DISCRETE WAVELET TRANSFORM PROCESSOR

SPECIFICATION VERSION 1.04

ICs, LLC PO Box 2236
2040 Warren Wagon Road
McCall, ID 83638
208 315 0029
ics@ics-rhbd.com

August 2012 (Revised Table 3.5 values)

Features

- Implements 9/7 Integer and 9/7 Real DWT transform outlined in CCSDS 122.0-B-1 [1]
- 20 Mpixel/s data rate (with clock at 100 Mhz)
- 16-bit maximum parallel input width (signed, unsigned or offset supported)
- 4-bit serial programming interface
- 24-bit 2's complement parallel output (programmable right shift and rounding supported for float mode)
- Supports a broad range of user defined 3 level 2 dimensional transforms
- Supports user defined subband weighting for 9/7 Integer wavelet
- Supports images as small as 32x32 pixels
- Supports widths up to 8192
- Supports infinite image heights (non-infinite heights must be divisible by 32)
- 3.3V Pad ring voltage (required to interface to BPE and memory chips)
- 2.5V Core voltage
- TID > 100 KRad
- SEL Immune to 120 LET (Mev-cm^2/mg)
- Low SEU-induced error rate (LET onset threshold > 55 LET)
- 256 pin ceramic QFP (only 244 pins used)
- Typical power usage for 16-bit image ~ 200mW/MSample/sec
- Functions as stand-alone wavelet processor for feature and signature extraction with user-programmable wavelet coefficients
- With BPE chip, executes CCSDS122B "Image Data Compression" standards
  - Tunable compression rate or quality control from lossy to fully lossless
  - Space frame-based image data compression with reconfigurable frame compression rate for deep-space and planetary missions
  - Space push-broom and whisk-broom sensor data compression with on-the-fly compression rate reconfiguration for near Earth and planetary missions
  - Applicable to science data with more than two dimensions with excellent performance
  - Enabler for intelligent downlink rate control with fully embedded bit stream

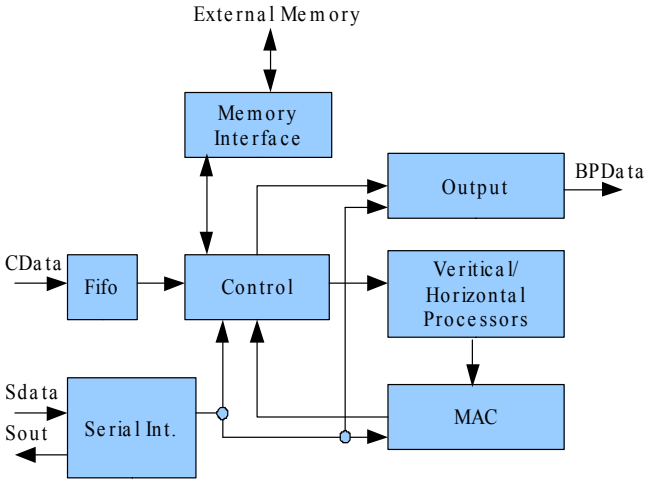


Table of Contents

1.0 GENERAL DESCRIPTION

The DWT implements a 3 level 2 dimensional discrete wavelet transform on an image that is input in a simple line-by-line format. The exact algorithm is outlined in the CCSDS 122.0-B-1 [1] documentation. A compression system based on this DWT, the BPE [2] and SRAM chips would be as follows:

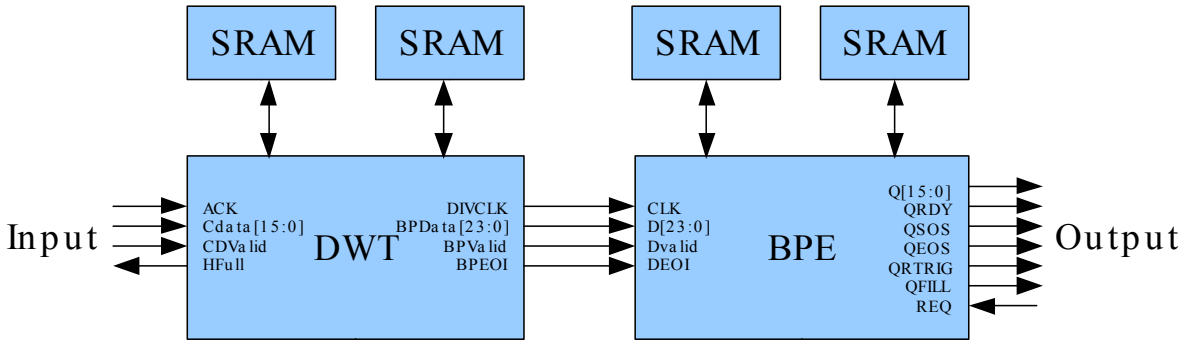


Figure 1.1: Example System Configuration

The DWT contains six major blocks: the external memory interface, the main control block, the output block, the serial interface, the vertical and horizontal processors and the MAC (multiply accumulate) block as shown in the figure below.

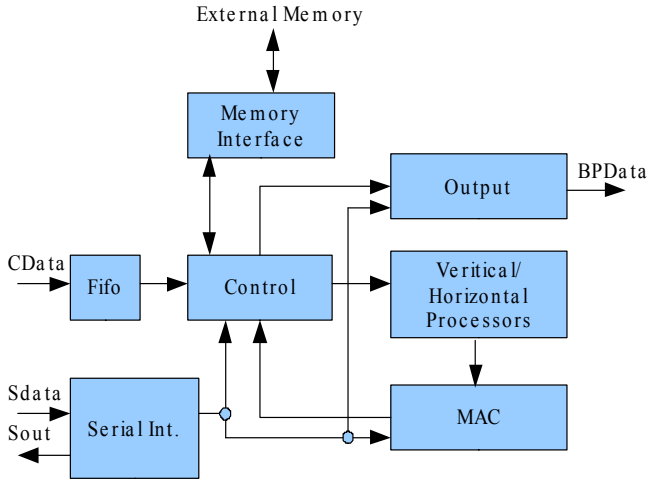


Figure 1.2: DWT Block Diagram

The basic operation of the chip consists of two modes: configuration and processing. The chip can be configured after reset and before any data is sent. The easiest way to configure the chip is to set Default to either 9/7 Integer (1) or 9/7 Real (0), reset the chip, use the serial interface to program width, height and out_address and then start inputing data. Once any data is sent the chip enters processing mode. The only way to return to configuration mode is via a reset. Once configured, data is essentially sent in a line-by-line fashion via the parallel data interface. To see in-depth algorithm details please refer to the CCSDS document.

2.0 OVERVIEW

There are many possible ways to implement the DWT algorithm specified in the CCSDS. Different approaches have different implications on area, power, RAM requirements, frequency, etc. For a space application meeting the required data rates, one issue rises to the top fairly quickly: finding RH (radiation hardened) external memories that meet the required data rates. Thus, this design tries to minimize the amount of external RAM needed and the amount of bandwidth required while still maintaining a reasonable clock frequency.

The DWT chip uses a system clock, driven by an external CLK input, at a maximum rate of 100 Mhz. The input imaging data on the CData input, is clocked in at a rate which must be less than or equal to 1/5 of the CLK period using the ACK input signal in conjunction with the CDValid signal. The DWT is programmed via the serial interface. Signals on this bus are latched on the rising edge of SCLK. SCLK is asynchronous to CLK, with no phase requirements. However, the high and low time of SCLK each must be greater than 1.2 CLK cycles. Once the DWT chip is programmed, data is then presented on the parallel interface. Signals on this bus are latched on the rising edge of ACK. ACK is asynchronous to CLK, with no phase requirements. However, the high and low time of ACK each must be greater than 1.2 CLK cycles. (Please refer to ACK timing diagram in section 5.0 for more detail.)

The DWT processes input imaging data in 3 steps summarized in Figure 2.1. In the first step the DWT chip performs the first level horizontal wavelet filtering and writes the results to external RAM, i.e. the SRAM in Figure 2.1. In the second step the DWT chip reads from the external RAM vertically, performs the first level vertical filtering and both the horizontal and vertical filtering for the next two wavelet transform levels, and then writes the results to external SRAM (but not in the order expected by the BPE). In the third and final step, the DWT chip reads the data back from external SRAM in the proper sequence desired by downstream processing.

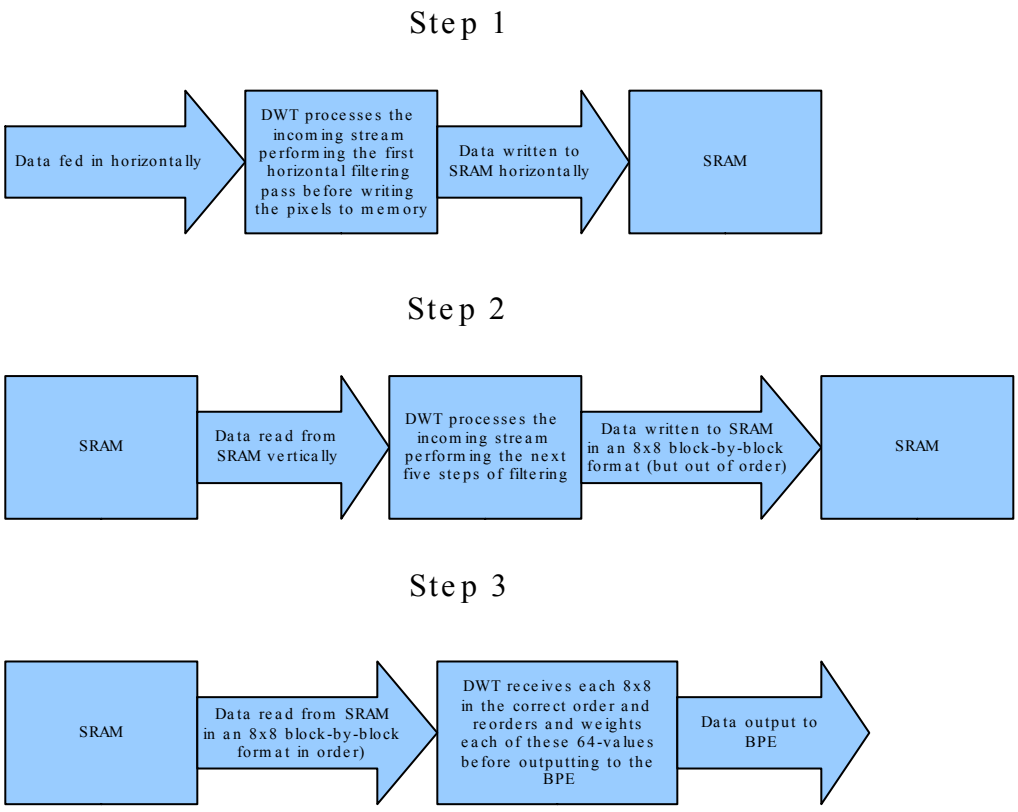


Figure 2.1: Data Flow Diagram

The DWT processes one strip (a strip is defined here as a 32-pixel high portion of the image) at a time. This is why image height must be divisible by 32 (or infinite). The external controller can pad lines in order to support images where the height is not divisible by 32. In order to reduce required memory bandwidth, it inputs enough of the image above and below that strip to produce the required values for each subsequent stage of processing so no intermediate storage or retrieval of data is needed. Because of this, the DWT must read several pixels above and below the strip being processed (unless it is the edge of the image). These values are then passed through the filter (MAC block) and passed on to the next processing step if appropriate. The final processed data must then be stored to memory and reread in the appropriate order for the BPE chip. For this reason, even if data is being input slowly into the DWT, it will tend to send 32*width processed pixels at a time to the BPE at the BPE's maximum data rate, and then pause until the next strip is ready to be sent.

The incoming image data is treated as a full 16-bit value. The Offset register (see table 3.3) is then added to this incoming value and the value is capped to a 16-bit result. This result is then sign extended or not (note that according to Config[6] in table 3.3 sign extension should only be done if the three bit value in Config[9:7] is greater or equal to 4) to a 24-bit 2's complement internal number. Finally, this 24-bit value is left shifted according to Config[9:7] (see table 3.3) before processing. This allows the user many options for offsetting, sign extending and resolution control. The reset values that the chip defaults to (shown in tables 3.4 and 3.5) assume that the incoming bits are right justified non-negative 16-bit values and applies no offset to them. See the appendix for pseudocode for this function. The output values are 2's complement 24-bit. The value of the LSB depends on Config[9:7]. If this value is 0, then the output represents an integer. For every value Config[9:7] is incremented, the value of the LSB is divided by 2. So, for example, if Config[9:7] is set to 3, the LSB of the output (assuming no right shifting on Config[12:10]) is 1/8. The purpose behind the incoming left shift is to allow maximum internal accuracy without overflowing in Real mode (in Integer mode these bits should always be set to 0). The right shift on the output allows the user to return to an integer representation if desired (or any representation with less resolution than was used internally).

3.0 CIRCUIT OPERATION

3.1 Top Level Diagram and Pin List

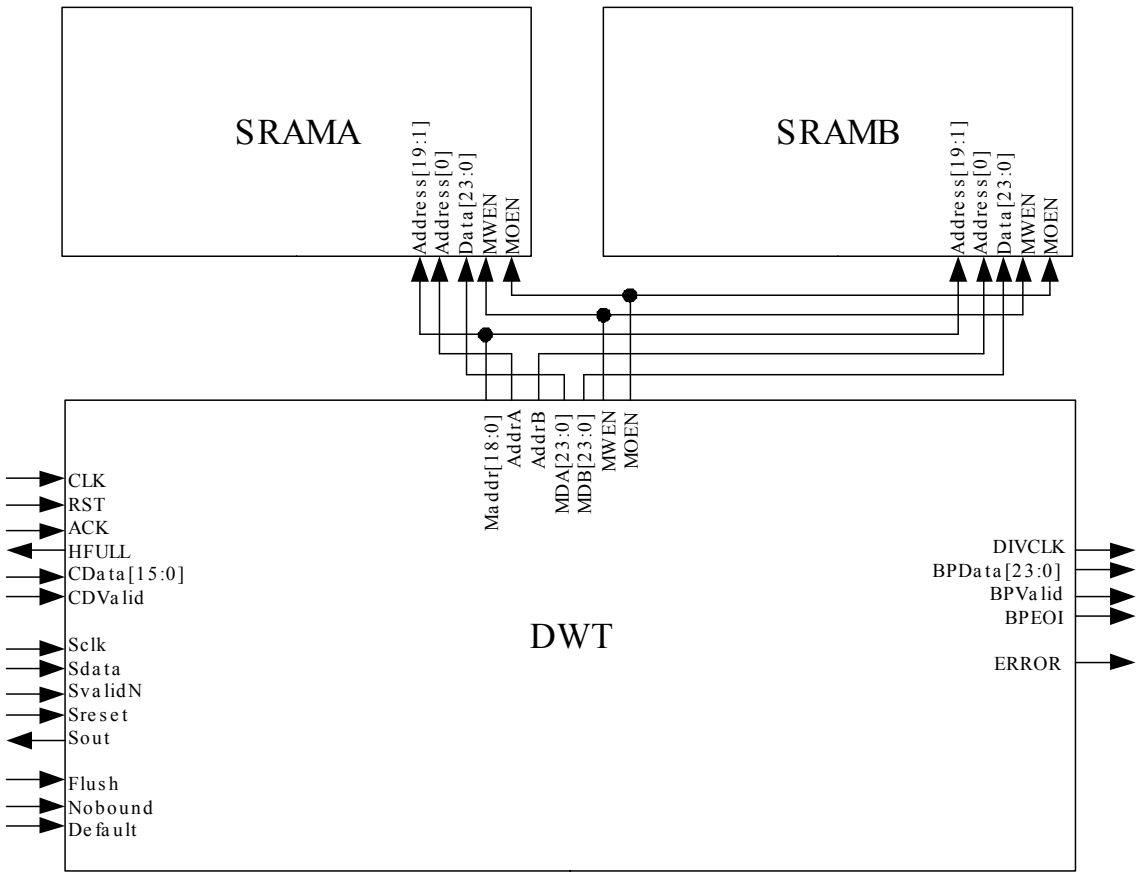


Figure 3.1: Example Connection Diagram

Table 3.1: Pin list		
Name	Type	Description
TESTEN	input	Must be tied to ground net on the board for normal operation (not shown in figure 3.1).
SCANEN	input	Must be tied to ground net on the board for normal operation (not shown in figure 3.1).
OSC_EN	input	Must be tied to ground net on the board (not shown in figure 3.1).
OSC_OUT	output	Should be left unconnected at the board level (not shown in figure 3.1).
TREE_EN	input	Must be tied to ground net on the board for normal operation (not shown in figure 3.1).
CLK	input	100 Mhz clk with a duty cycle between 45%/55% and 55%/45%.
RST	input	Reset. Active high. Must be held high for a minimum of 10 CLK cycles.
ERROR	output	Used to indicate an error condition that requires a reset to resolve. This pin will only assert in the case of a memory buffer overflow or an internal FIFO overflow. Only a chip reset will clear this condition. Note that this signal is active high.
ACK	input	Acknowledge signal used to latch CData[15:0] and CDValid (on it's rising edge). Note that ACK high time must be larger than 1.2 CLK cycles and ACK low time must be larger than 1.2 CLK cycles.
HFULL	output	Signals that the input FIFO is half full. This is an early warning that the maximum input rate of 1/5 of CLK's frequency is being exceeded. If the input FIFO overflows then the ERROR pin will be asserted and the chip must be reset to recover. Note that this pin is synchronous to CLK and not ACK.
CData[15:0]*	input	Chip data input. Used to input pixel data, synchronous to ACK and used in conjunction with CDValid. This value is latched on the rising edge of ACK. Unsigned data not requiring the full 16 bit range can simply tie the upper bits to 0. Signed data (Config[6]=1) must use sign extend to the full 16 bits.
CDValid	input	Used to denote CData is valid. Should not be asserted until 10 CLK cycles after the final serial programming data has been entered. This value is latched on the rising edge of ACK. If CDValid is low then CData is ignored. Note that this signal is active high.
Flush	input	Function unavailable in current version. Must be kept low. Do not set this pin as it may cause an internal FIFO to overflow. To flush pipeline, see section 3.2.3.
Default	input	Used to initially configure the chip when RST goes inactive. Needs to be valid before the inactive edge of RST and must be held valid for a minimum of 10 clock cycles after the inactive edge of RST. A value of 1 during this time period configures the chip for 9/7 integer mode and a value of 0 during this time period configures the chip for 9/7 real mode. After this time period the value of this pin is ignored.
Sreset	input	Used to reset the serial bus level counter. Note that this signal is active high.
Sclk	input	Serial interface input clock. Asynchronous to CLK but the frequency of Sclk must be <= 1/2.2 of the frequency of CLK.
Sdata	input	Serial interface data line.
SvalidN	input	Serial interface valid line. Note that this signal is active low.
Sout	output	Serial interface output line. Used to read back register values in conjunction with the SRead register. Sout is updated on the rising edge of SCLK and should, therefore, be sampled on or before the rising edge of SCLK.
Nobound	input	Used to ignore upper and lower image boundaries (no effect on the top boundary of the first image after a reset, in other words this image is still reflected according to the algorithm specification). This signal is used to achieve, effectively, infinite image height processing. Must not be used with Flush. Note that this signal is active high.
MAddr[18:0]	output	Memory address lines. These are the upper 19 address bits for each SRAM.
AddrA	output	Low bit memory address for memory chip A.
AddrB	output	Low bit memory address for memory chip B.
MDA[23:0]	inout	Data lines for memory chip A.
MDB[23:0]	inout	Data lines for memory chip B.
MWEN	output	Active low memory write enable.
MOEN	output	Active low memory output enable.
DIVCLK	output	Output clock which is essentially CLK/5. Intended to feed the BPE clock input but can also be used to drive ACK.
DIVCLK2	output	Identical to DIVCLK (but, due to internal delay differences, should never be shorted to DIVCLK).
BPData[23:0]*	output	Output data intended for the downstream encoder chip. Synchronous to DIVCLK.
BPValid	output	Active high, this signal signifies that BPData has valid data.
BPEOI	output	Active high, this signal signifies that the current data on BPData represents the final value for the current image being output (end of image).

* - Lower eight bits of these busses used for scan ins and scan outs when in test mode.

3.2 Initialization

The chip initializes itself to 9/7 Real or 9/7 Integer mode at the inactive edge of RST, based on the Default pin. These default values are listed in tables 3.5 and 3.6. Any programming through the serial interface after this time period (for 10 cycles after the falling edge of RST) will supersede these initial defaults (essentially the Default pin has no effect on the chip after the initial reset period). It should be noted that there are a minimum of three values that must be programmed before operation: Width, Height and Out_Add.

The DWT chip is programmed using a series of configuration shift registers (the chain counters) that must be accessed in a serial manner using the serial interface. There are six different configuration chains.

Table 3.2.0: General Chain Descriptions

Configuration Chain Number	Function
0	sets the values for input image width, height and Out_Add which is a function of SRAM size (see table 3.7);
1	sets the value for the Config register;
2	sets the input data offset value;
3	programs user-supplied DWT filter coefficients and rounding options for these coefficients;
4	programs user-defined DWT 8x8 block readout sequence, custom defined weights, output value bias and output mask;
5	programs read-back register values in DWT functional blocks.

For default DWT modes specified in [1], only Chain 0 has to be programmed. For other than the default DWT modes, other chains will need to be programmed.

3.2.1 Programming Configuration Chains

To start programming a chain, the serial configuration interface is first reset by latching in Sreset=1 at the rising edge of Sclk (or reseting the DWT). Note that if Sreset is high at the rising edge of Sclk, the value of SvalidN is ignored. Then 20 bits should be scanned into the serial interface, MSB first. The MSB is reserved and should always be set to 0, bits 18 to 16 specify which chain the lower sixteen bits will be shifted into. (Note that the purpose of Sreset is to reset the internal counters for output and input bits. If the user can guarantee that the number of SCLK rising edges from the last Sreset or chip reset is a multiple of 20, then no Sreset cycle is needed.)

Next, SvalidN is set low and one or more 20-bit packets should be clocked into the serial interface, MSB first, using the rising edge of Sclk. After the last big of the last chain has been clocked in Svalid should be set high. Alternatively, one could also simply stop Sclk. This sequence is illustrated in figure 3.2.0 below.

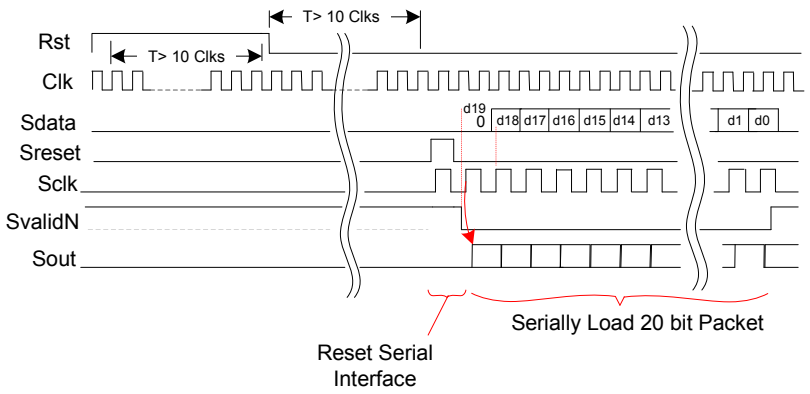


Figure 3.2.0: Serial Interface Example

The number of configuration bits in each of the chains varies. When the number of configuration bits in a chain is greater than 16 bits, then multiple packets must be used to program the entire chain (MSB to LSB). Once a chain has been started to be programmed the programming of that entire chain must be completed. Partial programming of a chain is prohibited. The MSB of each packet is reserved and should always be set to 0. Bits 18 to 16 specify which chain the lower sixteen bits will be shifted into. Table 3.2.1 lists the configuration shift registers (chains) and corresponding bit position.

For example, to program chain zero one would need to shift in 4 20-bit values. The upper 4 bits of each of these 20 bit values would all be 0 (to specify all these values are to be written to chain 0). So the order of bits clocked into Sdata would be as shown in figure 3.2.1.

Bits: 19-16				Bits: 15 - 0	
0	0	0	0	C_o[63:48]: Width	packet 0
0	0	0	0	C_o[47:32]: Height	packet 1
0	0	0	0	C_o[31:16]: PAD[11:0], Out_Add[19:16]	packet 2
0	0	0	0	C_o[15:0]: Out_Add[15:0]	packet 3

Figure 3.2.1: Chain 0 Programming Packets

So the first four bits entered would be "0", followed by C_0[63], followed by C_0[62], etc. Chain one consists of only one sixteen bit value so programming chain one would require writing only one packet of 20 bits, with the order of bits being (from left to right):

0001:C_1[15:0]

So the first four bits entered when programming chain one would be 0, 0, 0, 1, followed by C_1[15], followed by C_1[14], etc.

For default DWT modes specified in [1], only chain 0 has to be programmed. For custom modes other chains will need to be programmed (see appendix for examples).

Note that chains are programmed via a shift chain method so once one begins programming a chain one must finish with the appropriate number of bits or the values will not be shifted to the correct locations. Each chain is, however, independent from the other chains so one can, for example, program chain 4 without programming chains 1, 2 or 3. Note, again, that the chains are entered from MSB to LSB in the order specified in the table below.

Table 3.2.1: Detailed Chain Descriptions		
Chain	Chain Location	Description
0	C_0[63:48]	Width. Must be >= 32 and <= 8192 and it must be divisible by 8.
0	C_0[47:32]	Height. Must be >= 32 and <= 8192 and it must be divisible by 32. (If the Nobound pin is going to be asserted then this value must be >= 512. If the Nobound pin is set and this value is less than 512 the DWT will behave in an erratic manner.)
0	C_0[31:0]	(PAD[11:0],Out_Add[19:0]). This address specifies the starting address pointer for the processed data which has to be written to the external RAM so it can be re-ordered for the BPE. PADDED VALUES ARE IGNORED AND CAN BE SET TO ANY VALUE. See section 3.10.
1	C_1[15:0]	Config register.
2	C_2[15:0]	Offset[15:0]. This offset is in 2's complement form and is added to incoming 16-bit data.
3	C_3[367:336]	(PAD[8:0];Mult4_hp[22:0]) These are the multiplier constants. Each are 23 bits in length. and are in the 2's complement form 2.21.
3	C_3[335:304]	(PAD[8:0];Mult3_hp[22:0])
3	C_3[303:272]	(PAD[8:0];Mult2_hp[22:0])
3	C_3[271:240]	(PAD[8:0];Mult1_hp[22:0])
3	C_3[239:208]	(PAD[8:0];Mult0_hp[22:0])
3	C_3[207:176]	(PAD[8:0];Mult4_lp[22:0])
3	C_3[175:144]	(PAD[8:0];Mult3_lp[22:0])
3	C_3[143:112]	(PAD[8:0];Mult2_lp[22:0])
3	C_3[111:80]	(PAD[8:0];Mult1_lp[22:0])
3	C_3[79:48]	(PAD[8:0];Mult0_lp[22:0])
3	C_3[47:32]	Roundm2. This is a rounding value used at a certain point in the calculations.
3	C_3[31:16]	Roundm. This is a rounding value used at a certain point in the calculations.
3	C_3[15:0]	Mask. Used in conjunction with roundm.
4	C_4[559:552]	RMap[63][7:0]. The 64 RMap values configure the order of the output (lower 6 bits specify reordering) and whether the values are left shifted by 0, 1, 2 or 3 bits (the upper two bits - used to support subband weighting). RMap[0] specifies the first value to be output, RMap[1] the second, etc.
4	C_4[551:64]	RMap[62:2][7:0]
4	C_4[63:56]	RMap[1][7:0]
4	C_4[55:48]	RMap[0][7:0]
4	C_4[47:32]	Bias. Sixteen bit unsigned bias added to the output values if rounding is used.
4	C_4[31:0]	(PAD[7:0],OMask[23:0]). Output mask used in conjunction with the bias register.
5	C_5[15:0]	SRead[15:0]. Used to read back register values from various blocks.

3.2.2 Reading Configuration Parameters

The serial interface pin Sout is used to read back configuration parameters residing in the DWT chip. Chain 5 controls which bits will be clocked out of Sout on the rising edge of Sclk. The output from Sout also uses a 20-bit packet format with the leading four most significant bits always set to "0". The remaining 16 bits in the packet carry the read-back parameter bits. Thus the 20-bit packet available at Sout pin will always have the form: $0000C_{15}C_{14}C_{13}...C_2C_1C_0$ as shown in figure 3.2.2.

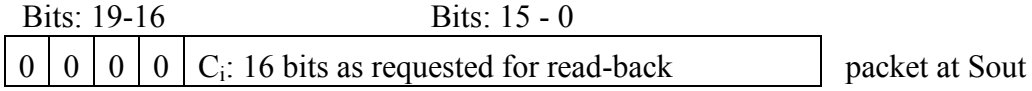


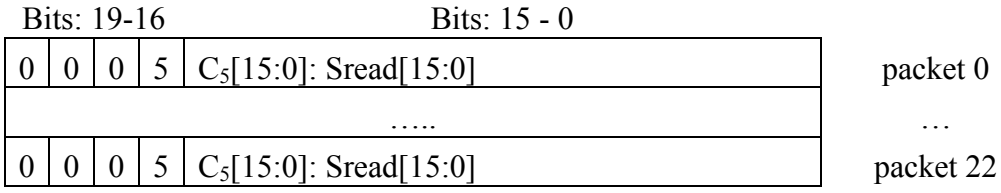
Figure 3.2.2: 20-bit Read-Back Packet Format at Sout

To read back configuration parameters, chain 5 can be used to access the values of these parameters which reside in different functional blocks of the chip (shown in figure 1.2). However, not all chains exist in every block. The following list shows which block to access for each chain:

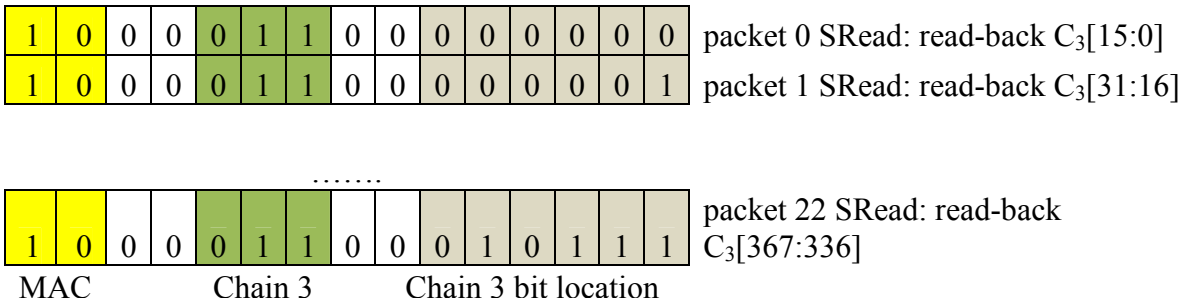
DWT Functional Block	Accessible Configuration Chain
Control Block	0, 1, 2, 5
MAC Block	1,3,5
Output Block	0,1,4,5

The 16-bit SRead register can be programmed to access chains residing in the Control Block, Mac Block or the Output block. The format for SRead is described in Table 3.3.

For example, to read back chain 3, the MAC block needs to be accessed. This is accomplished by writing to chain 5 a value which programs Sread to point to chain 3 in the MAC block. To read back all of the 368 bits in chain 3 would require programming Sread (in chain 5) 23 times to read back each 16-bit section as shown in figure 3.2.3.



(a). Chain 5 for Read-Back Entire Chain 3 values



(b). SRead in Chain 5 for Read-Back Chain

Figure 3.2.3: Read-Back Chain 3 Bit Values via Programming Chain 5

One way to accomplish this procedure would be as follows:

1) Reset serial interface by asserting Sreset on the rising edge of Sclk.
2) Program SRead to point at the lower 16 bits of chain 3 in the MAC block.
3) Wait from 10 clocks after writing SRead for Sout to become valid.
4) Cycle Sclk and sample Sout on the rising edge of Sclk. This will correspond to the MSB of a 20-bit packet, the lower 16 of which will correspond to the lower 16 bits of chain 3 in the MAC block.
5) Cycle Sclk and sample Sout on the rising edge. This will correspond to the bit just below the MSB of the 20-bit packet described above.
6) Keep cycling Sclk and sampling Sout until all 20 bits are retrieved.
7) Jump to step 2, programming Sread to point at the next packet to be read until all 23 values have been read.

Note, again, that 10 CLK cycles are required after SRead is set before Sout can be considered valid. Sout will change with each SCLK regardless of SvalidN, so it's possible to read a value out with or without writing a value in. Note that partial read back of a chain is permitted by simply programming SRead to the 16-bit section in question and reading out that 20-bit packet.

A more detailed example of using the serial interface to read a register is shown in figure 3.2.4.

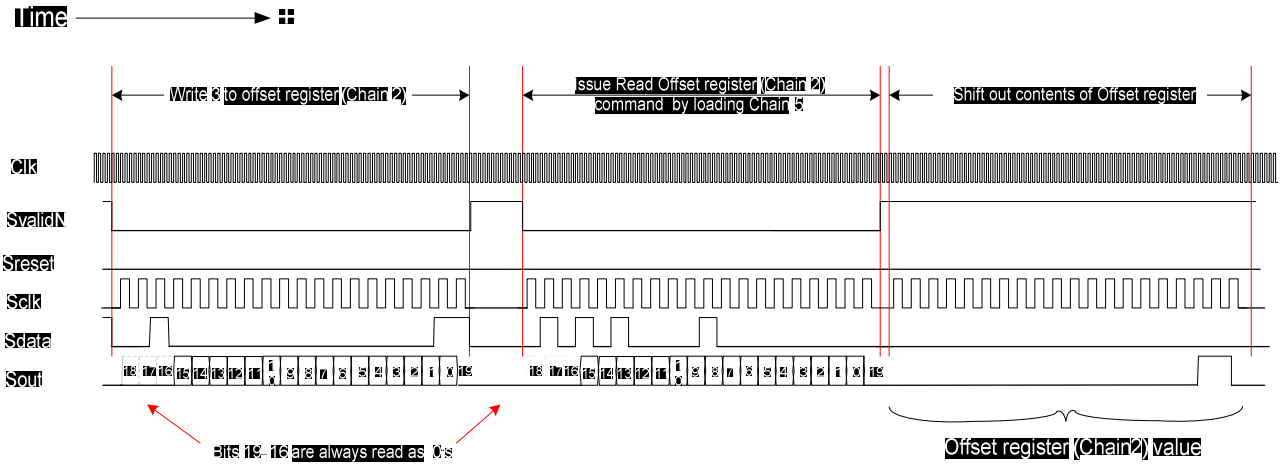


Figure 3.2.4: Detailed Serial Interface Example

Detailed descriptions of registers are given below.

Table 3.3: Register Descriptions

Register	Description
Width	Width register. Minimum acceptable value is 32. Maximum is 8192 but can also be limited by the amount of external RAM being used. Also, this value must be divisible by 8. Using inappropriate values will result in unpredictable behavior.
Height	Height register. Minimum acceptable value is 32. Maximum is 8192. Also, this value must be divisible by 32. Using inappropriate values will result in unpredictable behavior.
Out_Add	Twenty bit 2's complement address register. Let X denote the number of address locations in each SRAM. This register should be set to -(X/4). See table 3.7 for example values.
Config	General configuration register. Bits: 15-13: Test bits. These bits should not be set during normal operation. 12-10: Programmable output right shift. In integer mode this should be set to 0. In real mode it should be set to the same value as bits 9-7 in order to return the internal value to an integer. HOWEVER, it can be set to values lower than bits 9-7 if more output precision is desired (at the cost of compression ratio since these lower bits are essentially random).

	9-7: Incoming left shift. This determines how many bits are to the right of the decimal point internally. For "Integer" modes this should always be set to 0. For "Real" modes this should be set to the largest value that still allows enough internal range so that the internal numbers never overflow the internal 24-bit resolution. For the specified 9/7 Real mode filter tap values with 16-bit magnitude data, internal values can range up to 21-bit values, so a 3 is used here. For 15-bit incoming unsigned data, this could be set to 4, for 14-bit unsigned data it could be set to 5, etc. 6: Sign extend. Used to sign extend the incoming 16-bit values. NOTE: This only works if bits 9-7 contain a value of 4 or greater. ALSO, incoming signed data that is less than 16-bit MUST BE EXTENDED TO FULL 16-bit values by replicating the sign bit (as is standard for 2-s complement numbers). 5: Reserved. This bit should not be set during normal operation. 4: Flush mode. Function unavailable in current version. Must be kept at low. Do not set this bit (or the flush pin) as it may cause an internal FIFO to overflow. To flush pipeline, see section 3.2.3. 3: Lifting emulation mode. This bit must be set to emulate a lifting filter (required to be set for 9/7 integer mode). If this mode is set then the lower 10 bits of the lowpass multiplier constants should be zero. 2: Rounding mode. Should be set to 1 (round 1/2 down) for integer mode and 0 (round 1/2 up) for real mode. This configures an internal rounding constant used in the MAC block. 1: DIVCLK divider. Should be set to 0 for normal operation, in which case DIVCLK will have a frequency of 1/5 of CLK. If set to 1 then DIVCLK will have a frequency of 1/6 of CLK (under such a setting the data rate in would have to be limited to CLK/6 as well). 0: BIST. This bit should never be set during normal operation.
Offset	16 bit value added to CData[15:0] (result is capped to a 16-bit value) before the left shift specified in Config[9:7] or sign extension of Config[6] is applied.
mult4_hp	23 bit value corresponding to the +/- 4 coefficient for the highpass filter in the 2's complement form 2.21 (the msb represents the value -2, the next bit has the value 1, the next 1/2, etc. and the last bit has the value 1/(2^21)). Example: -2 would translate to 0x400000, -1 to 0x600000, 1 to 0x200000, 1/2 to 0x100000, etc.
mult3_hp	23 bit value corresponding to the +/- 3 coefficient for the highpass filter in the 2's complement form 2.21.
mult2_hp	23 bit value corresponding to the +/- 2 coefficient for the highpass filter in the 2's complement form 2.21.
mult1_hp	23 bit value corresponding to the +/- 1 coefficient for the highpass filter in the 2's complement form 2.21.
mult0_hp	23 bit value corresponding to the center coefficient for the highpass filter in the 2's complement form 2.21.
mult4_lp	23 bit value corresponding to the +/- 4 coefficient for the lowpass filter in the 2's complement form 2.21.
mult3_lp	23 bit value corresponding to the +/- 3 coefficient for the lowpass filter in the 2's complement form 2.21.
mult2_lp	23 bit value corresponding to the +/- 2 coefficient for the lowpass filter in the 2's complement form 2.21.
mult1_lp	23 bit value corresponding to the +/- 1 coefficient for the lowpass filter in the 2's complement form 2.21.
mult0_lp	23 bit value corresponding to the center coefficient for the lowpass filter in the 2's complement form 2.21.
Roundm2	A 16-bit rounding value used with the final add after the multiplies and only in lifting mode. This is not the rounding constant configured by Config[2]. This should be set to 0x01ff for integer mode and 0x0000 for real modes.
Roundm	A 16-bit rounding value used in conjunction with the mask register for lifting mode for the lowpass calculation. This is not the rounding constant configured by Config[2]. This should be set to 0x007f for integer mode and 0x0000 for real modes.
Mask	A 16-bit mask value used to truncate after rounding in lifting mode for the lowpass calculation. This should be set to 0xff00 in integer mode and 0xffff in real mode.
RMap[63:0]	Each of these 64 registers is 8 bits wide. The lower six bits specify the reordering while the upper two bits specify the left shift that will be applied (0,1,2 or 3) in order to support subband weighting.
Bias	A 16-bit positive bias which can be added to the output after subband weighting and before the right shift specified in Config[12:10]. It is intended to allow rounding before right shifting down (essentially throwing away precision). Thus, this number should generally be set to 2^(Config[12:10]-1).
OMask	A 24-bit mask applied to the output after the bias is applied (to allow truncation after the rounding bias is added).
SRead	A 16-bit register used to read back register values from various blocks and various chains. Bits 15-14: Specifies which block to read from. 00 - No block 01 - Control block 10 - MAC block 11 - Output block 13-11: Ignored. 10-8: Specifies which chain to read. Values from 0 to 5 are valid. 7-6: Ignored. 5-0: Specifies which 16-bit section to read. A zero would denote the lowest 16-bit section of the chain ([15:0]).

	It should be noted that not all chains exist in every block. The Control block contains chains 0, 1, 2 and 5. The MAC block contains chains 1, 3 and 5. And the Output block contains chains 0, 1, 4 and 5. Trying to read a chain from a block which does not contain it will lead to unpredictable results. To keep Sout from toggling unnecessarily, this register should be set to 0 before pixel data is entered into the DWT. Outputs are output in a MSB to LSB order on the Sout pin.

3.2.3 Flushing Data

The DWT chip is designed to keep the data pipeline section as full as possible. This is the only way to maintain the 20 Megasample/s processing rate. In this standard mode, however, the only way to get out the final processed values of an image is to start inputting the next image.

In order to flush all the data out one must enter three strips of data (a strip is defined as 32 rows at the currently programmed image width) of the next image (or dummy data).

3.3 9/7 Real Mode

If the Default pin is set to zero at the falling edge of reset, the registers are reset to the values shown in table 3.4.

Table 3.4: Register Reset Values for 9/7 Real Mode (Default=0)

Value in Hex	Register/Description
0000	Width. Resets to 0 and must be programmed before use.
0000	Height. Resets to 0 and must be programmed before use.
00000	Out_Add. This is a 20-bit register and should be set according to external memory size. It resets to 0.
0d80	Config float mode
0000	Offset for incoming pixel data.
000000	mult4_hp
7def4c	mult3_hp
014d54	mult2_hp
0d6103	mult1_hp
66c4ba	mult0_hp
0135e4	mult4_lp
7f3ca0	mult3_lp
7c75c4	mult2_lp
0c13af	mult1_lp
1b494f	mult0_lp
0000	Roundm2
0000	Roundm
ffff	Mask
2f	RMap[63]
23	RMap[62]
17	RMap[61]
0b	RMap[60]
27	RMap[59]
1b	RMap[58]
0f	RMap[57]
03	RMap[56]
2d	RMap[55]
21	RMap[54]
15	RMap[53]
09	RMap[52]

26	RMap[51]
1a	RMap[50]
0e	RMap[49]
02	RMap[48]
2b	RMap[47]
1f	RMap[46]
13	RMap[45]
07	RMap[44]
25	RMap[43]
19	RMap[42]
0d	RMap[41]
01	RMap[40]
29	RMap[39]
1d	RMap[38]
11	RMap[37]
05	RMap[36]
24	RMap[35]
18	RMap[34]
0c	RMap[33]
00	RMap[32]
2e	RMap[31]
22	RMap[30]
16	RMap[29]
0a	RMap[28]
3b	RMap[27]
35	RMap[26]
37	RMap[25]
31	RMap[24]
2c	RMap[23]
20	RMap[22]
14	RMap[21]
08	RMap[20]
39	RMap[19]
33	RMap[18]
36	RMap[17]
30	RMap[16]
2a	RMap[15]
1e	RMap[14]
12	RMap[13]
06	RMap[12]
3a	RMap[11]
34	RMap[10]
3f	RMap[9]
3d	RMap[8]
28	RMap[7]
1c	RMap[6]
10	RMap[5]
04	RMap[4]
38	RMap[3]
32	RMap[2]
3e	RMap[1]
3c	RMap[0]

0004	Bias
fffffff	Omask
0000	SRead

This mode is basically a straight forward symmetrical filter. Enough precision is provided to provide near lossless transform to inverse transform operation. (Currently over two billion random pixels have been passed through the DWT -> inverse DWT transform in the Real mode with no bit deviations. This does not guarantee the Real mode is lossless, but it demonstrates that, if the error rate is not zero, it is very low.)

3.4 9/7 Integer Mode

If the Default pin is set to one at the falling edge of reset, the registers are set to the values shown in the table below.

Table 3.5: Register Reset Values for 9/7 Integer Mode

Value in Hex	Register/Description
0000	Width. Resets to 0 and must be programmed before use.
0000	Height. Resets to 0 and must be programmed before use.
00000	Out_Add. This is a 20-bit register in 2's complement that should be set according to external memory size. It resets to 0.
000c	Config
0000	Offset
000000	mult4_hp
020000	mult3_hp
000000	mult2_hp
6e0000	mult1_hp
200000	mult0_hp
008000	mult4_lp
000000	mult3_lp
7b8000	mult2_lp
080000	mult1_lp
200000	mult0_lp
01ff	Roundm2
007f	Roundm
ff00	Mask
2f	RMap[63]
23	RMap[62]
17	RMap[61]
0b	RMap[60]
67	RMap[59]
5b	RMap[58]
4f	RMap[57]
43	RMap[56]
2d	RMap[55]
21	RMap[54]
15	RMap[53]
09	RMap[52]
66	RMap[51]
5a	RMap[50]
4e	RMap[49]
42	RMap[48]
2b	RMap[47]
1f	RMap[46]

13	RMap[45]
07	RMap[44]
65	RMap[43]
59	RMap[42]
4d	RMap[41]
41	RMap[40]
29	RMap[39]
1d	RMap[38]
11	RMap[37]
05	RMap[36]
64	RMap[35]
58	RMap[34]
4c	RMap[33]
40	RMap[32]
6e	RMap[31]
62	RMap[30]
56	RMap[29]
4a	RMap[28]
7b	RMap[27]
75	RMap[26]
b7	RMap[25]
b1	RMap[24]
6c	RMap[23]
60	RMap[22]
54	RMap[21]
48	RMap[20]
79	RMap[19]
73	RMap[18]
b6	RMap[17]
b0	RMap[16]
6a	RMap[15]
5e	RMap[14]
52	RMap[13]
46	RMap[12]
ba	RMap[11]
b4	RMap[10]
bf	RMap[9]
fd	RMap[8]
68	RMap[7]
5c	RMap[6]
50	RMap[5]
44	RMap[4]
b8	RMap[3]
b2	RMap[2]
fe	RMap[1]
fc	RMap[0]
0000	Bias
ffffff	Omask
0000	SRead

The 9/7 Integer mode transform-inverse transform operation is guaranteed lossless. Therefore, a lossless compression mode is possible with DWT set to this mode and the BPE configured appropriately.

3.5 Control Block

The control block serves many functions. It controls the flow of information between all the other blocks and keeps the data pipeline full, prioritizing which blocks get access to the the memory interface in order to keep the processors, MAC and output blocks as busy as possible.

It's also responsible for formatting the input data according to the Config and offset registers. As data enters the chip the first operation performed on it is to add the Offset register to it. It is then sign extended or not, depending on how the Config register is programmed, as it is expanded to 24 bits. This number is then left shifted by the value specified by config[9:7] and sent to the first level horizontal pixel processor.

3.6 Vertical and Horizontal Processors

The vertical and horizontal processors are responsible for reordering and reflecting the data properly before it is presented to the MAC block. Certain control bits are passed along with each pixel from the control block to facilitate this task.

3.7 MAC Block

The MAC block receives 9 24-bit values on a data bus with some control bits that specify whether the values should be run through the low-pass or high-pass filter. The values are then processed and sent back to control with additional bits to mark what processor they were received from. This is essential as the control block must keep track of the (x,y) location of incoming pixels in order to determine if they are to be written back to memory or sent back to the next level pixel processor.

When in real DWT mode, the MAC operates in the configuration depicted in Figure 3.4 for both high-pass and low-pass filters. Note that Data1 through Data9 represent the sequence of pixels being processed while Data5 is the center value. Basically the multiplier implements the filter in a very straightforward manner in this mode. Please note that the rounding constant depicted in figure 3.4 does not refer to any register. This rounding constant has only two possible values and is fully specified by setting (or clearing) bit 2 of the Config register.

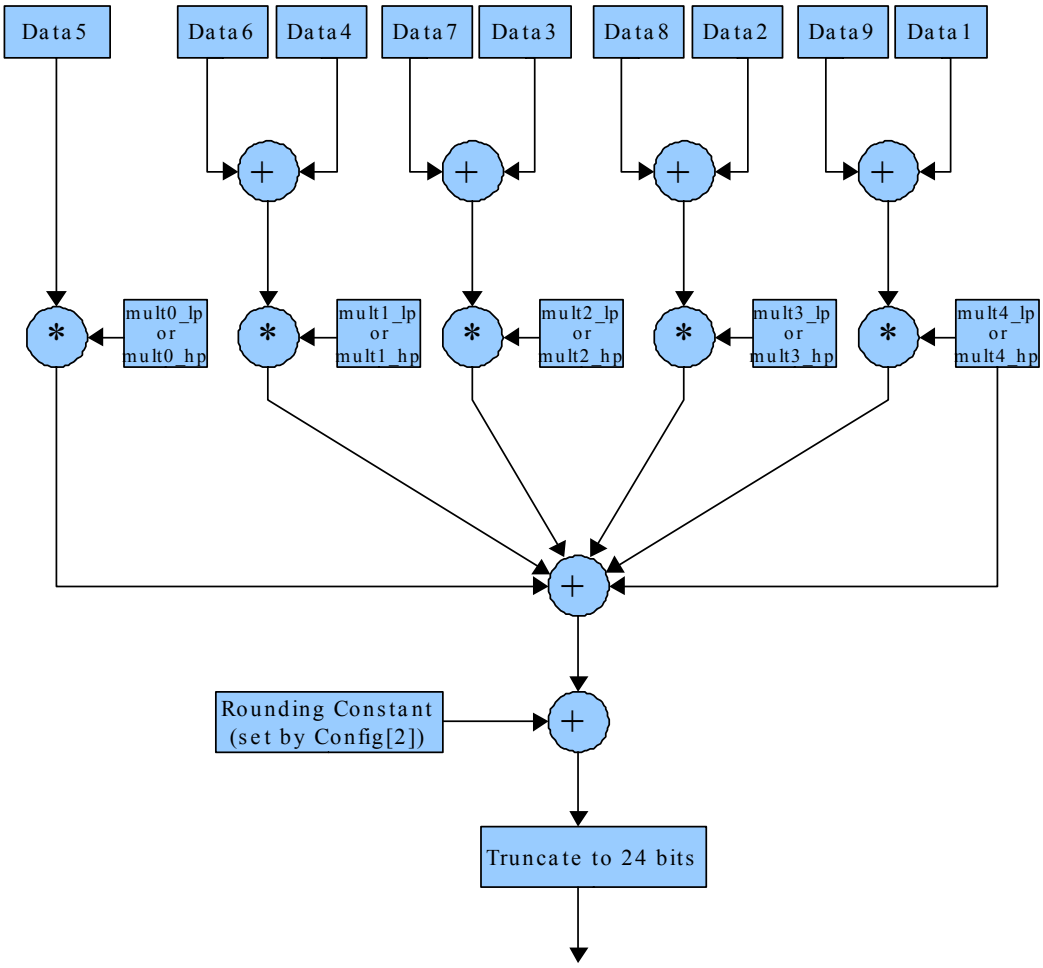


Figure 3.4: Real Filter Configuration

For the integer DWT, the low-pass filter is implemented using a lifting scheme [3]. The integer DWT produces integer wavelet coefficients, thus allowing lossless compression with bit-to-bit accuracy to be performed in the down stream BPE coder. For performing high pass filtering on the input pixels with integer DWT, the same configuration shown in Figure 3.4 is used. For low-pass filtering the input pixels, the processing is performed using the configuration shown in Figure 3.5

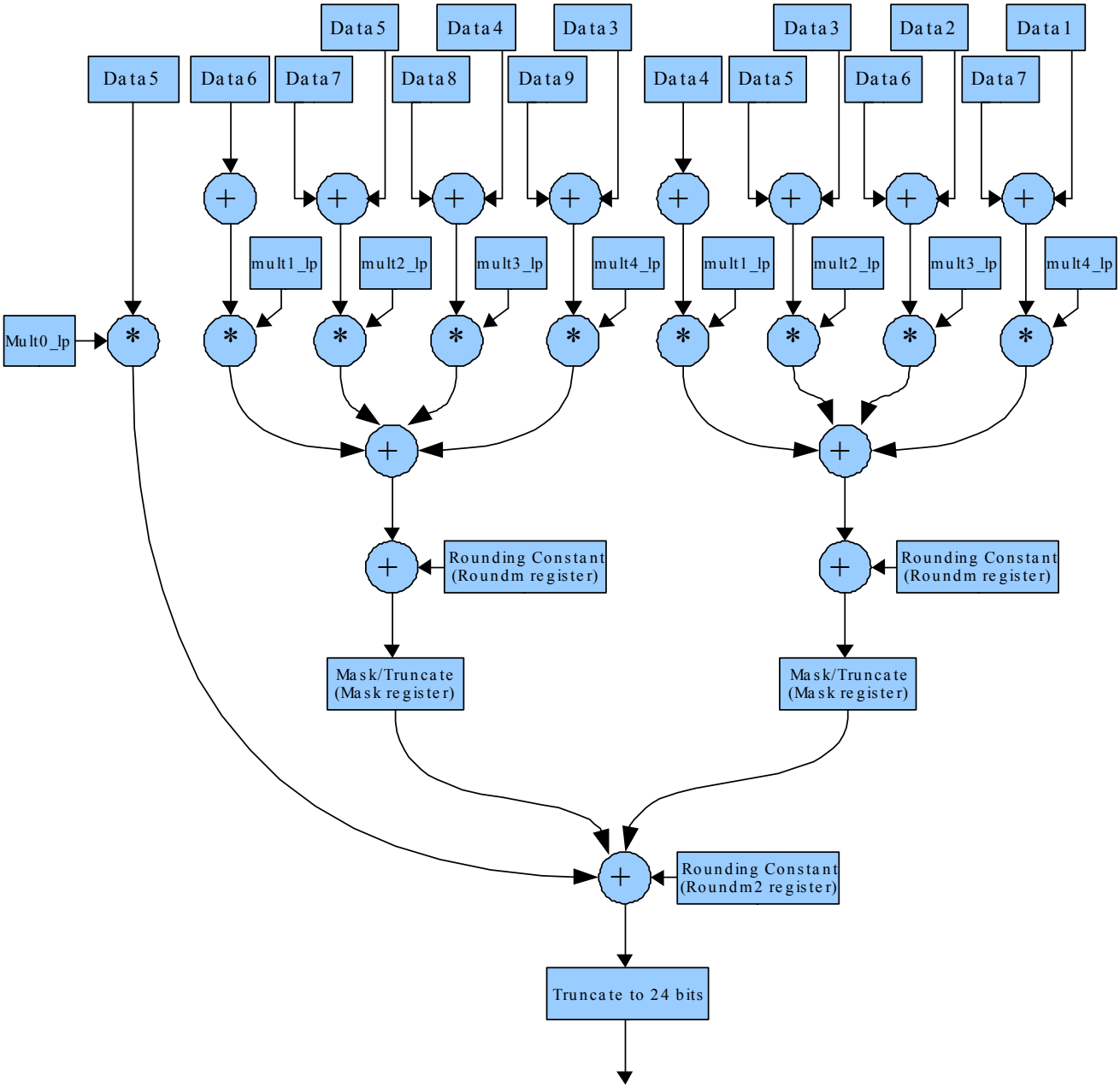


Figure 3.5: Lifting Mode Filter Configuration

To calculate the low-pass filter results, the lifting scheme for the integer DWT described in [3], the DWT chip first computes two high-pass coefficients before applying an updating procedure which produces one low-pass filter result. Figure 3.4 shows such a "duo-filter" scheme in which the updating multipliers have been propagated back to the predicting state and combined with the high-pass filters. Therefore, the values for mult4_lp, mult3_lp, mult2_lp and mult1_lp are scaled values from mult3_hp, mult2_hp, mult1_hp and mult0_hp. For the default integer DWT, the scaling factor is 1/4. With this "duo-filter" scheme, the multiplier hardware in the lifting mode can be reduced to 12 bits for the filter tap coefficients instead of the 23 bits used in Figure 3.3. The floor function for the default integer DWT mode [1] is implemented with appropriate rounding constants, i.e. Roundm and Roundm2, which are added to the filter results as well as a truncation scheme implemented by a binary masking specified by the Mask register.

Because of the architecture used for implementing the lifting scheme for low-pass results, only limited types of integer DWT are supported.

It should be noted that the MAC does not saturate on overflows. The DWT is designed and configured to have enough internal bits so that it does not overflow given the filter constants given in the CCSDS document (see [3]).

3.8 Output Block

The output block receives processed 8x8 blocks from the control block which it retrieves from memory as the start of step three in figure 2.1. The computed 8x8 DWT block coefficients, as a result of Step 2 processing in Figure 2.1, are stored in the external SRAM. These coefficients are then retrieved from the external SRAM in order to complete Step 3 in Figure 2.1. The coefficient retrieval in Step 3 follows a fixed order shown in Table 3.6, in parenthesis for all the 64 coefficient in the ten subbands (LL3, HL3, LH3, HH3, HL2, LH2, HH2, HL1, LH1, HH1). These subbands are arranged in the local DWT block described in [1]. To provide these coefficients to the downstream processing, coefficient reordering may be needed.

Consider an 8x8 pixel block, consistent with the CCSDS documentation, where the order in which these values arrive at the output block noted as shown in table 3.6.

Table 3.6: 8x8 CCSDS Block Filter Ordering

LL3 (60)	HL3 (62)	HL2 (50)	HL2 (56)	HL1 (4)	HL1 (16)	HL1 (28)	HL1 (40)
LH3 (61)	HH3 (63)	HL2 (52)	HL2 (58)	HL1 (6)	HL1 (18)	HL1 (30)	HL1 (42)
LH2 (48)	LH2 (54)	HH2 (51)	HH2 (57)	HL1 (8)	HL1 (20)	HL1 (32)	HL1 (44)
LH2 (49)	LH2 (55)	HH2 (53)	HH2 (59)	HL1 (10)	HL1 (22)	HL1 (34)	HL1 (46)
LH1 (0)	LH1 (12)	LH1 (24)	LH1 (36)	HH1 (5)	HH1 (17)	HH1 (29)	HH1 (41)
LH1 (1)	LH1 (13)	LH1 (25)	LH1 (37)	HH1 (7)	HH1 (19)	HH1 (31)	HH1 (43)
LH1 (2)	LH1 (14)	LH1 (26)	LH1 (38)	HH1 (9)	HH1 (21)	HH1 (33)	HH1 (45)
LH1 (3)	LH1 (15)	LH1 (27)	LH1 (39)	HH1 (11)	HH1 (23)	HH1 (35)	HH1 (47)

Reordering is as simple as entering the desired sequence based on the map above. For example, if the LL3 value is to be output first for each 8x8 block, then RMap[0] should be set to 60 (or 0x3C in hex). For a baseline implementation in which BPE ASIC [2] is used, the output order for these 8x8 coefficients can take the "scan-order" starting from the top eight coefficients, followed by the eight coefficients on the second row and so on until the last eight coefficients on the bottom row. This preferred ordering can be programmed by the 64 RMap registers. In this case, the six lower bits for the RMap values (from RMap[0] to RMap[63]) are:

{60, 62, 50, 56, 4, 16, 28, 40, 61, 63, 52, 58, 6, 18, 30, 42,
48, 54, 51, 57, 8, 20, 32, 44, 49, 55, 53, 59, 10, 22, 34, 46,
0, 12, 24, 36, 5, 17, 29, 41, 1, 13, 25, 37, 7, 19, 31, 43,
2, 14, 26, 38, 9, 21, 33, 45, 3, 15, 27, 39, 11, 23, 35, 47}

The upper two bits of each RMap register specifies a programmable left shift of 0, 1, 2 or 3 for that value and allows the implementation of various subband weighting schemes. For more details see appendix.

For the 9/7 Real DWT mode in [1], setting the Default pin to 0 at the falling edge of reset will set these RMap values accordingly as shown in table 3.4. If the Default pin is set to 1 at the falling edge of reset then the higher 2 bits of each RMap register is used to specify the subband weights and these values are shown in table 3.5.

Before issuing the data to the downstream encoder, three data manipulations can be performed: offset, truncation and right shift. The offset is achieved by programming the Bias register up to a 16-bit value, which is added to the 24-bit internal values. The truncation is performed by programming the 24-bit Omask register which is applied to the sum of the internal value and the Bias register using a bit-wise binary AND operation. Finally, this value is right shifted according to Config[12:10] before being output to the BPData bus.

For the two default modes these final three data manipulations are not used. Thus, Omask is set to 0xffffff, Config[12:10] is set to 0x0 and Bias is set to 0x0000 in both default modes. For other applications, see appendix.

3.9 Memory Interface Block

The memory interface block consists of several FIFO's and it sends/receives data, addresses and commands from the control block. It then communicates with the off chip SRAM in order to store or retrieve data depending on what the command bits indicate. Since there are several data flows going on in parallel, the exact order of accesses is complex and depends upon the timing of incoming data.

Maximum supported widths can be limited by the external SRAM capacity. Each SRAM needs to have 128*width locations (by 24+ bits). See the table below for examples. (The memory bus is designed to support the high speed radiation hardened SRAM chips by Aeroflex.) NOTE THAT OUT_ADDR MUST BE PROGRAMMED ACCORDING TO THE ADDRESS LOCATIONS IN EACH SRAM AND NOT ACCORDING TO THE MAXIMUM SUPPORTABLE IMAGE WIDTH DESIRED. As an example, if each RAM was 128kx32bits, a 4 megabit RAM, and the image width you intended to use was 256, then OUT_ADDR should be programmed with the value 0xF8000 and NOT 0xFE000.

Table 3.7: Maximum Supportable Image Width

SRAMA/B Address Bits	Address Locations for each SRAM	Maximum Supportable Image Width	Out_addr value	Out_addr value (in hex)
12 bits	4k	32	-1024	0xFFC00
13 bits	8k	64	-2048	0xFF800
14 bits	16k	128	-4096	0xFF000
15 bits	32k	256	-8192	0xFE000
16 bits	64k	512	-16384	0xFC000
17 bits	128k	1024	-32768	0xF8000
18 bits	256k	2048	-65536	0xF0000
19 bits	512k	4096	-131072	0xE0000
20 bits	1024k	8192	-262144	0xC0000

3.10 Nobound Signal

The Nobound signal basically causes the internal chip to ignore vertical edge reflections (except for the first one which is, of course, the top of the first image). This signal should be asserted before any pixel data is entered but it doesn't have to be set before programming is completed via the serial bus.

If Nobound is to be asserted then the image height must be set to be no smaller than 512. Note that the register value is used for internal processing in this case and bears no effect on ending an image at the specified Height value. Once this signal is set and pixel data is entered into the device, deasserting it will have unpredictable results.

Nobound should never be used in conjunction with the the Flush signal. The only way to empty out the data pipeline when Nobound is set is to add pad data for a minimum of 95 lines worth of the image.

3.11 Input Data Rate/Error Signal

The Error pin represents a data overflow in one of the critical FIFO's that the chip monitors, the external memory (which is treated as a circular buffer) or the input FIFO on the parallel data bus. If this occurs data has likely been lost, the internal counters will almost certainly be wrong and the data output will not recover. Only a reset can recover from this condition.

This can only be caused if data is being fed into the chip at a rate the chip can not sustain. If flushing is disabled (Flush pin and Config[4] set to 0), then this should not occur unless the input rate is exceeding the frequency of the clock divided by five (20 megasamples/s assuming the clock is at it's maximum 100 Mhz frequency) or small images are being processed. If flushing is enabled or small image sizes are being processed, this rate decreases. A conservative rule is to wait for 900 clock cycles between images, if flushing is enabled, to avoid overflowing the chip in this situation. (Since input pixels can only be sent at 1/5 of the CLK rate, this is equivalent to a 180 pixel gap.) Another approach is to wait until the processed data exits the chip.

Small images can cause this issue as well because there is a certain amount of overhead at the top and bottom edge of an image. For images with a height larger than 256 this is not an issue. For smaller height values, an overflow can occur.

Two ways to meter chip input rate is to monitor the HFULL signal, which signals that the input FIFO is half full, or tying DIVCLK to ACK and inputting data synchronously at that 1/5 CLK rate. It should be noted, however, that neither of these methods automatically avoid an overflow when flushing is enabled (by either setting the Flush pin or Config[4] to 1) or when small images are being processed.

3.12 Flush Pin

Function unavailable in current version. Must be kept at low. Do not set this pin as it may cause an overflow in an internal fifo.

3.13 Latency

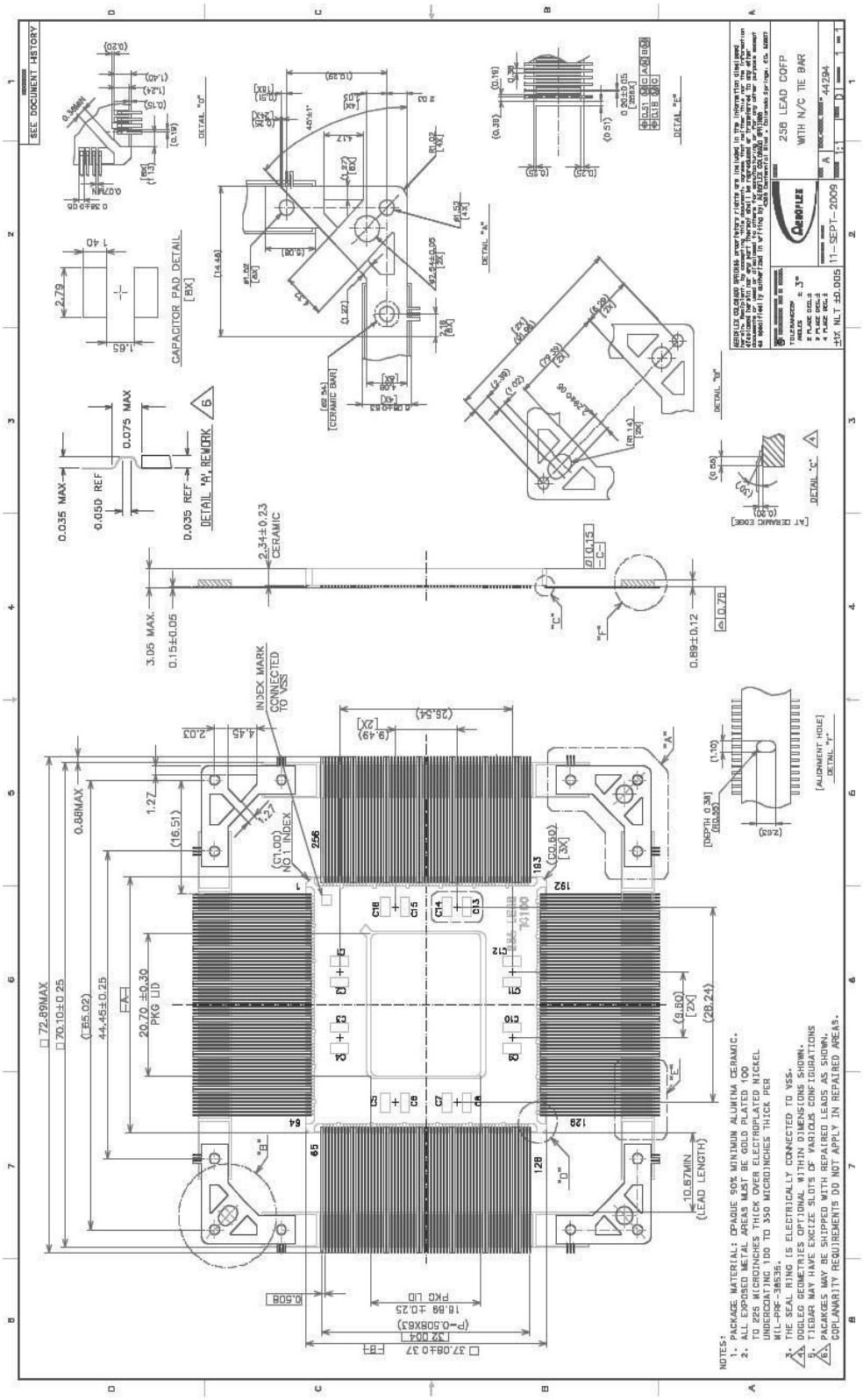
Latency for the DWT is large. Assuming flush is enabled for the last pixel input, the time from there to the time that the last processed pixel is output from the chip will be between 5*32*width and 3*5*32*width clock cycles later. So for the smallest image, 32x32, the DWT will see a latency roughly between 5 and 15 thousand clock cycles. For a moderately sized image with a width of 512, the latency jumps to between 82 and 246 thousand clock cycles. For the maximum width of 8192, the latency is between 1.3 to 3.9 million clock cycles.

This is due to several factors. Consider that the DWT, worst case, processes information approximately as fast as it is input. Also, it can't start processing a strip (remember a strip is defined in this document as a 32-pixel high x full width wide portion of the image) until the following strip is fulling entered. This means that when the final pixel is input, the DWT still has two 32xWIDTH strips left to process (at 5 clocks per pixel), and then the information must be output back to RAM and read out in the correct order for the BPE.

3.14 Power Cycling

It should be noted that if VDD and DVDD are not brought up simultaneously, then VDD should precede DVDD in order to avoid a "high current" state. DVDD should never be applied while VDD is low.

4.0 PINOUT AND PACKAGE DESCRIPTION



Pinout:

PIN	NAME	PIN	NAME	PIN	NAME	PIN	NAME
1	VDD	65	VDD	129	VDD	193	VDD
2	VSS	66	VSS	130	VSS	194	VSS
3	NC	67	NC	131	NC	195	NC
4	NC	68	NC	132	NC	196	NC
5	NC	69	NC	133	NC	197	NC
6	NC	70	NC	134	NC	198	NC
7	NC	71	NC	135	NC	199	NC
8	NC	72	NC	136	NC	200	NC
9	NC	73	NC	137	NC	201	NC
10	NC	74	NC	138	NC	202	NC
11	NC	75	NC	139	NC	203	NC
12	NC	76	NC	140	MDA[16]	204	NC
13	BPData[5]	77	NC	141	MDA[15]	205	NC
14	BPData[4]	78	NC	142	MDA[14]	206	NC
15	BPData[3]	79	NC	143	MDA[13]	207	NC
16	BPData[2]	80	NC	144	MDA[12]	208	CData[6]
17	BPData[1]	81	OSCOUT	145	MDA[11]	209	CData[5]
18	BPData[0]	82	MAddr[16]	146	MDA[10]	210	CData[4]
19	BPValid	83	MAddr[15]	147	MDA[9]	211	CData[3]
20	BPEOI	84	MAddr[14]	148	MDA[8]	212	CData[2]
21	MDB[23]	85	MAddr[13]	149	MDA[7]	213	CData[1]
22	DVDD	86	DVDD	150	DVDD	214	DVDD
23	DVSS	87	DVSS	151	DVSS	215	DVSS
24	MDB[22]	88	MAddr[12]	152	MDA[6]	216	CData[0]
25	MDB[21]	89	MAddr[11]	153	MDA[5]	217	ACK
26	MDB[20]	90	MAddr[10]	154	MDA[4]	218	CDValid
27	MDB[19]	91	MAddr[9]	155	MDA[3]	219	Flush
28	MDB[18]	92	MAddr[8]	156	MDA[2]	220	Default
29	MDB[17]	93	MAddr[7]	157	MDA[1]	221	TREE_EN
30	MDB[16]	94	MAddr[6]	158	MDA[0]	222	BPData[23]
31	MDB[15]	95	MAddr[5]	159	CLK	223	BPData[22]
32	MDB[14]	96	MAddr[4]	160	RST	224	BPData[21]
33	MDB[13]	97	MAddr[3]	161	Nobound	225	BPData[20]
34	MDB[12]	98	MAddr[2]	162	Sclk	226	BPData[19]
35	MDB[11]	99	MAddr[1]	163	Sreset	227	BPData[18]
36	MDB[10]	100	MAddr[0]	164	Sdata	228	BPData[17]
37	MDB[9]	101	MWEN	165	SvalidN	229	BPData[16]
38	MDB[8]	102	AddrB	166	Sout	230	BPData[15]
39	MDB[7]	103	AddrA	167	DIVCLK	231	BPData[14]
40	MDB[6]	104	MOEN	168	DIVCLK2	232	BPData[13]
41	MDB[5]	105	MDA[23]	169	ERROR	233	BPData[12]
42	VDD	106	VDD	170	VDD	234	VDD
43	VSS	107	VSS	171	VSS	235	VSS
44	MDB[4]	108	MDA[22]	172	HFULL	236	BPData[11]
45	MDB[3]	109	MDA[21]	173	CData[15]	237	BPData[10]
46	MDB[2]	110	MDA[20]	174	CData[14]	238	BPData[9]

47	MDB[1]	111	MDA[19]	175	CData[13]	239	BPData[8]
48	MDB[0]	112	MDA[18]	176	CData[12]	240	BPData[7]
49	MAddr[18]	113	MDA[17]	177	CData[11]	241	BPData[6]
50	MAddr[17]	114	NC	178	CData[10]	242	NC
51	TESTEN	115	NC	179	CData[9]	243	NC
52	SCANEN	116	NC	180	CData[8]	244	NC
53	OSC_EN	117	NC	181	CData[7]	245	NC
54	NC	118	NC	182	NC	246	NC
55	NC	119	NC	183	NC	247	NC
56	NC	120	NC	184	NC	248	NC
57	NC	121	NC	185	NC	249	NC
58	NC	122	NC	186	NC	250	NC
59	NC	123	NC	187	NC	251	NC
60	NC	124	NC	188	NC	252	NC
61	NC	125	NC	189	NC	253	NC
62	NC	126	NC	190	NC	254	NC
63	DVDD	127	DVDD	191	DVDD	255	DVDD
64	DVSS	128	DVSS	192	DVSS	256	DVSS

5.0 ELECTRICAL SPECIFICATION

Note that care must be taken on output pads to limit ringing to be no greater than 4.2V and no less than -0.8V. IBIS models available upon request.

5.1 Absolute Maximums

SYMBOL	PARAMETER	LIMITS
VDD	Power Supply	-0.3 to 2.95
DVDD	IO Power Supply	-0.3 to 3.8
$V_{I/O}$	Voltage on any given pin	-0.3 to 3.8[1]
T_{STOR}	Storage temperature	-65 to 150 C

[1]: Ringing from -0.8V to 4.2V can be tolerated for periods of less than 5ns.

5.2 Recommended Operating Conditions

SYMBOL	PARAMETER	LIMITS
VDD	Power Supply	2.375[2] to 2.75 Volts
DVDD	IO Power Supply	3.0 to 3.6 Volts
$V_{I/O}$	Voltage on any given pin	-0.3 to 2.75 Volts
$T_{JUNCTION}$	Junction operating temperature	-55 to 110 C[3]
T_{JA}	Thermal junction to ambient resistance	6.1 C/W
T_{JC}	Thermal junction to case resistance	4.8 C/W

[2]This can be lowered to 2.25 Volts by decreasing maximum operating frequency by 5%(this is multiplicative with temperature changes).

[3]Note that the maximum temperature of 110 C can be increased by decreasing maximum operating frequency. To increase maximum junction temperature to 125C, maximum frequency must be dropped to 67Mhz. To increase maximum junction temperature to 150C the maximum frequency must be dropped to 36 Mhz.

5.3 Operating Characteristics

SYMBOL	PARAMETER	Min.	Typ	Max.	Units	Conditions
VIH(min)	Min. High Level Input Voltage	0.7 * V_{DVDD}			volts	Guaranteed Logic '1'
VIL(max)	Max. Low Level Input Voltage			0.3 * V_{DVDD}	volts	Guaranteed Logic '0'
VIHC(min)	Min. High Level Input Voltage, High CLOCK Input	0.7 * V_{DVDD}			volts	Guaranteed Logic '1'
VILC(max)	Max. Low Level Input Voltage, Low CLOCK Input			0.3 * V_{DVDD}	volts	Guaranteed Logic '0'
VOH(min)	Min. High Level Output Voltage High	0.8 * V_{DVDD}			volts	IOH = -30.0 ma
VOL(max)	Max. Low Level Output Voltage Low			0.2 * V_{DVDD}	volts	IOL = 30.0 ma
IIH	High Level Input Current	-1		1	µA	Vin = VDD (pure input)
IIL	Low Level Input Current	-1		1	µA	Vin =VSS (pure input)
$IMAX_{DVDD}$	Maximum I/O operating current		0.1	0.7	A	@100 Mhz and max voltage and max utilization, max output loading
$IMAX_{VDD}$	Maximum core operating current		1.25	2	A	@100 Mhz and max voltage and max utilization

SYMBOL	PARAMETER	Min.	Typ	Max.	Units	Conditions
QI_{VDD}	Maximum Quiescent VDD current at 25C and 50Krad total dose[2]			150 uA		
QI_{DVDD}	Maximum Quiescent DVDD current at 25C and 50Krad total dose[2]			300 uA		

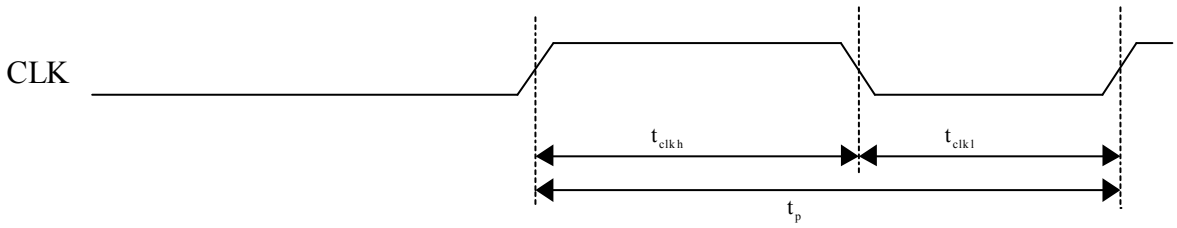
[2]Assuming dose rates below 40 millirads/second.

5.4 Power-Up Sequence

VDD must be brought up before or in conjunction with DVDD. DVDD must not be high while VDD is low as it results in a high current state in the level shifters within the pads.

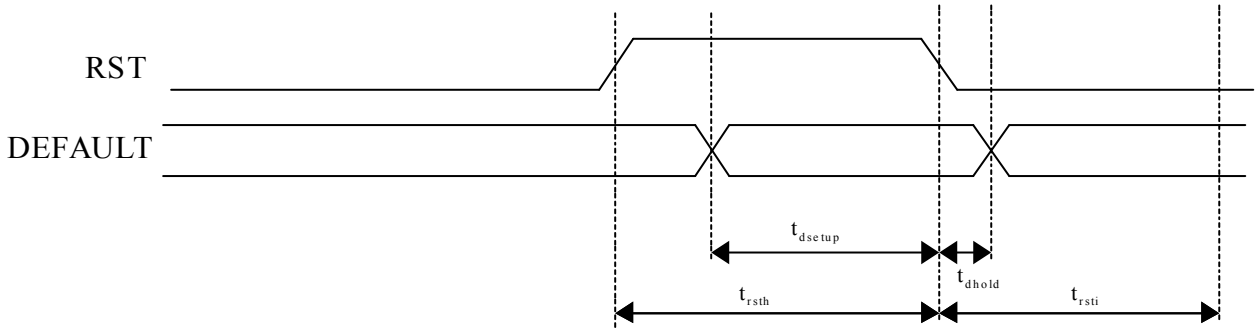
5.5 Timing Diagrams

CLK Timing



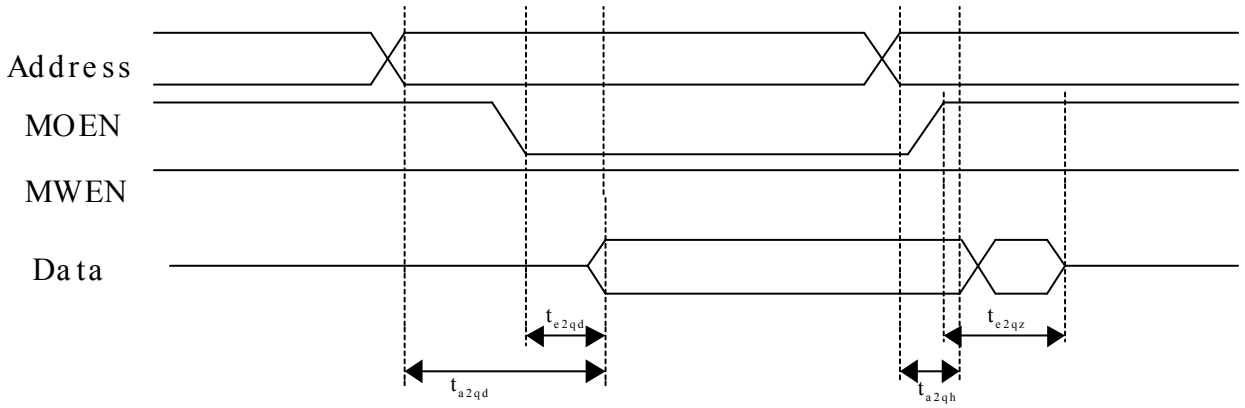
SYMBOL	PARAMETER	MIN	MAX	UNIT
t_{clke}	CLK rise/fall time.		2	ns
t_p	Clock period. Must be greater than or equal to 10 ns.	10		ns
t_{clkh}	Clock high time.	55%	45%	cycle
t_{clkl}	Clock low time.	45%	55%	cycle

RST Timing

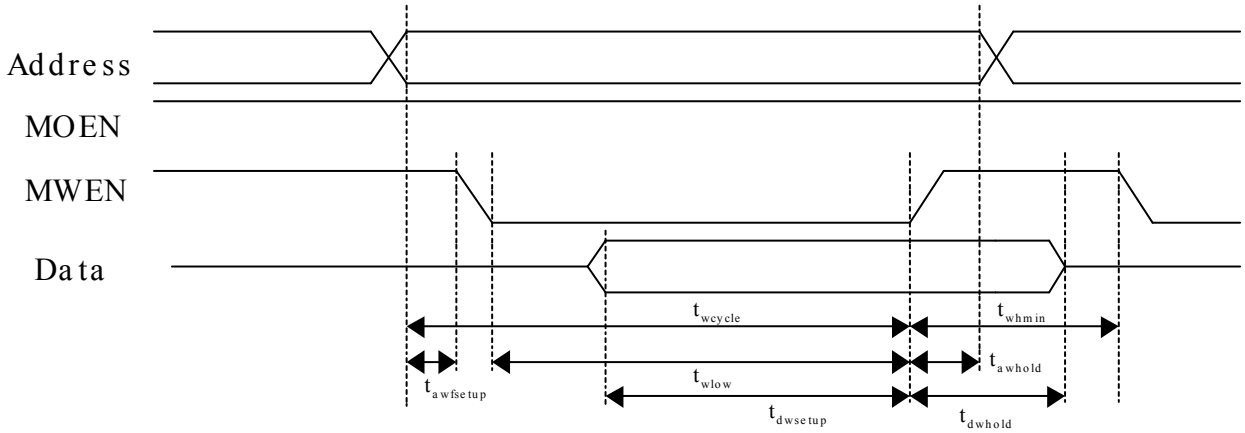


SYMBOL	PARAMETER	MIN	MAX	UNIT
t_{rste}	RST rise/fall time.		2	ns
t_{rsth}	RST high time.	10		cycles
t_{rsti}	RST inactive period required before any values are input into the device via the parallel or serial data ports.	10		cycles
t_{dsetup}	Default setup to falling edge of RST.	10		cycles
t_{dhold}	Default hold past falling edge of RST.	1.2		cycles

Memory Read

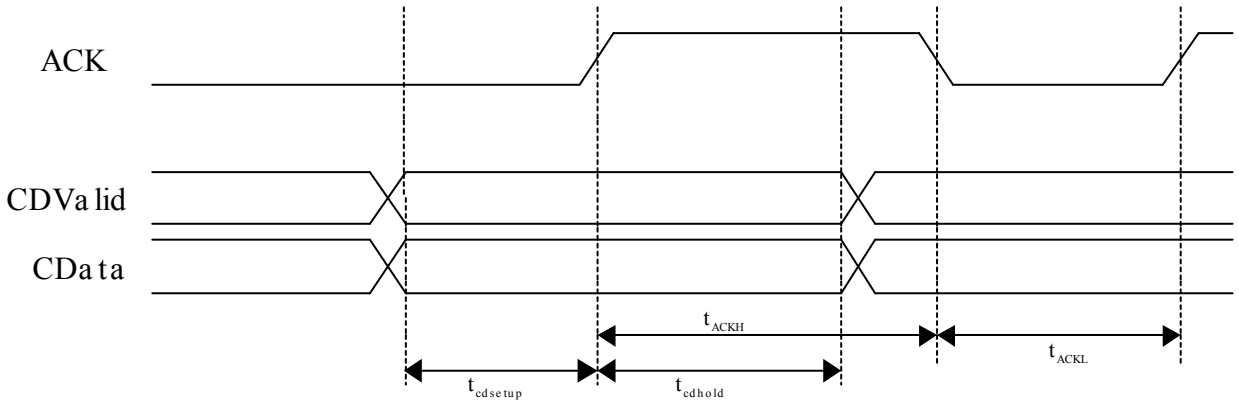


Memory Write



SYMBOL	PARAMETER	MIN	MAX	UNIT
t_{e2qd}	Memory output enable (MOEN) valid low to data valid. (Requirement for external memory.)		12	ns
t_{e2qz}	Memory output enable (MOEN) valid high to data tristate. (Requirement for external memory.)		7	ns
t_{a2qd}	Memory address valid to data valid. (Requirement for external memory.)		$1.5t_p$-2	ns
t_{a2qh}	Data valid hold time after memory address change. (Requirement for external memory.)	3		ns
t_{wlow}	Memory write enable low.	$1.5t_p$-2	$1.5t_p$+2	ns
t_{awhold}	Address hold time after rising edge of MWEN (write operation).	2		ns
$t_{awfsetup}$	Address setup time before falling edge of MWEN.	1		ns
$t_{dwsetup}$	Data setup time before rising edge of MWEN.	t_p		ns
t_{dwhold}	Data hold time after rising edge of MWEN.	$0.5t_p$-2		ns
t_{wcycle}	Write cycle time.	$2t_p$-3	$2t_p$	ns
t_{whmin}	Minimum high time for MWEN between each write.	$0.5t_p$-2		ns

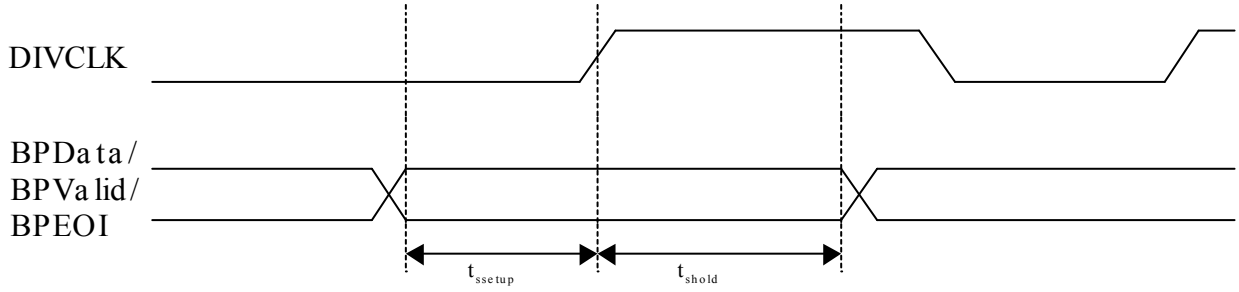
Data Bus Access



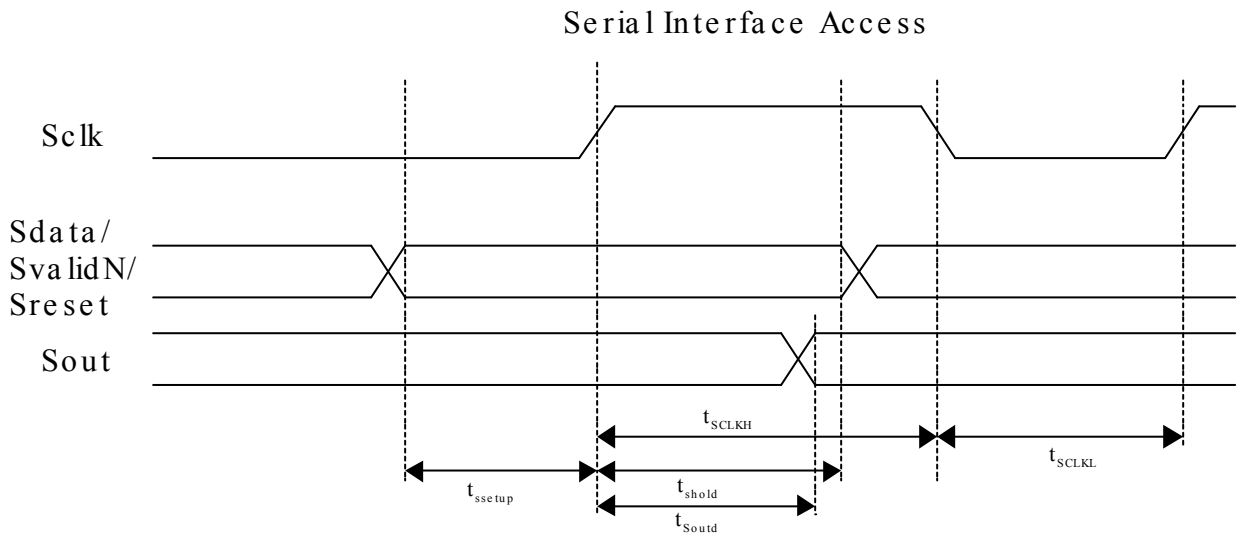
SYMBOL	PARAMETER	MIN	MAX	UNIT
t_{ACKE}	Rise/Fall time of ACK.		2	ns
$t_{cdsetup}$	CDValid and CData setup to rising ACK.	2		ns
t_{cdhold}	CDValid and CData hold time after rising ACK.	1		ns
t_{ACKH}	High time for ACK.	1.2		CLK cycles
t_{ACKL}	Low time for ACK.	1.2		CLK cycles

NOTE: Parallel data bus accesses where CDValid is high on the rising edge of ACK should not occur until 10 clock cycles after programming via the serial bus is completed.

BPE Interface Timing



SYMBOL	PARAMETER	MIN	MAX	UNIT
t_{bsetup}	BPData, BPValid and BPEOI setup to rising clock edge of DIVCLK.	3		ns
t_{bhold}	BPData, BPValid and BPEOI hold past rising edge of clock edge of DIVCLK.	3		ns

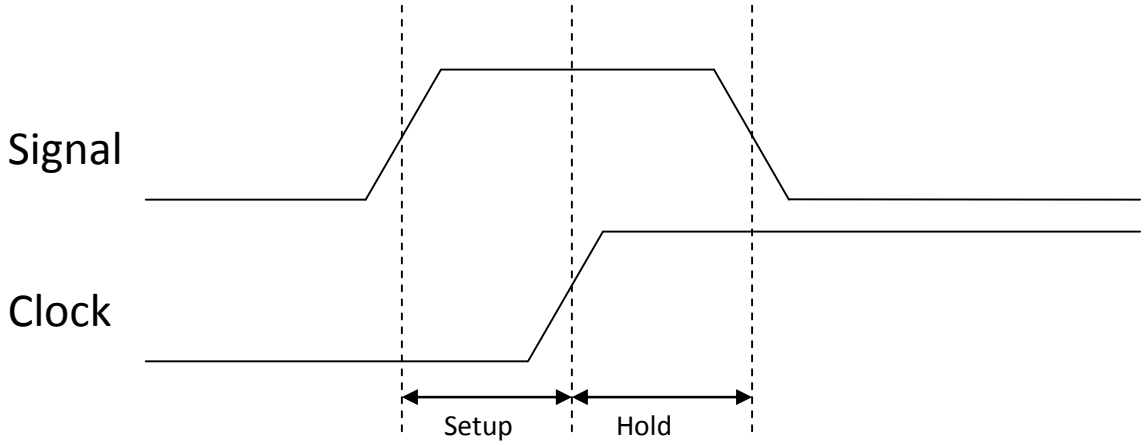


SYMBOL	PARAMETER	MIN	MAX	UNIT
t_{SCLKE}	Sclk rise/fall time.		2	ns
t_{ssetup}	Sdata, SvalidN and Sreset setup to rising Sclk.	2		ns
t_{shold}	Sdata, SvalidN and Sreset hold time after rising Sclk.	1		ns
t_{SCLKH}	High time for Sclk.	1.2		CLK cycles
t_{SCLKL}	Low time for Sclk.	1.2		CLK cycles
t_{Soutd}*	Sout delay from rising Sclk.		8	ns

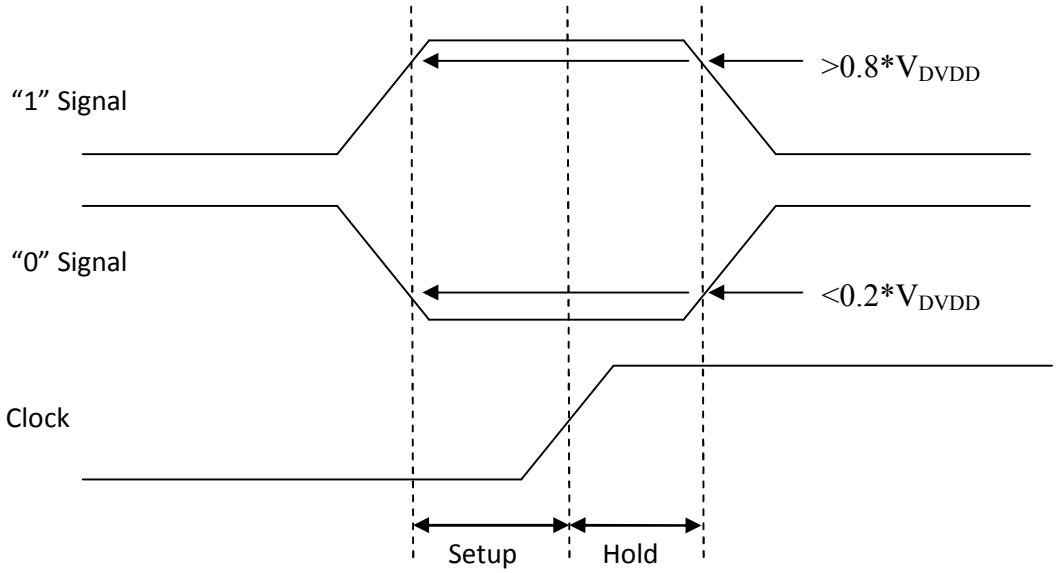
* NOTE: Sout is not guaranteed to be valid for 10 clock cycles after writing the Sread register, but once it is valid each successive bit will be valid no less than 8 ns after the rising edge of Sclk.

5.6 Reduced Edge Rate Timing

When edge rise/fall times are less than or equal to 2ns, setup and hold times are defined as crossover to crossover (that is, the point where the signals and clocks cross $V_{DVDD}/2$). When the edge rates are slower than this, setup is defined as the time from when the input signal is no more than $0.2*V_{DVDD}$ away from it's target value (so it must be <= $0.2* V_{DVDD}$ if it's final value is to be 0 and it must be >= $0.8* V_{DVDD}$ if it's final value is to be 1) to the crossover point of the edge sensitive signal it's referenced against. Furthermore, if the rise/fall time of the edge sensitive signal is greater than 2ns then the setup/hold time requirement must be increased by 25% of the difference of the rise/fall time and the ideal maximum of 2ns (that is by (rf_time – 2)*0.25 ns where rf_time is the maximum rise/fall time of the reference signal in nanoseconds).



Measuring Rise/Fall Times with Fast Edges



Measuring Rise/Fall Times with Slow Edges

Note that even when using the methods outline here for derating setup/hold values, clock edge rates (the clock signals include ack, clk and sclk) must be less than or equal to 5ns and the edge rates for all non-clock signals must be less than or equal to 20ns. Furthermore, it should be noted that slower edge rates on input signals increases crossbar currents on input pads, therefore increasing current draw on I/O power rails slightly.

6.0 References

[1] *Image Data Compression*. Recommendation for Space Data Systems Standards, CCSDS 122.0-B-1. Blue Book, Issue 1. Washington D.C.: CCSDS, November 2005. Available at www.ccsds.org.

[2] Preliminary specification for "Bit Plane Encoder", Version 0.7, U. of Idaho, April, 2007.

[3] *Image Data Compression*. Information Report for Space Data Systems Standards, CCSDS 120.1-G-1. Green Book, Issue 1. Washington D.C.: CCSDS, June 2007, available at www.ccsds.org.

Appendix:

A.1. Chain 2 Config Register

Config[9:7] and Config[12:10] can be used to increase internal precision for real DWT computation. For the default tap values in reference [3], the maximum dynamic range expansion for a 16-bit input data (assuming magnitude only) going through the default integer DWT will result in a 20-bit value (magnitude only). The chip's internal pipeline is in 2's complement form. This expands the number of bits needed to 21-bits. Thus, from [3], if real DWT mode is used, only a 21-bit pipeline is needed for input of 16-bit data. Therefore to fully utilize the 24-bit pipeline, for the default real DWT mode, the input 16-bit input data can be left-shifted by setting Config[9:7] = 3. This will increase internal computation precision and minimize rounding errors. Then at the output, setting Config[12:10]=3 will shift back the calculation values to correct range. For 15-bit input data, Config[9:7] can be set equal to 4, For 14-bit data it can be set equal to 5, etc.

Some examples are provided below:

Assume incoming data is 12 bit unsigned data, ranging from 0 to 4095. Assuming real mode, a good value for config that would maximize internal precision and return the output data to integer form would be (all bits not shown set to 0):

Config[12:10]=7
Config[9:7]=7
Config[6]=0

Assume incoming data is 15 bit signed data, ranging from -16384 to 16383. Assuming real mode, a good value for config would be:

Config[12:10]=4
Config[9:7]=4
Config[6]=1

Assume incoming data is 13 bit signed, ranging from -4096 to 4095. Assuming real mode, a good value for config would be:

Config[12:10]=6
Config[9:7]=6
Config[6]=1

NOTE: Signed numbers must be sign extended to a full 16-bit number AND config[6] must be set AND Config[9:7] must contain a value of 4 or greater (which means one can only sign extend 15-bit data or smaller). Offset can be used to remove the need for sign extension of the data if it can be shifted to be unsigned before entering the dwt as demonstrated in the example below.

Example: Assume incoming data is 13 bit unsigned, ranging from 0 to 8191. Offset is programmed with -4096 (0xf000 when expressed as a 2's complement 16-bit hexadecimal number). The new internal range would be -4096 to 4095. Since the range includes a negative number, the sign extend bit needs to be set. Programming for this mode would include:

Config[12:10]=6
Config[9:7]=6
Config[6]=1
Offset=0xf000

Thus, Config[6] must be set, and Config[9:7] still must be 4 or greater, but there is no need to sign extend the data.

A.2 Sign extension and offset pseudocode

Following is a section of pseudocode that describes how input data is manipulated according to the offset value, sign extend and left shift. Note that CONFIG refers to the Config register and OFFSET is the OFFSET register and DATA is the data fed to the internal data pipeline.

```
TMP[15:0] := CDATA[15:0] + OFFSET[15:0]
IF(CONFIG[6]==1 && TMP[15]==1) TMP[23:16]:= 0x0F;    /* note that this is 0x0F and NOT 0xFF which is why sign */
ELSE TMP[23:16]:= 0x00;                              /* extension only works when left shifting 4 or more bits  */
DATA[23:0] := TMP[23:0] << CONFIG[9:7];              /* note 0's are shifted in here */
```

A.3 Accessing the CRC Register for In-System Testing Purposes

There is an undocumented CRC register in the output block used primarily for manufacturing tests. It can be accessed by using the following SRead settings:

SRead[15:14]=3, SRead[10:8]=2 points to the lower 16 bits of this 26-bit CRC.
SRead[15:14]=3,SRead[10:8]=3 points to the upper 10 bits of this 26-bit CRC (it will be read out as the lower 10 bits of this location).

This CRC is calculated on outgoing BPE data according to the following verilog code:

```
################################################
reg [23:0] bpe_data;
reg[25:0] crc;

if(reset==1)
  crc <= 1;
else if(bpe_valid==1)
  begin
if(crc[25]==1)
  crc <= {crc[24:0],1'b0}^26'H0000047^bpe_data;
else
  crc <= {crc[24:0],1'b0}^bpe_data;
end
################################################
```

Thus, the crc is only updated when bpe_data presents a valid output and it is only reset on a chip reset. Therefore, if a fixed image is presented to the DWT (assuming the same configuration is used each time) after a reset, the CRC will be the same regardless of differences in input rates. That is, whether the data is input slowly or at full rate, the final CRC will be the same if the configuration and input data were the same and the test is done after resetting the CRC and before any other data is passed through the DWT.

This can be used to test the DWT and attached SRAMs by inputting known data (possible a long sequence generated by another LFSR) with a specificied configuration and checking the CRC against the known good CRC. This can be especially thorough if the number of inputs are large and a variety of configurations are used. To thoroughly test the RAMs it is recommended that a minimum of input samples should be four times the number of RAM locations (which would be eight times the number of address locations for one of the RAMs).

A.4. Chain 3 Wavelet Coefficient

To program the DWT chip to perform a 3-level wavelet transform other than the default integer and real wavelets in [1] should follow the same format used in representing these default coefficients. Note that the chip will support wavelet coefficients with up to 9 taps, denoted as mult*i*_hp, mult*i*_lp, $i = 0, \ldots, 4$, for both high and low frequency bands respectively.

The coefficients should be entered in a 2's complement 23-bit format with precision expressed in 2.21 format. That is, the first significant bit represents a value of "-2"; the second most significant bit represent "1". The next bit would be for value "1/2" and so on as shown in Figure A.4.0.

-2	1	2^{-1}	2^{-2}	2^{-3}	…….	2^{-21}

Figure A.4.0: Value Represented by 23-bit in 2.21 Format

For the default real DWT filter coefficients, using the 2.21 format, the corresponding bit representation is given in Table A.4.1. The last column lists the hex value of the filter coefficients (taps) which are the default values in Table 3.4.

Table A.4.1: Default Real DWT Filters in 23-bit 2's Complement Format

Filter Tap	Tap Value from [1]	-2	1	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	24-bit Hex
mult0_hp	-0.788485616406	1	1	0	0	1	1	0	1	1	0	0	0	1	0	0	1	0	1	1	1	0	1	0	66c4ba
mult1_hp	0.418092273222	0	0	0	1	1	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0d6103
mult2_hp	0.040689417609	0	0	0	0	0	0	1	0	1	0	0	1	1	0	1	0	1	0	1	0	1	0	0	014d54
mult3_hp	-0.064538882629	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	0	1	0	0	1	1	0	0	7def4c
mult0_lp	0.852698679009	0	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	1	1b494f
mult1_lp	0.377402855613	0	0	0	1	1	0	0	0	0	0	1	0	0	1	1	1	0	1	0	1	1	1	1	0c13af
mult2_lp	-0.110624404418	1	1	1	1	1	0	0	0	1	1	1	0	1	0	1	1	1	0	0	0	1	0	0	7c75c4
mult3_lp	-0.023849465020	1	1	1	1	1	1	1	0	0	1	1	1	1	0	0	1	0	1	0	0	0	0	0	7f3ca0
mult4_lp	0.037828455507	0	0	0	0	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	1	0	0	0135e4