompressed video data over the VDATA bus.

At this point SWIRQ1 is asserted [3]. The host should poll this bit at address 0x6h. When both SWIRQ1s are set and then cleared, the host should assert SCOMM[5] and keep this pin asserted [4]. On asserting SCOMM[5] the ADV202s will start to output video on the two VDATA buses [4].

It can be assumed that both ADV202s will be ready to request data at close to the same time. The host should service the first DREQ0 received and write the data to the ADV202 from the external location that has been assigned to this ADV202 previously in encode mode. After servicing DREQ0 from ADV202\_1, ADV202\_2 should already have DREQ0 asserted. DREQ0 from the ADV202\_2 will remain asserted until the host asserts DACK0 and WR to the ADV202\_2. The host should write the data from the external location that has been assigned to ADV202\_2 to the HDATA bus.

It is recommended to set the number of accesses to eight accesses of 32-bit words to guarantee continuous video data output and to avoid possible CODE FIFO overflows.



### Multichip Sync

Synchronization at the inputs (encode mode) is necessary in order to make sure that each ADV202 starts encoding the same field at the same time.

Synchronization at the outputs (decode mode) is only necessary if the separate outputs are sent directly to an HD video encoder/serializer, i.e., to a part which expects all data to be aligned at the input. If another configuration is used, for example sending the Y and CbCr streams to a buffer or FPGA first, synchronization at the ADV202 outputs is not necessary.

There are several factors which determine synchronization at the outputs in an application using two or more ADV202s.

Both ADV202s must output the field after decompression at the same time. This is not guaranteed since each ADV202 takes different times processing each field, and each ADV202 has to start decode on the same field on the first valid sample. Both parts are set into high speed video mode via the firmware, including all indirect register settings from address 0xFFFF0400 to 0xFFFF044C.

In a decode master/slave configuration it is expected that the master H, V, F outputs are connected to the slave H, V, F inputs and that each SCOMM[5] pin is connected to the same GPIO output on the host.

In a decode slave/slave configuration the common HVF for both ADV202s is generated by some external house sync and each SCOMM[5] pin is connected to the same GPIO output on the host. The EAV/SAV timing codes are generated according to the HVF inputs.

In an encode configuration the timing information is derived from the EAV/SAV codes contained in the input data.

SWIRQ1, Software Interrupt 1 in the EIRQIE register must be unmasked on both devices to enable multichip sync in decode or encode mode. The host has to poll both ADV202s for the SWIRQ1 flag to be set in the EIRQFLG register. Only when both SWIRQ1s are active, the host should assert SCOMM[5]. In multichip sync mode, SCOMM[5] has the functionality to initiate output on the VDATA bus when the part is configured in decode mode or initiate data to be clocked in over the VDATA bus in encode mode.

For a decode master/slave configuration the following also has to be considered:

Every slave ADV202 has a fixed timing delay from HSYNC/active input to video data out. By design this is seven CLK cycles on the ADV202. 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 multisync mode (for example, when SWIRQ1 is enabled).

For more information about the ADV202, visit our product page at [www.analog.com](http://www.analog.com).





