OPB Delta-Sigma DAC (v1.01a)

DS487 December 1, 2005 **Product Specification**

Introduction

Digital to analog converters (DACs) convert a binary number into a voltage directly proportional to the value of the binary number. A variety of applications use DACs including waveform generators and programmable voltage sources. The only external circuitry required is a low pass filter comprised of just one resistor and one capacitor.

Features

- Selectable DAC width
- Selectable full scale output
- Programmable interrupt generation
- 16 entry deep data FIFO
- 32 bit OPB slave interface

Discontinued IP

LogiCORE™ Facts		
Core Specifics		
Supported Device Family	QPro™-R Virtex™-II, QPro Virtex-II, Spartan™-II, Spartan-IIE, Spartan-3, Spartan-3E, Virtex, Virtex-II, Virtex-II Pro, Virtex-4, Virtex-E	
Version of Core	opb_deltasigma_dac	v1.01a
Resources Used		
	Min	Max
I/Os	80	92
LUTs	114	156
FFs	81	148
Block RAMs	N/A	N/A
Provided with Core		
Documentation	Product Specification	
Design File Formats	VHDL	
Constraints File	N/A	
Verification	N/A	
Instantiation Template	N/A	
Reference Designs	N/A	
Design Tool Requirements		
Xilinx Implementation Tools	ISE 6.2i or later with latest EDK	
Verification	ModelSim SE 5.8b or later	
Simulation	ModelSim SE 5.8b or later	
Synthesis	XST - ISE 6.2i	
Support		
Support provided by Xilinx, Inc.		

© 2005 Xilinx, Inc. All rights reserved. All Xilinx trademarks, registered trademarks, patents, and further disclaimers are as listed at http://www.xilinx.com/legal.htm. All other trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.
NOTICE OF DISCLAIMER: Xilinx is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Xilinx makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Xilinx expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Functional Description

A Delta-Sigma DAC uses digital techniques to convert a digital number into an analog voltage. Consequently, it is impervious to temperature change, and may be implemented in programmable logic. Delta-Sigma DACs are actually high-speed single-bit DACs. Using digital feedback, a string of pulses is generated. The average duty cycle of the pulse string is proportional to the value of the binary input. The analog signal is created by passing the pulse string through an analog low-pass filter. While an in-depth discussion of Delta-Sigma conversion is beyond the scope of this document, the basic architecture, implementation, and trade-offs are covered.

Delta-Sigma Architecture

Figure 1 is a top-level block diagram of a typical Delta-Sigma DAC implemented in a Virtex FPGA. As shown in this diagram, the inputs include reset and clock signals, in addition to the binary number bus.

DACoutDrvr (Virtex output pin) drives an external low-pass filter. V_{OUT} can be set from 0 V to V_{CCO}, where V_{CCO} is the supply voltage applied to the FPGA I/O bank driving the resistor-capacitor filter.

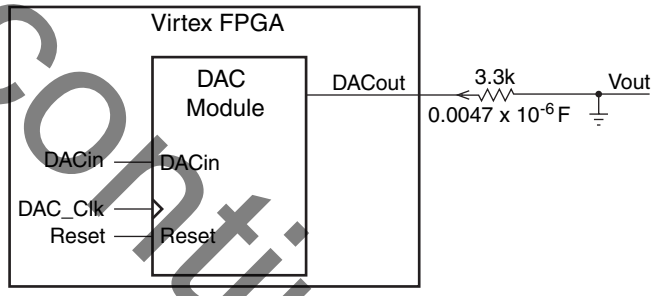


Figure 1: **DAC Module**

The classic current summing digital to analog converter uses matched resistors to convert a binary number to a corresponding voltage level. This technique works well for high-speed DACs when the binary number is up to ten bits wide. However, it is difficult to maintain accuracy over a range of temperatures as the number of bits increases.

Delta-Sigma DACs are used extensively in audio applications. They are suited for low frequency applications that require relatively high accuracy.

As is standard practice, the DAC binary input in this implementation is an unsigned number with zero representing the lowest voltage level. The analog voltage output is also positive only. A zero on the input produces zero volts at the output. All ones on the input cause the output to nearly reach V_{CCO}. For AC signals, the positive bias on the analog signal can be removed with capacitive coupling to the load. Though the low pass filter can be driven with any of the Virtex SelectIO™ output standards that both sink and source current, this data sheet emphasizes the LVTTL standard.

Figure 2 is the block diagram of the OPB Delta-Sigma DAC. The width of the binary input in the implementation described below is configurable. For simplicity, the block diagram depicts a DAC with an 8-bit binary input.

The term “Delta-Sigma” refers to the arithmetic difference and sum, respectively. In this implementation, binary adders are used to create both the difference and the sum. Although the inputs to the Delta adder are unsigned, the outputs of both adders are considered signed numbers.

Discontinued IP

The Delta Adder calculates the difference between the DAC input and the current DAC output, represented as a binary number. Since the DAC output is a single bit, it is "all or nothing"; i.e., either all zeroes or all ones. As shown in Figure 2, the difference will result when adding the input to a value created by concatenating two copies of the most significant bit of the Sigma Latch with all zeros.

This also compensates for the fact that DACin is unsigned. The Sigma Adder sums its previous output, held in Sigma Latch, with the current output of the Delta Adder. In most cases, the Delta adder is optimized out when the high level design is synthesized.

This is because all bits on either the A or B inputs are zero, so A and B are simply merged, rather than added. As noted below, the DAC input can be widened by one bit to allow the full analog range of 0V to VCCO. In this case, the Delta adder is needed.

Discontinued IP

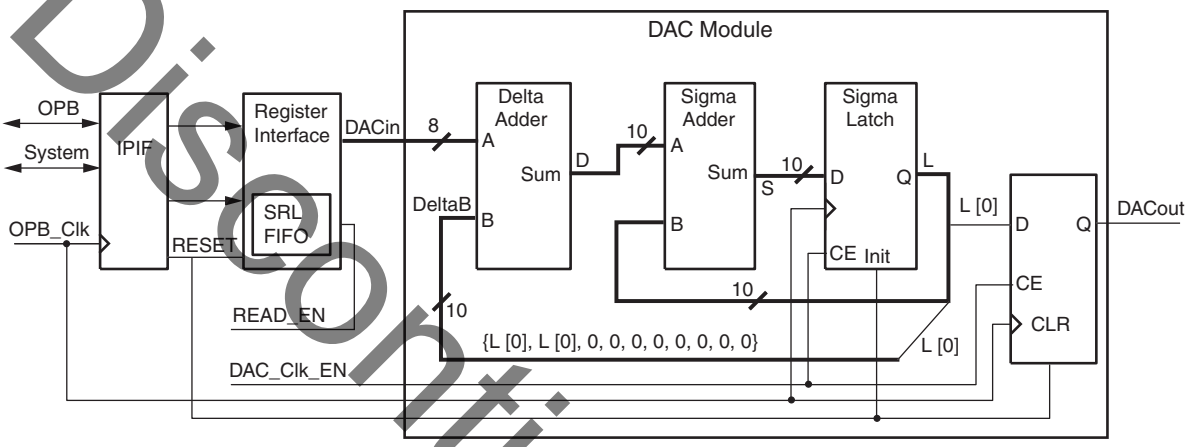


Figure 2: **OPB Delta-Sigma DAC Internal Block Diagram**

For the implementation in Figure1, the output voltage (VOUT) as a function of the DAC input may be expressed as follows:

$$V_{OUT} = (DACin/(2^{(C_NUMDAC_BITS)})) \times V_{CCO} \text{ Volts}$$

For example, for an 8-bit DAC (C_NUM_DAC_BITS = 8) the lowest V_{OUT} is 0 V when DACin is 0x00. The highest V_{OUT} is 255/256*V_{CCO} volts when DACin is 0xFF.

For some applications, it may be important for V_{OUT} to swing through the entire voltage range: 0 V to V_{CCO} (rail-to-rail). This is accomplished by using the C_FULL_RANGE generic which increases the DACin bus width by one bit and leaving all other bus widths the same. For an 8-bit DAC with an input value of 256, $V_{OUT} = V_{CCO}$. Note for C_NUM_DAC_BITS = 8 that all DAC in values greater than 256 are illegal and should not be used. The valid range of digital inputs for given values of C_NUM_DAC_BITS and C_FULL_RANGE is given as 0x0 to 0x($2^{(C_NUMDAC_BITS)}$ -1 + C_FULL_RANGE), where $2^{(C_NUMDAC_BITS)}$ is always greater than or equal to ($2^{(C_NUMDAC_BITS)}$ -1 + C_FULL_RANGE).

As outlined in this specification, it is often advantageous to use a high-frequency clock. The desired clock may be faster than that which can be practically sourced externally. Virtex DCMs may be used to create the desired clock frequency.

Low-Pass Filter

The resistor/capacitor low-pass filter shown in Figure 1 is adequate for most applications. A 24 mA LVTTL output buffer is used to provide maximum current drive.

There are three primary considerations in choosing values for the resistor and capacitor:

- Output Source and Sink Current: Unlike normal digital applications, it is important that signal DACout always switch the entire voltage range from 0 V to V_{CCO} (rail-to-rail). If the value of R is too low and signal DACout can not switch rail-to-rail, the analog output is non-linear; i.e., the absolute output voltage change resulting from incrementing or decrementing DACin is not constant. The worst-case output impedance of the 24 mA LVTTL buffer is about 25 W. R must be 2.5 KW or greater to ensure rail-to-rail switching, with an error of 1% or less.
- Load Impedance: Keep the value of R low relative to the impedance of the load so that the current change through the capacitor due to loading becomes negligible.
- Time Constant: The filter time constant (τ = RC) must be high enough to greatly attenuate the individual pulses in the pulse string. On the other hand, a high time constant may also attenuate the desired low-frequency output signal. These potentially conflicting requirements are analyzed separately.

Discontinued IP

Pulse String Filtering

In the midrange voltages, signal DACout is switching rapidly, making it relatively easy to filter. When the DAC input is at 1 or the highest possible value, the signal DACoutDrvr is at the same level for all but one CLK cycle for each sample period. These are the most difficult output strings to filter; i.e., they are the "worst-case".

Although the filter noise may be calculated as an absolute peak-to-peak voltage, it is more useful to consider it as a fraction of the step voltage. The step voltage (V_S) is defined as the absolute change in V_{OUT} when the DAC input is incremented or decremented. For an 8-bit DAC, V_S equals (1/256) x V_{CCO}.

The worst-case peak-to-peak filter noise for an 8-bit DAC can be expressed as follows:

$$PPN_{FS} = (1-e^{-(1/f\tau)}) \times ((1-e^{-(255/f\tau)})/(1-e^{-(256/f\tau)})) \times 256$$

where:

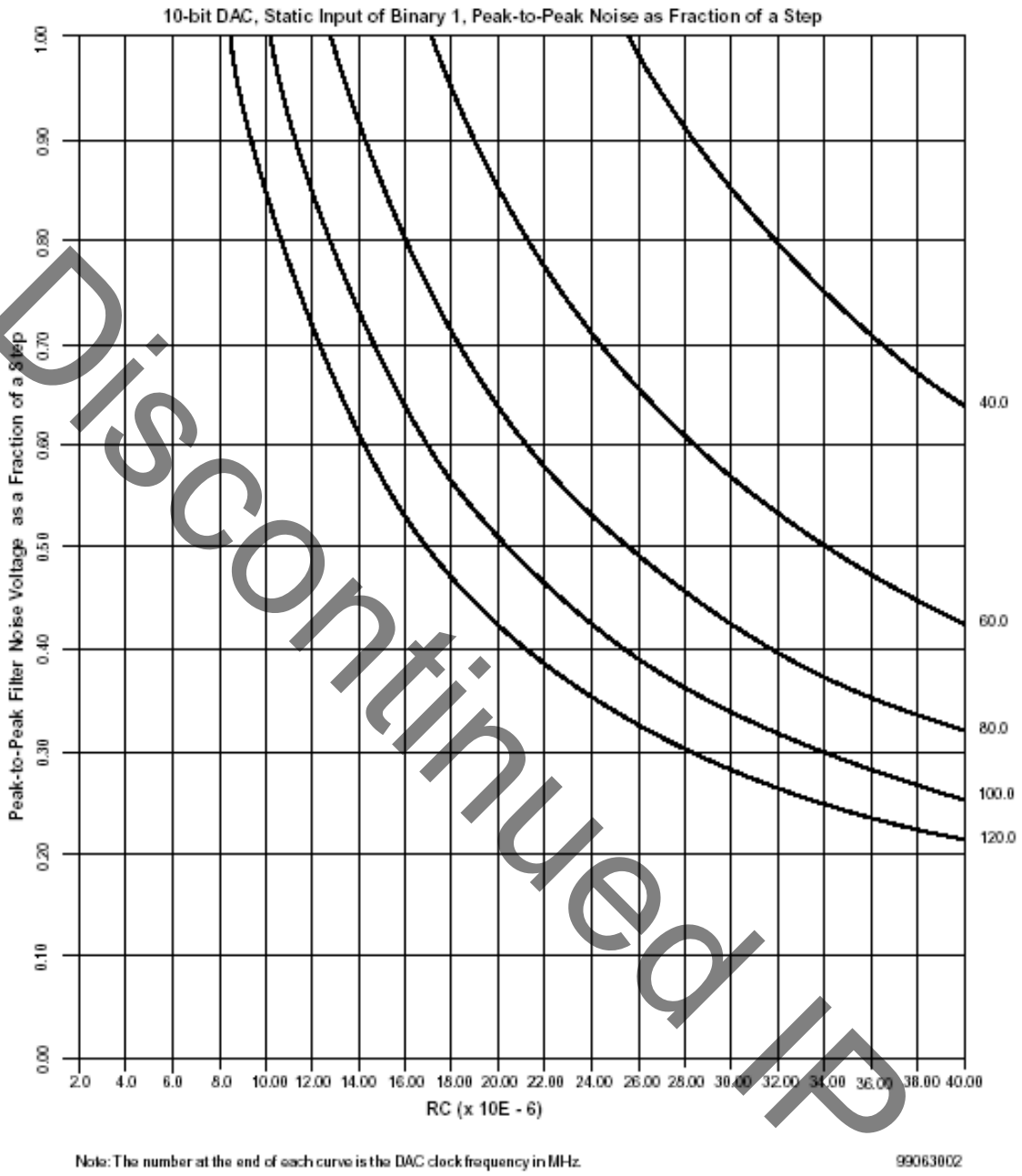
PPN_{FS} is peak-to-peak noise expressed as a fraction of step voltage

f is the DAC clock frequency

τ is the filter time constant, RC.

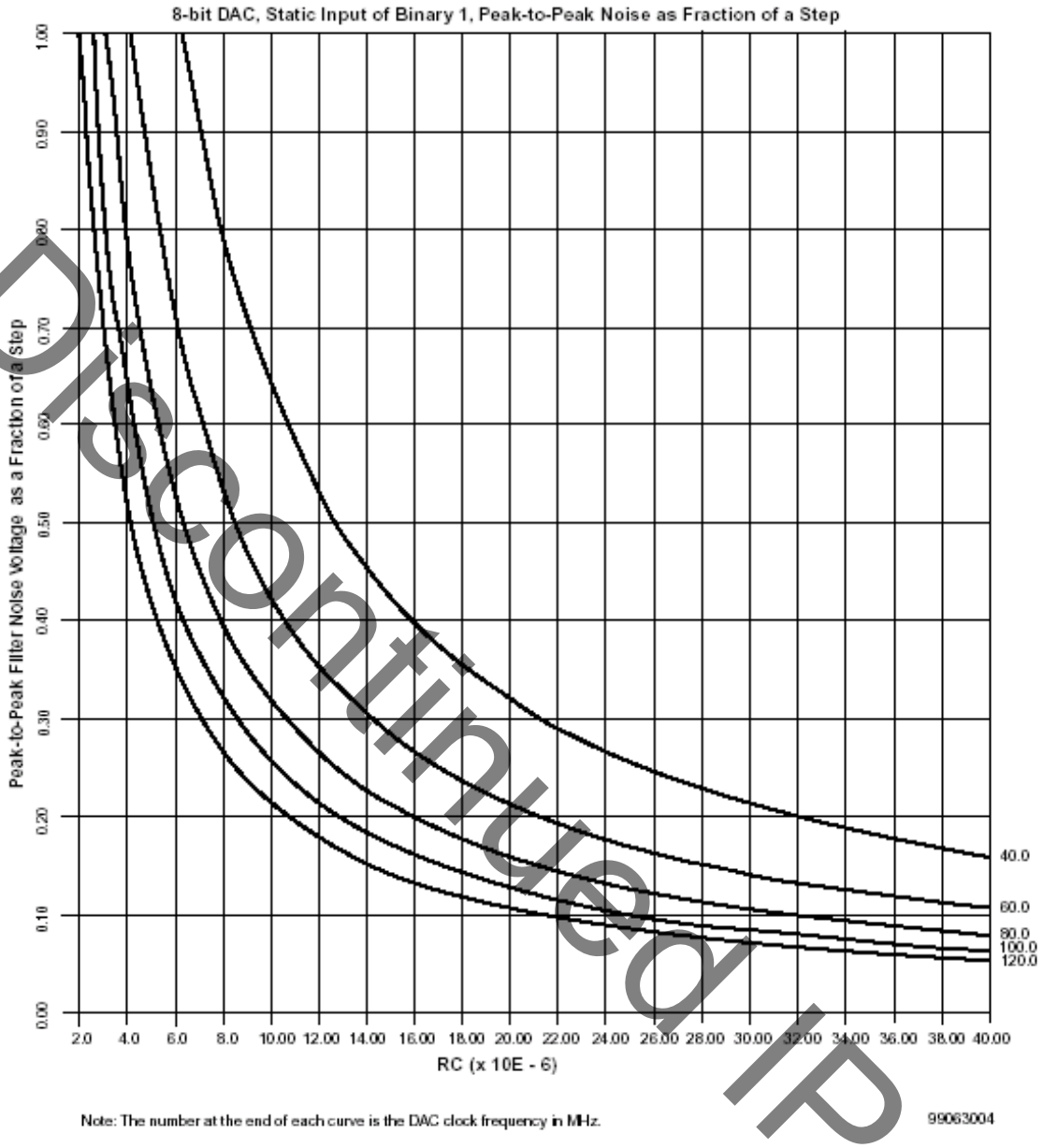
For simplicity, we did not generalize this equation to handle any width DAC. For other widths, change the constants 256 and 255 to the appropriate power of 2, and (power of 2) minus 1, respectively.

This equation was used to create Figure 3, Figure 4, and Figure 5. These charts may be used to determine the value of RC for the desired worst-case noise voltage and operating frequency. For example, for an 8-bit DAC with a clock frequency of 80 MHz, the user might choose an RC value of 13.0×10^{-6}, corresponding to a peak-to-peak noise voltage of about 0.25 V_S. This leaves 0.75 V_S noise margin between steps. Part of this noise margin is needed to handle other noise sources such as noise on V_{CCO}.



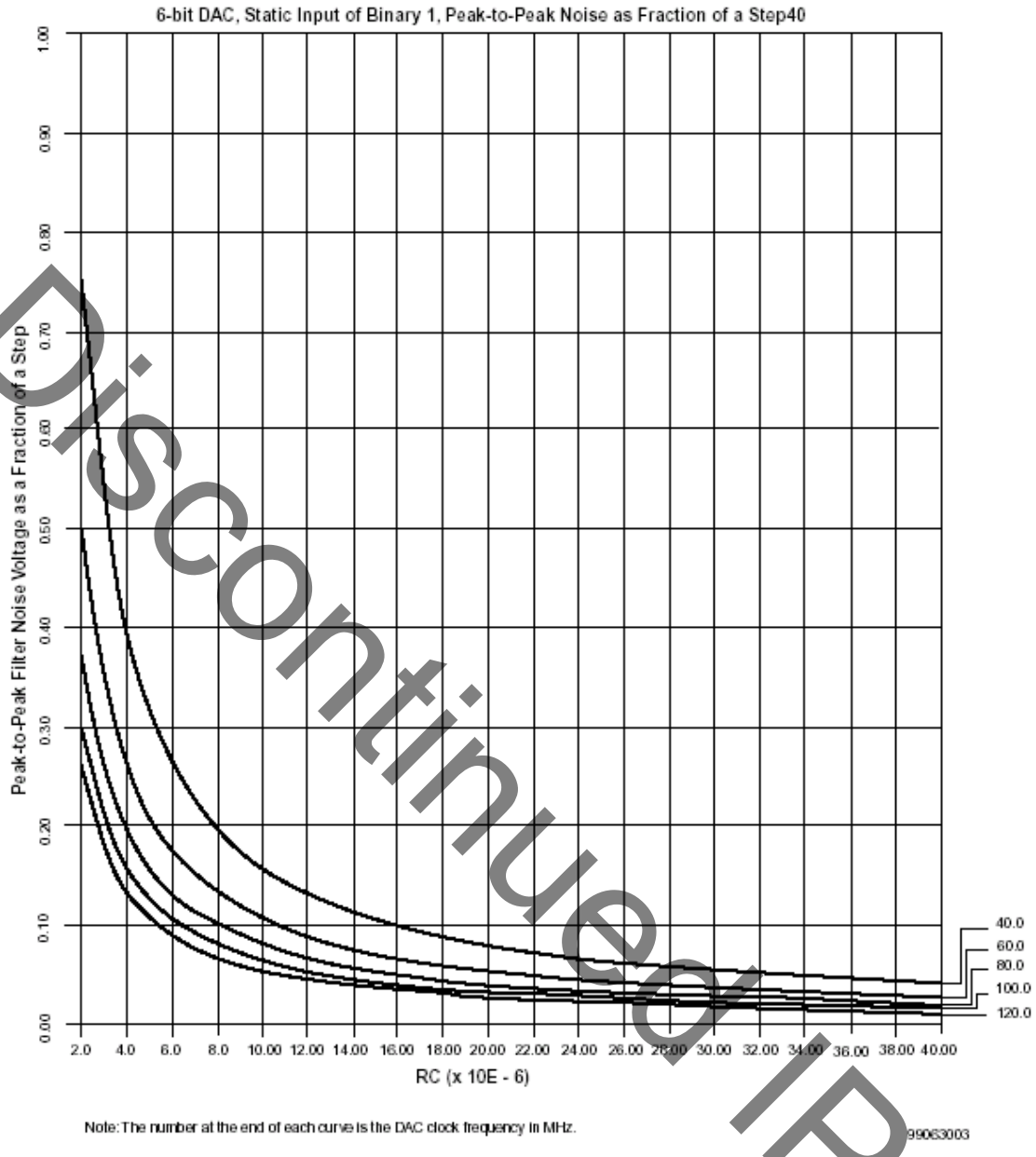
Discontinued IP

Figure 3: **Peak-to-Peak Noise as a Function of RC and Frequency (10-bit DAC)**



Discontinued IP

Figure 4: **Peak-to Peak Filter Noise as a Function of RC and Frequency (8-bit DAC)**



Discontinued IP

Figure 5: **Peak-to Peak Filter Noise as a Function of RC and Frequency (6-bit DAC)**

Output Attenuation

By convention, the cutoff frequency of a low pass filter is defined as the half-power point. The cutoff frequency of the simple RC filter may be expressed as:

$$f_C = 1/(2\pi\tau)$$

where:

f_C is the filter cutoff frequency

τ is the filter time constant, RC.

The above equation was used to create Figure 6.

Figure 6 may be used in conjunction with Figure 3, Figure 4, or Figure 5 to choose the RC time constant that is optimum for a particular application. All figures cover the same RC range.

Figure 4 shows an RC value of 13.0×10^{-6} results in a peak-to-peak noise voltage of 0.25 V when a DAC clock frequency of 80 MHz is used on a 8-bit DAC. From Figure 6, it can be determined that the filter cutoff frequency for this RC value is about 12 KHz. If the expected output is essentially a DC level, e.g., a programmable voltage generator, then RC may be increased to reduce the clock noise. On the other hand, if the fundamental frequency of the analog output is high, or it has sharp edges, then a lower RC may be needed. When determining the actual component values, remember that R should be at least 2500 Ω.

The user may implement a more sophisticated filter if the simple RC filter has inadequate cutoff or drive characteristics for the application.

Discontinued IP

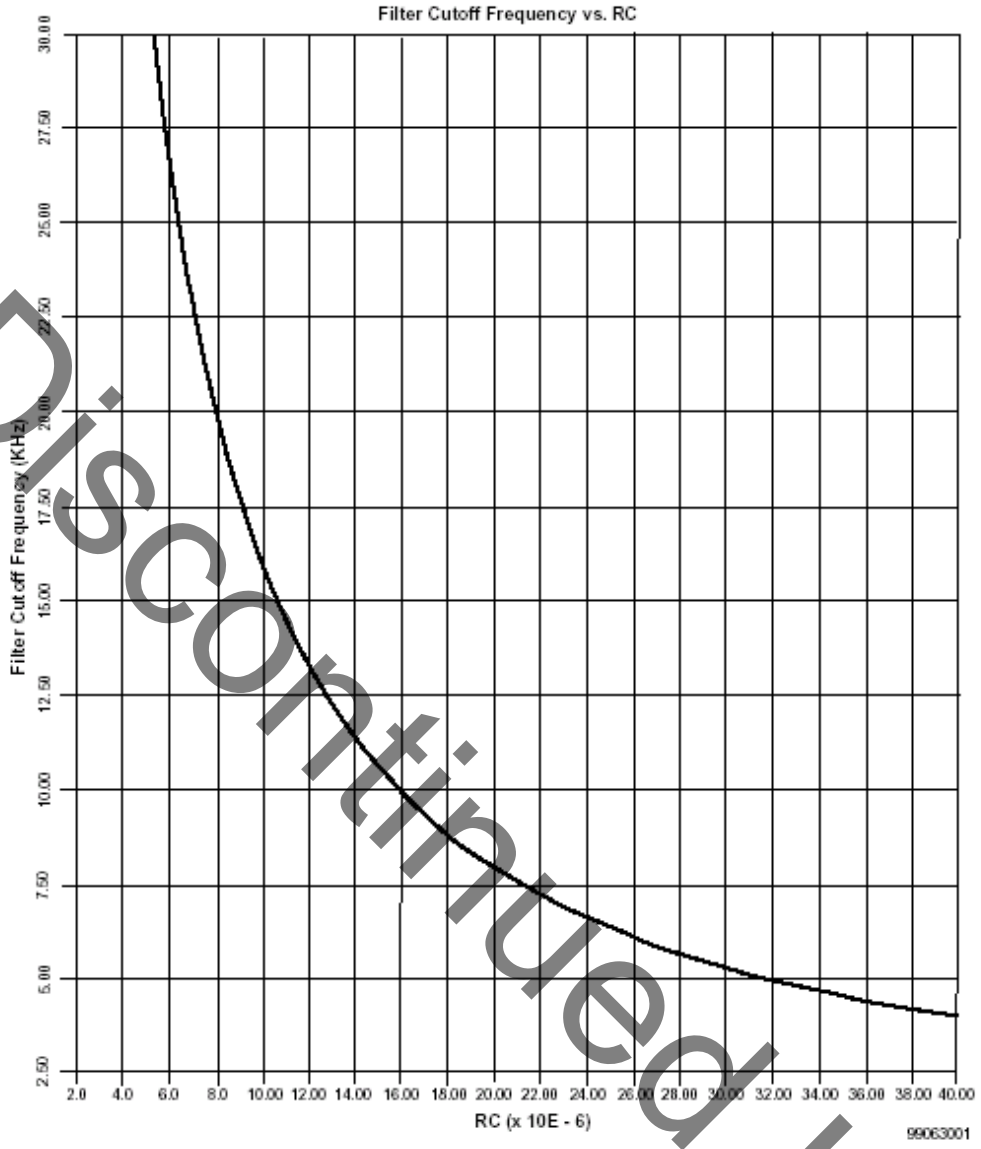


Figure 6: **Filter Cutoff Frequency as a Function of RC**

Discontinued IP

Sampling Rate

To resolve each DACin sample to the full precision of a Delta-Sigma DAC, the sample rate, i.e., the rate that DACin changes, must be less than or equal to $1/(2^{(C_NUM_DAC_BITS)})$ of the CLK frequency. In some applications, such as a programmable voltage source, this is not an issue.

As DAC width and the desired sample frequency increases, it may not be possible to meet the above criterion. In practice, the sample rate sometimes exceeds $1/(2^{(C_NUM_DAC_BITS)})$ of the CLK frequency. Though this compromises precision at higher frequencies, it is often possible to get satisfactory results. For example, the16-bit audio DACs in a CD system would require a clock frequency of 2.9 GHz for full resolution of the highest frequencies. In practice, a much lower clock frequency is used. One reason this

is acceptable is because the sensitivity of the human ear to noise becomes lower as the frequency increases.

This section lists some of the ways this DAC can be used in real-world applications.

- **Programmable Voltage Generator**: A variable voltage between 0 V and V_{CCO} can be generated with a granularity determined by the bus width of DACin. In these applications, the voltage typically does not change quickly, so RC may be large to minimize noise.
- **Virtex VREF Generator**: This is a specific application of a Programmable Voltage Generator. For some Virtex SelectIO receivers standards, a reference voltage is required for each bank of receivers. If a DAC is used to generate this voltage, V_{REF} can be dynamically changed to verify operating margins when conducting system tests. See XAPP133 for more information on SelectIO.
- **Waveform Generator**: Various analog waveforms, such as sine, sawtooth, triangle, etc., can be created by sequentially feeding the proper values to DACin. The values are normally pre-stored in SRAM. Virtex Block SelectRAM+ is ideal for this purpose. See XAPP130 for more information on Block SelectRAM+.
- **Sound Generator:** Delta-Sigma DACs are widely used in sound reproduction, speech synthesis, etc. Since the analog output is changing rapidly, RC must be chosen with an acceptable trade-off between noise and frequency response.
- **RGB Color Generator**: Although Delta-Sigma DACs are too slow to directly generate Red-Green-Blue signals for a raster display, they are applicable in some color generation systems that do not operate in real time.
- **Analog to Digital Conversion:** This DAC may be used as a voltage reference in an ADC. See XAPP155 for a complete discussion of this application.

Discontinued IP

OPB Delta-Sigma DAC I/O Signals

The I/O signals for the OPB Delta-Sigma DAC are listed in Table 1.

Table 1: **OPB Delta-Sigma DAC I/O Signals**

Port	Signal Name	Interface	I/O	Initial State	Description
DAC Signals					
P1	Dac_clk_en	System	I		DAC clock enable. This allows the OPB_Clk to clock the sigma latch and the D flip-flop.
P2	Read_en	System	I		Read enable. A new value is read from the FIFO when this input is high.
P3	Dac_Out	System	O	0	DAC output. This is the pulse string that drives the external low pass filter.
OPB Signals					
P10	Sln_DBus(0:C_OPB_DWIDTH-1)	IPIF	O	0	DAC output data bus
P11	Sln_xferAck	IPIF	O	0	DAC transfer acknowledge
P12	Sln_Retry	IPIF	O	0	DAC retry
P13	Sln_ToutSup	IPIF	O	0	DAC timeout suppress

Table 1: **OPB Delta-Sigma DAC I/O Signals *(Contd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P14	Sln_ErrAck	IPIF	O	0	DAC error acknowledge
P15	OPB_ABus(0:C_OPB_AWIDTH-1)	IPIF	I		OPB address bus
P16	OPB_BE(0:C_OPB_DWIDTH/8-1)	IPIF	I		OPB byte enables
P17	OPB_DBus(0:C_OPB_DWIDTH-1)	IPIF	I		OPB data bus
P18	OPB_RNW	IPIF	I		Read not Write (OR of all master RNW signals)
P19	OPB_select	IPIF	I		Master has taken control of the bus (OR of all master selects)
P20	OPB_seqAddr	IPIF	I		OPB sequential address
System					
P30	OPB_Clk	System	I		System clock
P30	OPB_Rst	System	I		System Reset (active high)
P32	IP2INTC_Irpt	System	O	0	System Interrupt
P33	Freeze	System	I		System Freeze Input (Unconnected in OPB Delta-Sigma DAC)

Discontinued IP

Delta-Sigma DAC Design Parameters

The OPB Delta-Sigma DAC design is user configurable via a set of design parameters implemented as VHDL input generics. The generics for the OPB Delta-Sigma DAC are listed in Table 2.

Table 2: **OPB Delta-Sigma DAC Design Parameters**

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
OPB Delta-Sigma DAC Features					
G1	Number of DAC bits	C_NUM_DAC_BITS	2 to 16	8	Integer
G2	Allow the DAC output to go to full scale(1)	C_FULL_RANGE	1 = The DAC output will go to full scale 0 = The DAC output will be 1 LSB less than full scale(1)	0	Integer
OPB/IPIF Interface					
G21	Device Block ID(5)	C_DEV_BLK_ID	See note 5.	0	Integer
G22	Module Identification Register Enable(5)	C_DEV_MIR_ENABLE	See note 5.	0	Integer

Table 2: **OPB Delta-Sigma DAC Design Parameters *(Contd)***

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G23	OPB High Address	C_HIGHADDR	Address range must be 2^n - 1 and greater than or equal to 0x1FF(4)	None(2,3)	std_logic_vector
G24	Location of the first IPIF register.	C_BASEADDR	Valid Address Range(4).	None(2,3)	std_logic_vector
G25	OPB Address Bus Width(5)	C_OPB_AWIDTH	16 - 32	32	integer
G26	OPB Data Bus Width(5)	C_OPB_DWIDTH	8,16,32,64,128	32	integer

Notes:
1. With C_FULL_RANGE set to 0, the DAC output will be $(2^n - 1)/2^n$. Where n is the number of DAC bits, C_NUM_DAC_BITS.
2. No default value will be specified to insure that the actual value is set, i.e. if the value is not set, a compiler error will be generated.
3. For example, C_BASEADDR = 0xE0000000, C_HIGHADDR = 0xE00001FF
4. Address range specified by C_BASEADDR and C_HIGHADDR must be at least 0X200 and must be a power of 2.
5. See the Processor IP Reference Guide under Part 1: Embedded Processor IP, under IPIF, under OPB IPIF Architecture.

Allowable Parameter Combinations

The address range specified by C_BASEADDR and C_HIGHADDR must be a power of 2, and C_HIGHADDR must be at least 0x1FF. For example, if C_BASEADDR = 0xE0000000 C_HIGHADDR must be at least = 0xE00001FF.

OPB Delta-Sigma DAC Port Dependencies

The width of some of the OPB Delta-Sigma DAC signals depends on parameters selected in the design. The dependencies between the OPB Delta-Sigma DAC design parameters and I/O signals are shown in Table 3.

Table 3: **OPB Delta-Sigma DAC Parameter Port Dependencies**

Generic	Name	Affects	Depends	Relationship Description
Design Parameters				
G26	C_OPB_DWIDTH	P10, P16, P17		Specifies the OPB Data Bus width
G25	C_OPB_AWIDTH	P15		Specifies the OPB Address Bus width
G23	C_HIGHADDR		G24	Specifies the OPB High address range
I/O Signals				
P10	Sln_DBus(0:C_OPB_DWIDTH-1)		G26	Width varies with the size of the OPB Data bus.

Discontinued IP

Table 3: **OPB Delta-Sigma DAC Parameter Port Dependencies**

Generic	Name	Affects	Depends	Relationship Description
P15	OPB_ABus(0:C_OPB_AWIDTH-1)		G25	Width varies with the size of the OPB Address bus.
P16	OPB_BE(0:C_OPB_DWIDTH/8-1)		G26	Width varies with the size of the OPB Data bus.
P17	OPB_DBus(0:C_OPB_DWIDTH-1)		G26	Width varies with the size of the OPB Data bus.

OPB Delta-Sigma DAC Register Descriptions

The OPB Delta-Sigma DAC contains addressable registers for read/write operations as shown in Table 4. The base address for these registers is set in the parameter C_BASEADDR. C_BASEADDR + 0x100 represents the address of the first register in the OPB Delta-Sigma DAC, the OPB Delta-Sigma DAC Control Register. The address of each register is then calculated by an offset to the base address.

The IPIF contains the interrupt registers. The base address for these registers is set by the parameter C_BASEADDR which represents the address of the first register - in this case, the ISR. The ISR is not used by the OPB Delta-Sigma DAC design, the location of the first accessible IPIF register is the GIE, offset of 0x1C. The address of each IPIF register is then calculated by an offset to the base address.

Table 4 shows all of the OPB Delta-Sigma DAC registers and the IPIF interrupt registers and their addresses.

Table 4: **OPB Delta-Sigma DAC Registers**

Register Name	OPB Address	Access
Device Global Interrupt Enable Register (GIE) (1)	C_BASEADDR + **0x01C**	Read/Write
IP Interrupt Status Register (IPISR)(1, 2)	C_BASEADDR + **0x020**	Read/Write
IP Interrupt Enable Register (IPIER)(1,2)	C_BASEADDR + **0x028**	Read/Write
IPIF Software Reset Register (IPSRR)(1)	C_BASEADDR + **0x040**	Write Only
Control Register (CR)	C_BASEADDR + **0x100**	Read/Write
Data FIFO (FIFO)	C_BASEADDR + **0x104**	Read/Write
Data FIFO Occupancy (OCCY)	C_BASEADDR + **0x108**	Read
Data FIFO programmable depth interrupt Register (PIRQ)	C_BASEADDR + **0x10C**	Read/Write

Notes:
1. See the *Processor IP Reference Guide* for a complete description of these registers.
2. The bit mapping for these registers is provided in Figure 11.

Discontinued IP

Control Register (cr_i)

The enable bit (EN) when set to '0' will prevent the OPB Delta-Sigma DAC from creating a pulse string, and the Dac_out will be zero. See Table 5.

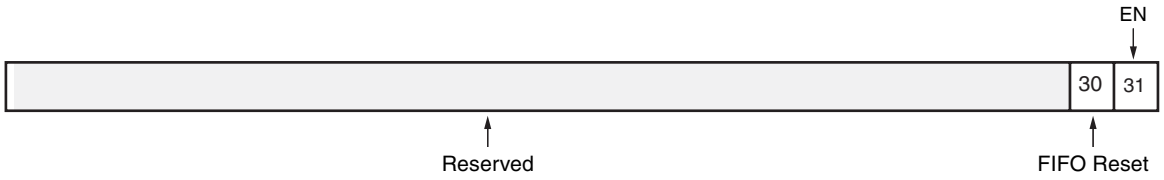


Figure 7: **OPB Delta-Sigma DAC Control Register**

Table 5: **OPB Delta-Sigma DAC Control Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0- 29	Reserved			**Reserved.**
30	FIFO_RST	Read/Write	0	FIFO Reset. "1" resets the Data FIFO. "0" Data FIFO normal operation.
31	EN	Read/Write	0	**DAC Enable.** "1" enables the OPB Delta-Sigma DAC. "0" resets and disables the OPB Delta-Sigma DAC. The DAC output will be zero while in reset.

Data FIFO

This 16 entry deep SRL FIFO contains data to be output by the OPB Delta-Sigma DAC. The Data FIFO is shown in Table 6. Reading of this location will result in reading the current word being output from the FIFO. Attempting to write to a full FIFO is not recommended and results in that data byte being lost.

When the Data FIFO is empty and the DAC is enabled, the DAC output will be zero.

Figure 8 shows the location for data on the OPB when C_NUM_DAC_BITS is set to 8 and C_FULL_RANGE is set to 0. If C_FULL_RANGE is set to 1 and C_NUM_DAC_BITS is set to 8 then data bits 23 through 31 will be used, but any value greater than 0X100 will result in an undefined DAC output.

Figure 8: **Data FIFO**

Discontinued IP

Table 6: **OPB Delta-Sigma DAC Data FIFO Bit Definitions**

Bit(s)	Access	Reset Value	Description
0 thru (31 - C_NUM_DAC_BITS - C_FULL_RANGE)			Reserved
(32 - C_NUM_DAC_BITS - C_FULL_RANGE) thru 31	Read/Write	Indeterminate(1)	DAC Output Data

Notes:
1. The value that was available before the reset occurred will still appear on the FIFO outputs

Data FIFO Occupancy Register (OCCY)

This field contains the occupancy value for the Data FIFO. Reading this register can be used to determine if the FIFO is empty, also the Data FIFO Empty Interrupt conveys that information. The value read is the binary count value, therefore reading all zeros implies that no location is filled and reading 10000 implies that all sixteen locations are filled. See Table 7.

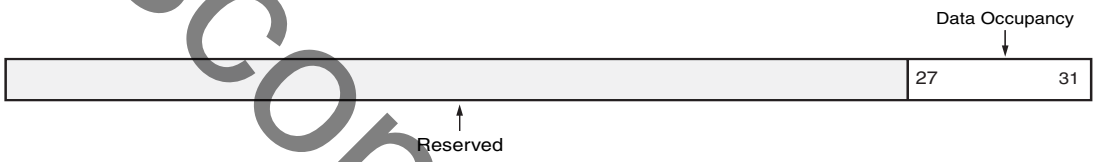


Figure 9: **Data FIFO Occupancy Register (OCCY)**

Table 7: **Data FIFO Occupancy Register Bit Definitions**

Bit(s)	Name	Access	Reset Value	Description
0 - 26	Reserved			Reserved.
27 -31	Occupancy Value	Read	0x0	Bit 27 is the MSB. A value of 10000 implies that all 16 locations in the FIFO are full.

Data FIFO Programmable Depth Interrupt Register (PIRQ)

This field contains the value which will cause the PIRQ Interrupt to be set. When this value is greater than or equal to the OCCY value, the PIRQ interrupt will be set and remain set until the equality or greater than is no longer true. See Table 8.

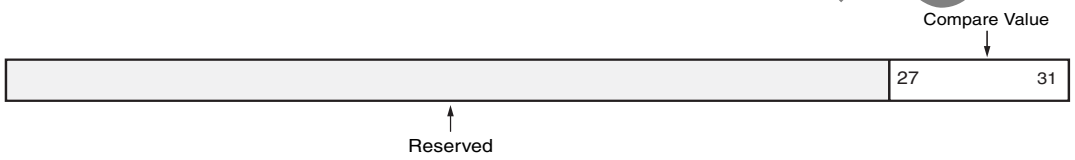


Figure 10: **Data FIFO Programmable Depth Interrupt Register (Dt_FIFO_PIRQ)**

Discontinued IP

Table 8: **Data FIFO Programmable Depth Interrupt Register Bit Definition**

Bit(s)	Name	Access	Reset Value	Description
0 - 26	Reserved			
27 - 31	Compare Value	Read/Write	0x0	Bit 27 is the MSB. A value of 00101 implies when either five or less than five locations in the Data FIFO are filled, FIFO PIRQ interrupt will be set.

OPB Delta-Sigma DAC Interrupt Descriptions

Interrupts

The interrupt signals generated by the OPB Delta-Sigma DAC are managed by the Interrupt Source Controller in the IPIF. This interface provides many of the features commonly provided for interrupt handling. The IPIER and IPISR contain the bit mapping as shown in Figure 11. Please refer to the *Processor IP Reference Guide* under Part 1 for a complete description of the GIE, IPISR and IPIER. The OPB Delta-Sigma DAC has two unique interrupts that are sent to the IPIF. The number in the parenthesis is the IPIF interrupt bit number.

Figure 11: **Interrupt Mapping**

Table 9: **Data FIFO Interrupt Register Bit Definitions**

Bit(s)	Name	Access	Reset Value	Description
0 - 29	Reserved			Reserved.
30	FIFO EMPTY	Read/Write	0	This interrupt will be set and remain set as long the Data FIFO is empty. Clearing this interrupt requires that the Data FIFO be written
31	FIFO PIRQ	Read/Write	0	This interrupt will be set and remain set as long the PIRQ is equal to, or greater than the OCCY. Clearing this interrupt requires that the Data FIFO be filled to a value greater than PIRQ.

Discontinued IP

Flow Description

The following is a brief discussion on setting the DAC registers to initiate a conversion.

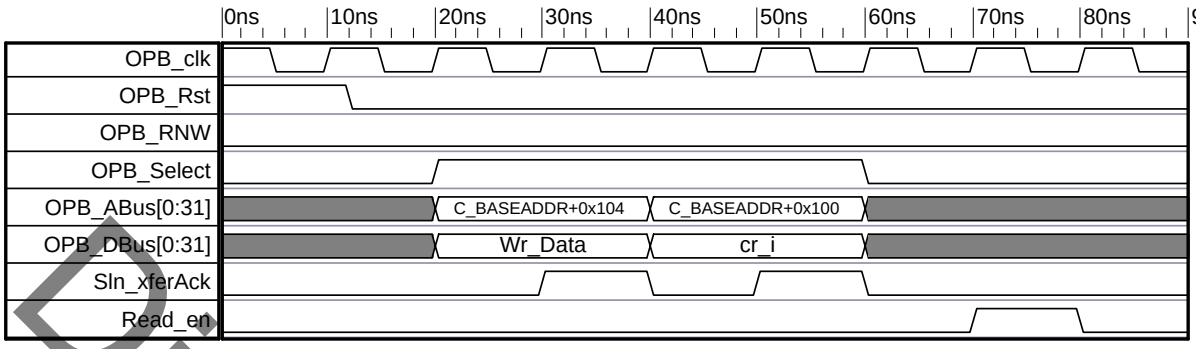


Figure 12: **Conversion of single data word**

To generate a DC value:

- Write the data to be converted into the Data FIFO.
- Enable the DAC by writing a '1' to the control register.
- Drive Read_en high for one OPB_Clk, the value written into the Data FIFO will start being converted two Dac_Clk_en later.
- If a new value is needed, write the new value into the Data FIFO.
- Drive Read_en high for one OPB_Clk, the value written into the Data FIFO will start being converted two Dac_Clk_en later.

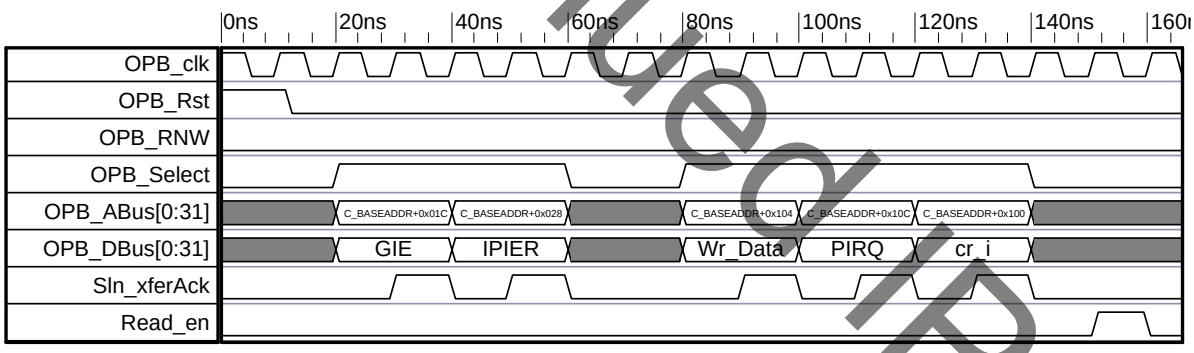


Figure 13: **Conversion of multiple Data Words**

To generate a waveform:

- Initialize the interrupt registers as required if interrupts are to be used.
- Write the data to be converted into the Data FIFO(Wr_Data is the digital word to be converted).
- Set the PIRQ to generate an interrupt before the FIFO is empty, if interrupts are to be used.
- Enable the DAC by writing a '1' to the control register.
- Drive Read_en high for one OPB_Clk, the first value written into the Data FIFO will start being converted two Dac_Clk_en later.

Discontinued IP

- After the appropriate number of DAC_Clk_en have occurred drive the Read_en high for one clock and the next value will start being converted two Dac_Clk_en later.
- If the read_en strobes high and the Data FIFO is empty the DAC output will be driven to '0'. DAC output will remain zero until data is written to the Data FIFO and a read_en occurs.

Design Implementation

Design Tools

Xilinx's ISE is the synthesis, map and place and route tool used for the OPB_DeltaSigma_DAC design.

Target Technology

The intended target technology is a Virtex-II Pro FPGA.

Device Utilization and Performance Benchmarks

Since the OPB_DeltaSigma_DAC is a module that will be used with other design pieces in the FPGA, the utilization and timing numbers reported in this section are just estimates. As the OPB_DeltaSigma_DAC is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the OPB_DeltaSigma_DAC design will vary from the results reported here.

In order to analyze the OPB_DeltaSigma_DAC timing within the FPGA, a design was created that instantiated the OPB_DeltaSigma_DAC with the following parameters set.

The OPB_DeltaSigma_DAC benchmarks are shown in Table 10 for a Virtex-II Pro XC2VP7-6 ff672 FPGA.

Table 10: **DAC FPGA Performance and Resource Utilization Benchmarks (Virtex-II Pro XC2VP7-6)**

Parameter Values				Device Resources				f_{MAX}
C_NUM_DAC_BITS	C_FULL_RANGE	C_DEV_BLK_ID	C_DEV_MIR_ENABLE	Slices	Slice Flip-Flops	4-input LUTs	GCLKs	MHz
2	0	0	0	91	81	114	1	166.945
4	0	0	0	93	85	116	1	153.257
8	0	0	0	104	102	127	1	164.231
8	1	0	0	107	106	130	1	156.937
16	0	0	0	128	142	152	1	155.376
16	1	1	1	134	148	156	1	180.278

Notes:

1. These benchmark designs contain only the OPB_DeltaSigma_DAC with registered inputs/outputs without any additional logic. Benchmark numbers approach the performance ceiling rather than representing performance under typical user conditions.

Discontinued IP

Reference Documents

Analog Devices Data Converter Reference Manual, Volume I, 1992

High Performance Stereo Bit-Stream DAC with Digital Filter, R. Finck, IEEE Transactions on Consumer Electronics, Vol. 35, No. 4,Nov. 1989.

Virtex Synthesizable Delta-Sigma DAC, J. Logue, XAPP154 September 23, 1999 (Version 1.1)

Revision History

Date	Version	Revision
11/8/04	1.0	Initial Xilinx release.
4/4/05	1.1	Updated for EDK 7.1.1 SP1 release; updated trademarks and supported device listing.
8/30/05	1.2	Converted to new DS template; updated figures to Xilinx graphic standards.
12/1/05	1.3	Added Spartan-3E to supported device listing.

Discontinued IP