Fast Fourier Transform v3.1

DS260 November 11, 2004 **Product Specification**

Introduction

The Fast Fourier Transform (FFT) is a computationally efficient algorithm for computing the Discrete Fourier Transform (DFT). The FFT core uses the Cooley-Tukey algorithm for computing the FFT[1].

Features

- Drop-in module for Virtex-II™, Virtex-II Pro™, Spartan-3™, and Virtex-4™ FPGAs
- Forward and inverse complex FFT
- Transform sizes $N = 2^m$, $m = 3 - 16$
- Data sample precision $b_x = 8,12,16,20,24$
- Phase factor precision $b_w = 8,12,16,20,24$
- Arithmetic types:
    - Unscaled (full-precision) fixed point
    - Scaled fixed point
    - Block floating point
- Rounding or truncation after the butterfly
- On-chip memory
- Block RAM or Distributed RAM for data or phase factor storage
- Run-time configurable forward or inverse operation
- Optional run-time configurable transform point size
- Run-time configurable scaling schedule for scaled fixed point
- Three architectures offer an exchange between core size and transform time
- For use with Xilinx CORE Generator™ system v6.3i and later

Overview

The FFT core computes an N-point forward DFT or inverse DFT (IDFT) where N can be 2^m, $m = 3–16$. The input data is a vector of N complex values represented as b_x-bit two's-complement numbers – b_x bits for each of the real and imaginary components of the data sample ($b_x = 8,12,16,20,24$). Similarly, the phase factors b_w can be 8, 12, 16, 20, or 24 bits wide.

All memory is on-chip using either Block RAM or Distributed RAM. The N element output vector is represented using b_y bits for each of the real and imaginary components of the output data. Input data is presented in natural order, and the output data can be in either natural or bit / digit reversed order.

Three arithmetic options are available for computing the FFT:

- Full-precision unscaled arithmetic
- Scaled fixed-point, where the user provides the scaling schedule
- Block-floating point

Note: For the scaled fixed-point and block-floating point options, superfluous LSBs can be either rounded or truncated after scaling.

Several parameters can be run-time configurable: the point size N, the choice of forward or inverse transform, and the scaling schedule. Both forward / inverse and scaling schedule can be changed frame by frame. Changing the point size resets the core.

Three architecture options are available:

- Pipelined, Streaming I/O. Allows continuous data processing.
- Radix-4, Burst I/O. Offers a load/unload phase and a processing phase; it is smaller in size but has a longer transform time.
- Radix-2, Minimum Resources. Uses a minimum of logic resources and is also a two-phase solution.

For detailed information about each option, see "Architecture Options" on page 4.

Theory of Operation

The FFT is a computationally efficient algorithm for computing a Discrete Fourier Transform (DFT). The DFT $X(k),\ k = 0,\dots,N-1$ of a sequence $x(n),\ n = 0,\dots,N-1$ is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-jnk2\pi/N} \quad k = 0,\dots,N-1$$

Equation 1: **DFT**

where N is the transform size and $j = \sqrt{-1}$. The inverse DFT (IDFT) is

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{jnk2\pi/N} \quad n = 0,\dots,N-1$$

Equation 2: **IDFT**

Algorithm

The FFT core uses the radix-4 and the radix-2 decomposition for computing the DFT. For two-phase solutions, the decimation-in-time (DIT) method is used, while the decimation-in-frequency (DIF) method is used for the streaming solution. When using radix-4, the FFT consists of $\log_4(N)$ stages, with each stage containing $N/4$ radix-4 butterflies. Point sizes that are not a power of 4 need an extra radix-2 stage for combining data.

An FFT using radix-2 has $\log_2(N)$ stages, with each stage containing $N/2$ radix-2 butterflies.

The inverse FFT (IFFT) is computed by conjugating the phase factors.

Finite Word Length Considerations

The radix-4 and radix-2 FFT algorithms process an array of data by successive passes over the input data array. On each pass, the algorithm performs radix-4 or radix-2 butterflies, where each butterfly picks up four or two complex numbers and returns four or two complex numbers to the same memory. The numbers returned to memory by the processor are potentially larger than the numbers picked up from memory. A strategy must be employed to accommodate this dynamic range expansion. Note that a full explanation of scaling strategies and their implications is beyond the scope of this document; for more information about this topic, see items 3 and 4 in the "References" section on page 36.

For a radix-4 DIT FFT, the values computed in a butterfly stage (except the second) can experience a growth to $4\sqrt{2} \approx 5.657$.

For radix-2, the growth can be up to $1+\sqrt{2} \approx 2.414$. This bit growth can be handled in three ways:

- Performing the calculations with no scaling and carry all significant bits to the end of the computation
- Scaling at each stage using a fixed-scaling schedule
- Scaling automatically using block-floating point

All significant bits are retained when doing full-precision unscaled arithmetic. The width of the data path increases to accommodate the bit growth through the butterfly.

When using scaling, a scaling schedule is used to scale by a factor of 1, 2, 4, or 8 in each stage. If scaling is insufficient, a butterfly output may grow beyond the dynamic range and cause an overflow. As a result of the scaling applied in the FFT implementation, the transform computed is a scaled transform. The scale factor s is defined as

$$s = 2^{\sum_{i=0}^{\log N - 1} b_i}$$

Equation 3: **Scale Factor**

where b_i is the scaling (specified in bits) applied in stage i.

The scaling results in the final output sequence being modified by the factor *1/s*. For the forward FFT, the output sequence
$X'(k), k = 0,\ldots,N - 1$ computed by the core is defined in Equation 4.

$$X'(k) = \frac{1}{s} X(k) = \frac{1}{s} \sum_{n=0}^{N-1} x(n) e^{-jnk2\pi/N} \qquad k = 0,\ldots,N-1$$

Equation 4: **Scaled FFT**

For the inverse FFT, the output sequence is

$$x(n) = \frac{1}{s} \sum_{k=0}^{N-1} X(k) e^{jnk2\pi/N} \qquad n = 0,\ldots,N-1$$

Equation 5: **Scaled IFFT**

If a radix-4 algorithm uses a scaling schedule of all 2's, the factor of *1/s* will be exactly equal to the factor of *1/N* in the inverse FFT equation (Equation 2). For radix-2, a scaling schedule of all 1's will provide the factor of *1/N*. Otherwise, additional scaling will be needed.

With block floating point, each data point in a frame is scaled by the same amount, and the scaling is kept track of by a block exponent. Scaling is performed only when necessary, which is detected by the core.

Architecture Options

The FFT core provides three architecture options to offer a trade-off between core size and transform time.

- **Pipelined, Streaming I/O**. Allows continuous data processing.
- **Radix-4, Burst I/O**. Offers a load / unload phase and a processing phase; it is smaller in size but has a longer transform time.
- **Radix-2, Minimum Resources**. Uses a minimum of logic resources and is also a two-phase solution.

Pipelined, Streaming I/O

This solution pipelines several radix-2 butterfly processing engines to offer continuous data processing. Each processing engine has its own memory banks to store the input and intermediate data (Figure 1). The core has the ability to simultaneously perform transform calculations on the current frame of data, load input data for the next frame of data, and unload the results of the previous frame of data. The user can stream in input data and, after the calculation latency, can continuously unload the results. If preferred, this design can also calculate one frame by itself or frames with gaps in between.

This architecture supports unscaled full-precision and scaled fixed point arithmetic methods. In the scaled fixed point mode, the data is scaled after every pair of radix-2 stages.

The unloaded output data can either be in bit reversed order or in natural order. By choosing the output data in natural order, additional memory resource will be utilized.

This architecture covers point sizes from 8 to 65536. The user has flexibility to select how many pipelined stages to use block RAM for data and phase factor storage.

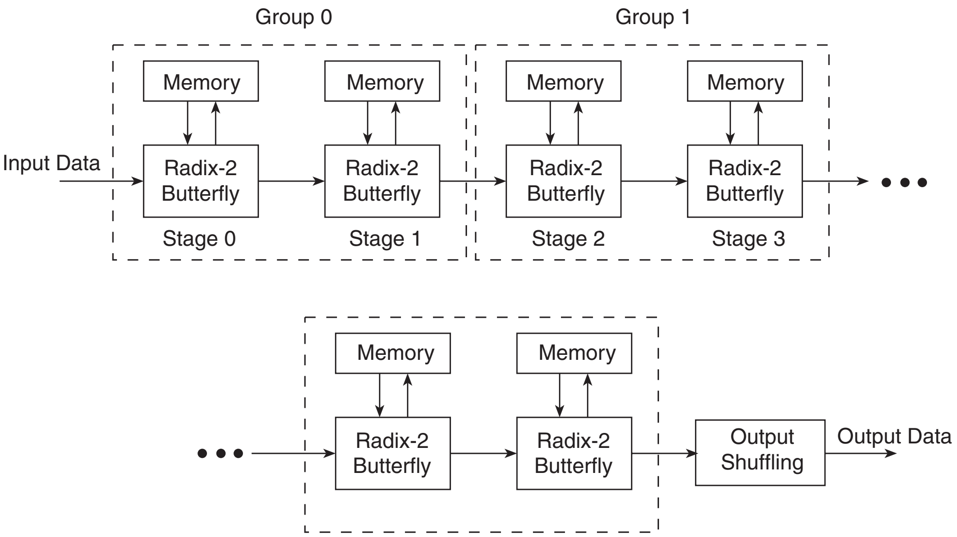


Figure 1: **Pipelined, Streaming I/O**

Radix-4, Burst I/O

With this solution, the FFT core uses one radix-4 butterfly processing engine and has two processes (Figure 2). One process is loading and/or unloading the data. The second process is calculating the transform. Data I/O and processing are not simultaneous. When the FFT is started, the data is loaded in. After a full frame has been loaded, the core will compute the FFT. When the computation has finished, the data can now be unloaded. During the calculation process, data loading and unloading cannot take place. The data loading and unloading processes can be overlapped if the data is unloaded in digit reversed order.

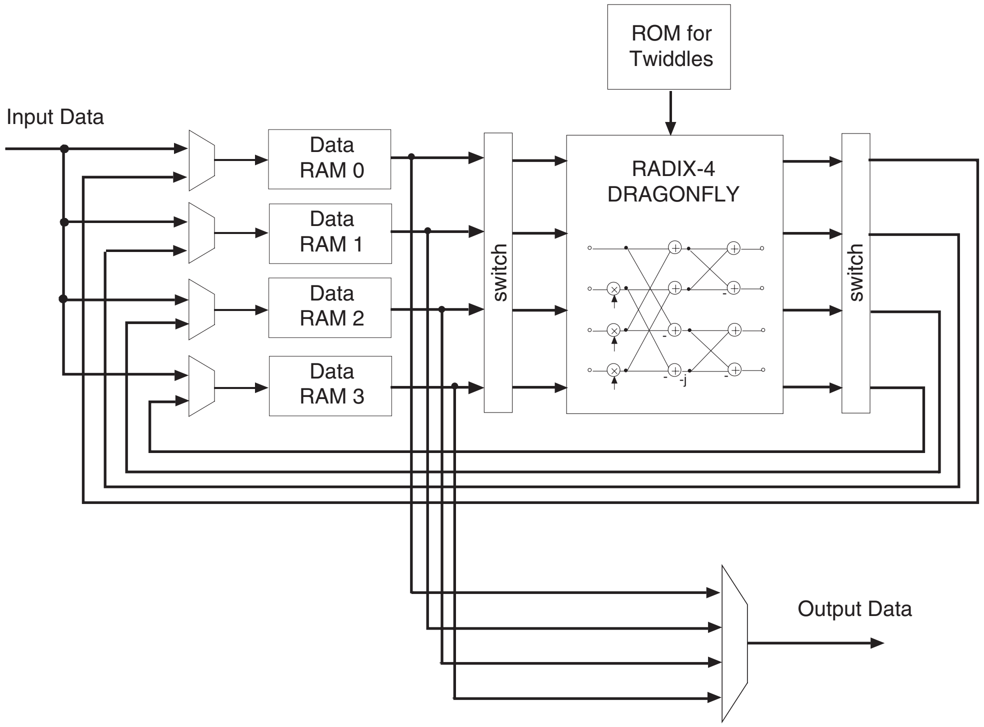


Figure 2: **Radix-4, Burst I/O**

This architecture has less resource usage than the Pipelined Streaming I/O architecture but a longer transform time and covers point sizes from 64 to 65536. All three arithmetic types are supported: unscaled, scaled, and block floating point. Phase factors can be stored in Block RAM or in Distributed RAM (for point sizes less than or equal to 1024).

Radix-2, Minimum Resources

This architecture uses one radix-2 butterfly processing engine (Figure 3) and has burst I/O like the radix-4 version. After a frame of data is loaded, the input data stream must halt until the transform calculation is completed. Then, the data can be unloaded. As with the Radix-4, Burst I/O architecture, data can be simultaneously loaded and unloaded if the results are presented in bit-reversed order. This solution supports point sizes $N = 8 – 65536$ and uses a minimum of block memories.

All three arithmetic types are supported (unscaled, scaled, and block floating point). Both the data memories and phase factor memories can be in either block memory or distributed memory (for point sizes less than or equal to 1024).

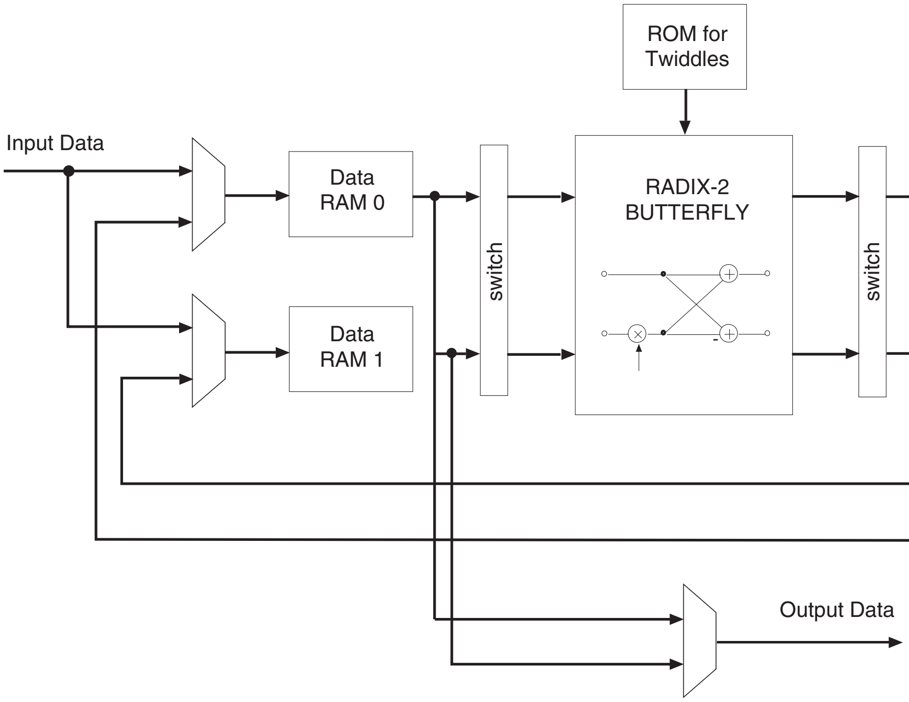


Figure 3: **Radix-2, Minimum Resources**

Core Symbol and Port Definitions

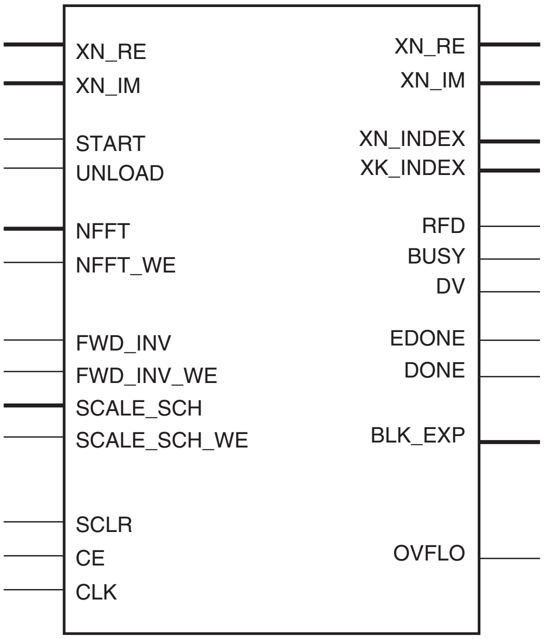


Figure 4: **Core Schematic Symbol**

Table 1: **Core Pinout**

Port Name	Port Width	Direction	Description
XN_RE	b_{xn}	Input	Input data bus: Real component (b_{xn} = 8, 12, 16, 20, 24)
XN_IM	b_{xn}	Input	Input data bus. Imaginary component (b_{xn} = 8, 12, 16, 20, 24)
START	1	Input	FFT start signal (Active High): START is asserted to begin the data loading and transform calculation (for the Burst I/O architectures). For continuous data processing, START will begin data loading, which proceeds directly to transform calculation and then data unloading.
UNLOAD	1	Input	Result unloading (Active High): For the Burst I/O architectures, UNLOAD will start the unloading of the results in normal order. The UNLOAD port is not necessary for the Pipelined, Streaming I/O architecture.
NFFT	5	Input	Point size of the transform: NFFT can be the size of the transform or any smaller point size. For example, a 1024-point FFT can compute point sizes 1024, 512, 256, and so on. The value of NFFT is $\log_2$ (point size). This port is optional for Pipelined, Streaming I/O architecture.

Table 1: **Core Pinout *(Continued)***

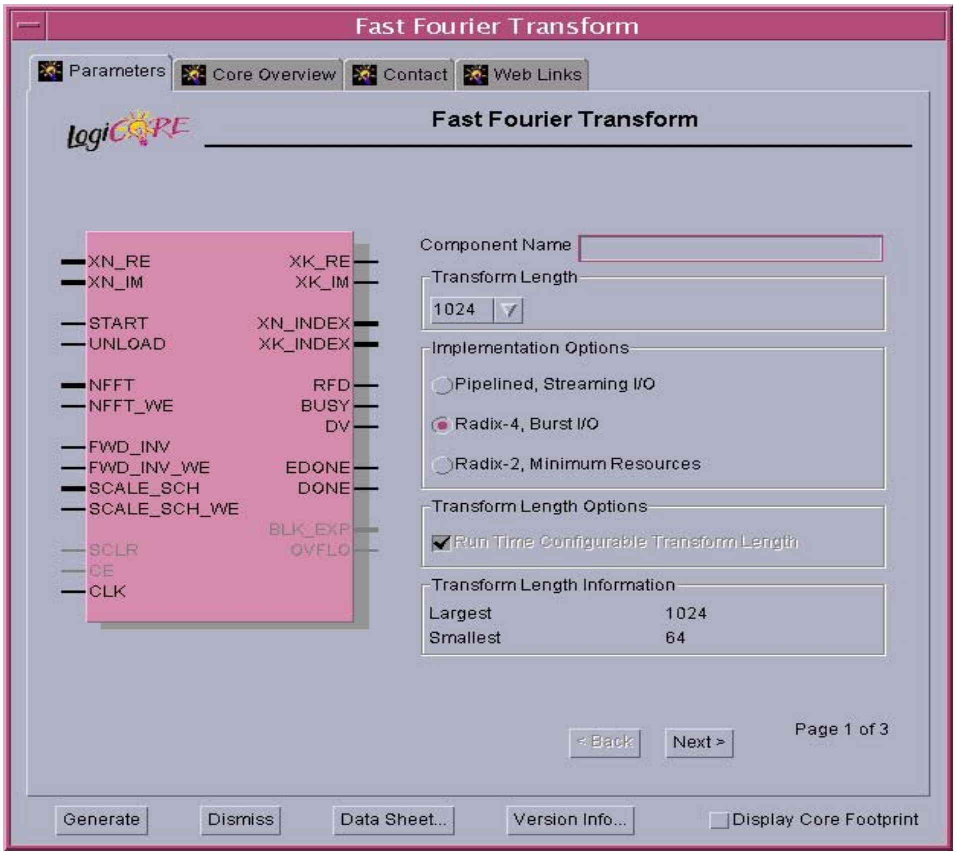
Port Name	Port Width	Direction	Description
NFFT_WE	1	Input	Write enable for NFFT (Active High): Asserting NFFT_WE will automatically cause the FFT core to stop all processes and to initialize the state of the core. This port is optional for Pipelined, Streaming I/O architecture.
FWD_INV	1	Input	Control signal that indicates if a forward FFT or an inverse FFT is performed. When FWD_INV=1, a forward transform is computed. If FWD_INV=0, an inverse transform is performed.
FWD_INV_WE	1	Input	Write enable for FWD_INV (Active High).
SCALE_SCH	$2 \times ceil\left(\frac{NFFT}{2}\right)$ for Streaming I/O and Radix-4 Burst I/O architectures or 2 x $NFFT$ for Radix-2 Minimum Resources	Input	Scaling schedule: For Radix-4 Burst I/O and Radix-2 minimum resources architecture, the scaling schedule is specified with two bits for each stage. The scaling can be specified as 3, 2, 1, or 0, which represents the number of bits to be shifted. An example scaling schedule for N =1024, Radix-4 burst I/O is [1 0 2 3 2]. For N=128, Radix-2 minimum resources, one possible scaling schedule is [1 1 1 1 0 1 2]. For Pipelined Streaming I/O architecture, the scaling schedule is specified with two bits for every pair of radix-2 stages. For example, a scaling schedule for N=256 could be [2 2 2 3]. When N is not a power of 4, the maximum bit growth for the last stage is one bit. For instance, [0 2 2 2 2] or [1 2 2 2 2] are valid scaling schedules for N=512, but [2 2 2 2 2] is invalid. The two MSBs of SCALE_SCH can only be 00 or 01. This port is only available with scaled arithmetic (not unscaled or block-floating point).
SCALE_SCH_WE	1	Input	Write enable for SCALE_SCH (Active High): This port is available only with scaled arithmetic.
SCLR	1	Input	Master reset (Active High): Optional port.
CE	1	Input	Clock enable (Active High): Optional port.
CLK	1	Input	Clock
XK_RE[(B-1):0]	b_{xk}	Output	Output data bus: Real component.
XK_IM[(B-1):0]	b_{xk}	Output	Output data bus: Imaginary component.
XN_INDEX	$\log_2$ (point size)	Output	Index of input data.
XK_INDEX	$\log_2$ (point size)	Output	Index of output data.
RFD	1	Output	Ready for data (Active High): RFD is High during the load operation.
BUSY	1	Output	Core activity indicator (Active High): This signal will go High while the core is computing the transform.
DV	1	Output	Data valid (Active High): This signal is High when valid data is presented at the output.

Table 1: **Core Pinout *(Continued)***

Port Name	Port Width	Direction	Description
EDONE	1	Output	Early done strobe (Active High): EDONE goes High one clock cycle immediately prior to DONE going active.
DONE	1	Output	FFT complete strobe (Active High): DONE will transition High for one clock cycle at the end of the transform calculation.
BLK_EXP	5	Output	Block exponent: Available only when block-floating point is used.
OVFLO	1	Output	Arithmetic overflow indicator (Active High): OVFLO will be High during result unloading if any value in the data frame overflowed. The OVFLO signal is reset at the beginning of a new frame of data. This port is optional and only available with scaled arithmetic.

Graphical User Interface

The FFT core graphical user interface (GUI) consists of three screens. Options for each screen are described below.



Component Name

Transform Length: 1024

Implementation Options
- ○ Pipelined, Streaming I/O
- ● Radix-4, Burst I/O
- ○ Radix-2, Minimum Resources

Transform Length Options
- ☑ Run Time Configurable Transform Length

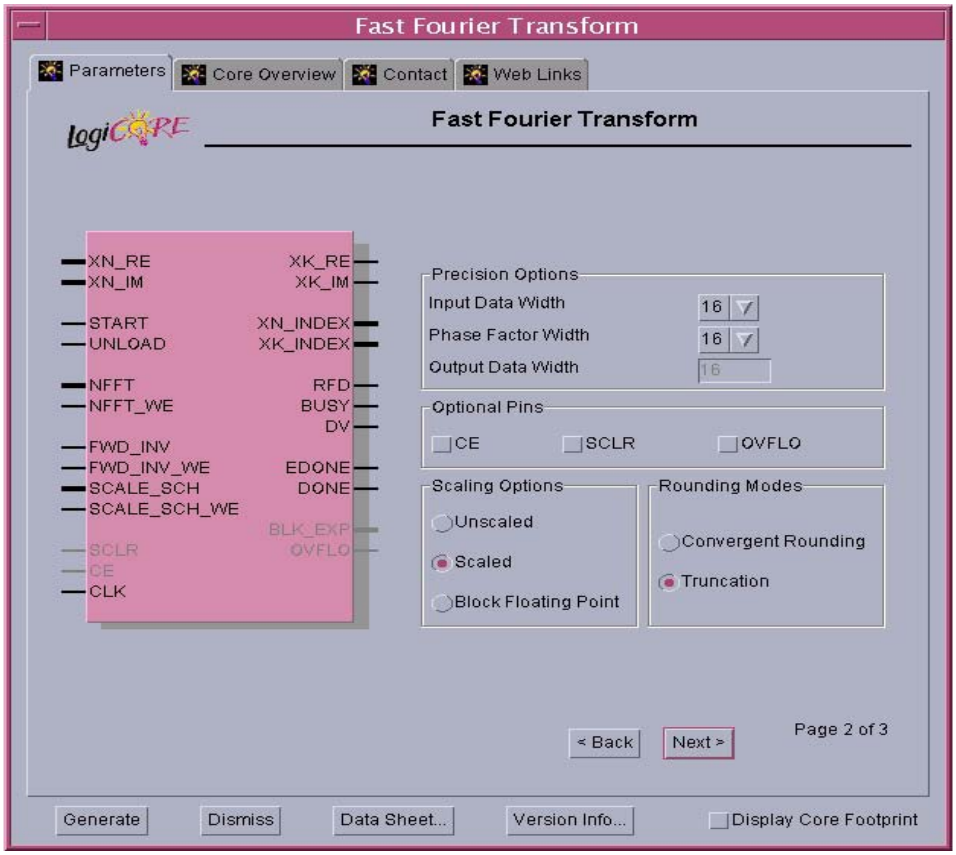
Transform Length Information

Largest	1024
Smallest	64

Figure 5: **XFFT Core GUI Main Window**

- **Component Name**: The name of the core component to be instantiated. The name must begin with a letter and be composed of the following characters: a to z, 0 to 9, and "_".
- **Transform Length**: Select the desired point size. All powers of two from 8 to 65536 are available.
- **Implementation Options**: Select an implementation option, as described in "Architecture Options" on page 4.
  - Radix-4, Streaming I/O, and Radix-2 Minimum Resources support point sizes 8 to 65536.
  - Radix-4 Burst I/O architecture supports point sizes 64 to 65536.
- **Transform Length Option**: Available only for the Radix-4, Streaming I/O architecture. Select the transform length to be run-time configurable or not. The core uses fewer logic resources when the transform length is not run-time configurable.
- **Transform Length Information**: When the transform length is run-time configurable, the core has the ability to reprogram the point size while the core is running; that is, the core can support the selected point size and any smaller point size. The GUI displays the supported point sizes based on

the Transform Length, Transform Length Option, and Implementation Option selected.



Precision Options
Input Data Width 16
Phase Factor Width 16
Output Data Width 16

Optional Pins
☐ CE ☐ SCLR ☐ OVFLO

Scaling Options
○ Unscaled
● Scaled
○ Block Floating Point

Rounding Modes
○ Convergent Rounding
● Truncation

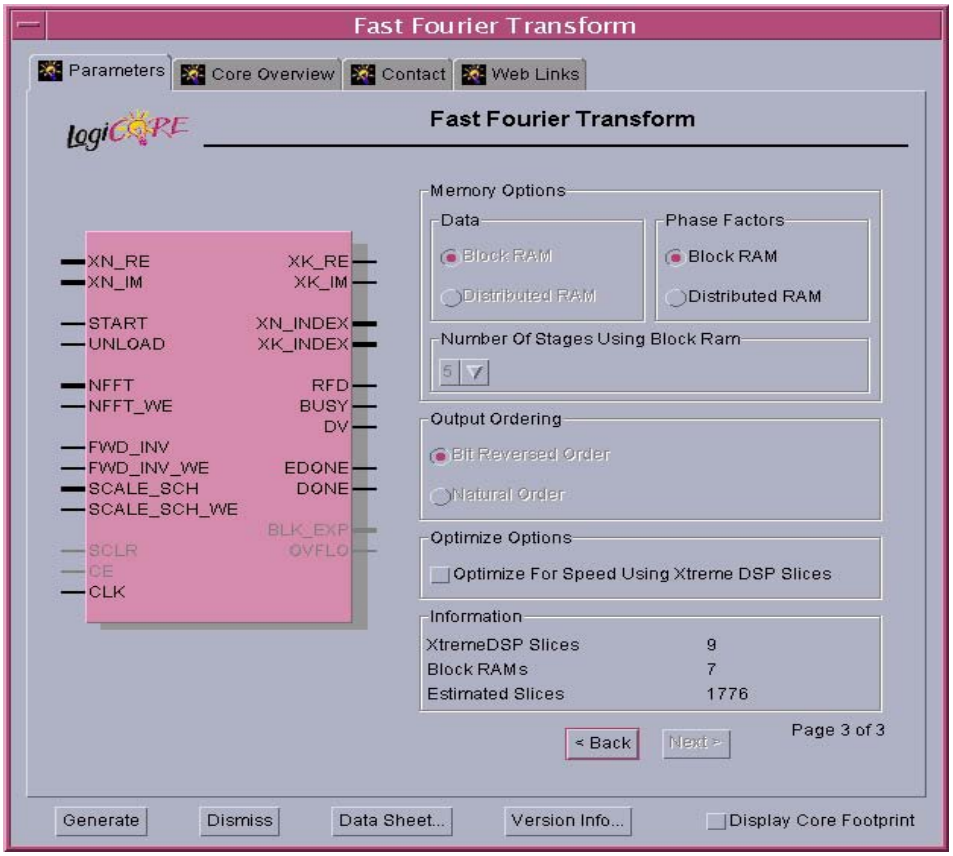
Page 2 of 3

☐ Display Core Footprint

Figure 6: **XFFT Core GUI Precision and Scaling Option Window**

- **Precision Options**
  - Input data width and phase factor data width can be 8, 12, 16, 20, or 24.
  - Output data width (which may be affected by the Scaling Option) is also displayed.
- **Optional Pins**
  - Clock Enable (CE), Synchronous Clear (SCLR), and Overflow (OVFLO) are optional pins. If no option is selected, some logic resources are saved.
- **Scaling Options**
  - Unscaled
  - Scaled
  - Block Floating Point. Note that Block Floating Point is unavailable with the Pipelined Streaming I/O architecture.
- **Rounding Modes**: At the output of the butterfly, the LSBs in the datapath need to be trimmed. These bits can be truncated or rounded using convergent rounding, an unbiased rounding scheme also known as round-to-nearest (even) number. When the fractional part of a number is equal to exactly one-half, convergent rounding rounds down if the number is odd (resulting in LSB=0), and rounds up if the number is even (resulting in LSB=1). Convergent rounding can be used to avoid

the DC bias that would be introduced by truncation.



Memory Options

Data

● Block RAM

○ Distributed RAM

Phase Factors

● Block RAM

○ Distributed RAM

Number Of Stages Using Block Ram: 5

Output Ordering

● Bit Reversed Order

○ Natural Order

Optimize Options

☐ Optimize For Speed Using Xtreme DSP Slices

Information

XtremeDSP Slices	9
Block RAMs	7
Estimated Slices	1776

☐ Display Core Footprint

Figure 7: **XFFT Core GUI Memory Options Window**

- **Memory Options**
  - For Pipelined Streaming I/O solution, the data can be partially stored in Block RAM and partially in Distributed RAM. The user can select the number of pipelined stages using Block RAM for data and phase factor storage.
  - For Radix-4 and Radix-2 burst I/O architecture, either Block RAM or Distributed RAM can be used for data and phase factor storage. Data storage in Block RAM is always available, and Distributed RAM data storage is supported by the Radix-2 Minimum Resource architecture. Phase factor storage is also always available in Block RAM. Phase factor storage can be in distributed RAM for all point sizes 1024 or under.
- **Output Ordering**: This output data can either be in bit reversed order or in natural order, and is available only for Pipelined Streaming I/O architecture.
- **Optimize Options**
  - For Radix-4 Burst I/O architecture in Virtex-4, the entire dragonfly can be computed in XtremeDSP slices. Selecting *Optimize For Speed Using XtremeDSP Slices* will allow a faster maximum clock speed at the cost of using more XtremeDSP slices. This checkbox is only available when the CORE Generator target architecture is Virtex-4.
- **Information**: Based on the options selected, this area displays the XtremeDSP slice count/embedded multiplier usage, block RAM numbers, and estimated slice count.

XCO Parameters

The following table describes valid entries for the xco parameters. Note that parameters are not case sensitive.

Table 2: **XCO Parameters**

XCO Parameter	Valid Values
rounding_modes	convergent_rounding truncation
ce	false true
scaling_options	scaled unscaled block_floating_point
memory_options_phase_factors	block_ram distributed_ram

output_data_width	If scaling_option is scaled or block_floating_point, then:
	output_data_width = input_data_width
	If scaling option is unscaled, then:
	output_data_width = input_data_width + $\log_2$ (transform_length) + 1

input_width	8, 12, 16, 20, 24
transform_length	8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536
run_time_configurable_transform_length	false true
memory_options_data	block_ram distributed_ram
number_of_stages_using_block_ram_for_data_and_phase_factors	0 - 12
output_ordering	bit_reversed_order natural_order
implementation_options	radix_4_burst_io pipelined_streaming_io radix_2_minimum_resources
component_name	Name must begin with a letter and be composed of the following characters: a to z, 0 to 9, and "_".
phase_factor_width	8, 12, 16, 20, 24
sclr	false true
ovflo	false true
optimize_for_speed_using_xtreme_dsp_slices	false true

Control Signals and Timing

Synchronous Clear

Asserting the Synchronous Clear (SCLR) pin results in resetting all output pins, internal counters, and state variables to their initial values. All pending load processes, transform calculations, and unload processes stop and are reinitialized. However, internal frame buffers retain their contents. NFFT will be set to the largest FFT point size permitted (the Transform Length value set in the GUI). The scaling schedule will be set to *1/N*. For the Radix-4 Burst I/O and Pipelined Streaming I/O architectures with a non-power-of-four point size, the last stage will have a scaling of 1, and the rest will have a scaling of 2.

Table 3: **Synchronous Clear Reset Values**

Signal	Initial / Reset Value
NFFT	maximum point size = *N*
FWD_INV	Forward = 1
SCALE_SCH	*1/N* [10 10... 10] for Radix-4 or Pipelined architecture when *N* is a power of 4. [01 10... 10] for Radix-4 or Pipelined architecture when *N* is not a power of 4. [01 01... 01] for Radix-2

Transform Size

The transform point size can be set through the NFFT port if the run-time configurable transform length option is selected. Valid settings and the corresponding transform sizes are provided in Table 4. If the NFFT value entered is too large, the core sets itself to the largest available point size (selected in the GUI). If the value is too small, the core sets itself to the smallest available point size: 64 for the Radix-4 Burst I/O architectures and 8 for the Radix-2 Minimum Resources and Pipelined Streaming I/O architecture.

NFFT values are read in on the rising clock edge when NFFT_WE is High. A new transform size re-times all current processes within the core, so every time a transform size is latched in, regardless of whether or not the new point size differs from the current point size, the core is internally reset. (Note that FWD_INV and SCALE_SCH are not reset.) Holding NFFT_WE High continues to reset the core on every clock cycle.

Table 4: **Valid NFFT Settings**

NFFT[4:0]	Transform size (*N*)
00011	8
00100	16
00101	32
00110	64
00111	128
01000	256
01001	512

Table 4: **Valid NFFT Settings *(Continued)***

NFFT[4:0]	Transform size (*N*)
01010	1024
01011	2048
01100	4096
01101	8192
01110	16384
01111	32768
10000	65536

Forward/Inverse and Scaling Schedule

The transform type (forward or inverse) and the scaling schedule can be set frame-by-frame without interrupting frame processing. The transform type can be set using the FWD_INV pin. Setting FWD_INV to 0 produces an inverse FFT, and setting FWD_INV to 1 creates the forward transform.

The scaling performed during successive stages can be set via the SCALE_SCH pin. The value of the SCALE_SCH bus is used as pairs of bits [... N4, N3, N2, N1, N0]: each pair representing the scaling value for the corresponding stage (except for Pipelined Streaming I/O architecture). In each stage, the data can be shifted by 0, 1, 2, or 3 bits, which corresponds to SCALE_SCH values of 00, 01, 10, and 11. Stages are computed starting with stage 0 as the two LSBs. For example, when $N = 1024$, [11 10 00 01 10] translates to a right shift by 2 for stage 0, shift by 1 for stage 1, no shift for stage 3, a shift of 2 in stage 3, and a shift of 3 for stage 4. The conservative schedule SCALE_SCH = [10 10 10 11 10] will completely avoid overflows in the Radix-4 architecture. For the Radix-2 architecture, the conservative scaling schedule of [01 01 01 10 01] will prevent overflow.

For the pipelined streaming architecture, consider every pair of adjacent radix-2 stages as a group. That is, group 0 contains stage 0 and 1, group 1 contains stage 2 and 3, and so forth. The value of the SCALE_SCH bus is also used as pairs of bits [... N4, N3, N2, N1, N0]. Each pair represents the scaling value for the corresponding group of two stages. In each group, the data can be shifted by 0, 1, 2, or 3 bits which corresponds to SCALE_SCH values of 00, 01, 10, and 11. Groups are computed starting with group 0 as the two LSBs. For example, when $N = 1024$, [11 10 00 01 10] translates to a right shift by 2 for group 0, shift by 1 for group 1, no shift for group 3, a shift of 2 in group 3, and a shift of 3 for group 4. The conservative schedule SCALE_SCH = [10 10 10 10 11] will completely avoid overflows in the Pipelined Streaming architecture. Note that when N is not a power of 4, the last group only contains one stage, and the maximum bit growth for the last group is one bit. Therefore, the two MSBs of scaling schedule can only be 00 or 01. A conservative scaling schedule for N=512 is SCALE_SCH=[01 10 10 10 11]

The user is allowed great flexibility to set the transform type (Forward / Inverse) and the scaling schedule. The FWD_INV and SCALE_SCH values are latched into temporary registers whenever the corresponding WE pins are High. FWD_INV_WE and SCALE_SCH_WE can be asserted at any time before the frame of data is loaded in. The core will read these temporary registers at XN_RE / XN_IM(0). These are the values that will be used for that frame of data. There is no way to alter those values once the transform calculation phase has started. Any WE assertions after XN_RE / XN_IM(0) affect the frame that follows.

Both the scaling schedule and the transform type are registered internally, so there is no need to hold these values on the pins. Also, if the scaling and transform type are constant through multiple frames, (that is, no new values are latched in) registered values will apply for successive frames. The scaling schedule and transform type are not reset when NFFT_WE is asserted.

The initial value and reset value of FWD_INV is forward = 1. The scaling schedule is set to *1/N*. That translates to [10 10 10 10... 10] for the Radix-4 architectures, and [01 01...01] for the Radix-2 architecture. The core will read in (2*number of stages) bits for the scaling schedule. So, when the point size decreases, the leftover MSBs will be ignored.

Overflow

The Overflow (OVFLO) signal (used only with fixed-point scaling) will be High during unloading if any point in the data frame overflowed. For the Radix-4 Burst I/O and Radix-2 Minimum Resources architectures, The OVFLO signal will go High as soon as an overflow occurs during the computation and remain High during the entire time the frame is unloading.

Block Exponent

The Block Exponent (BLK_EXP) signal (used only with the block-floating point option) contains the block exponent. This signal will be valid during the unloading of the data frame. The value present on the port represents the total number of bits the data was scaled during the transform. For example, if BLK_EXP has a value of 00101b = 5, this means the output data (XK_RE, XK_IM) was scaled by 5 bits (shifted right by 5 bits), or in other words, was divided by 32, in order to fully utilize the available dynamic range of the output data path.

Timing for Pipelined Streaming I/O

Asserting START starts the data loading phase, which will immediately flow into the transform calculation phase and then the data unloading phase. Pulsing START once will allow the transform calculation for a single frame. Pulsing START every N clock cycles will allow continuous data processing. Alternatively, holding START High will also allow continuous data processing (Figure 8). If START is pulsed at any time during the loading of the current frame (when XN_INDEX = 0 to N-1), it signals to the core that another frame needs to be loaded after the current one. If no NFFT_WE, FWD_INV_WE, or SCALE_SCH_WE were asserted before the initial START, then the defaults will be used. This architecture can also support non-continuous data streams (Figure 9). Simply assert START at any time to begin data loading. After the data frame is loaded, the core will proceed to calculate the transform and then output the results.

Input data (XN_RE, XN_IM) corresponding to a certain XN_INDEX should arrive four clock cycles later than the XN_INDEX it matches (Figure 10). In this way, XN_INDEX can be used to address external memory or a frame buffer storing the input data. RFD will remain High with XN_INDEX during the loading phase when it is valid to input data.

BUSY will go High while the core is calculating the transform. DONE will go High during the last cycle of the calculation phase. EDONE will go High one cycle before that. The cycle after DONE goes High, the core begins unloading. During the unloading phase, while valid output results are present on XK_RE/ XK_IM, DV (Data Valid) will be High. During unloading, XK_INDEX will correspond to the XK_RE/XK_IM being presented.

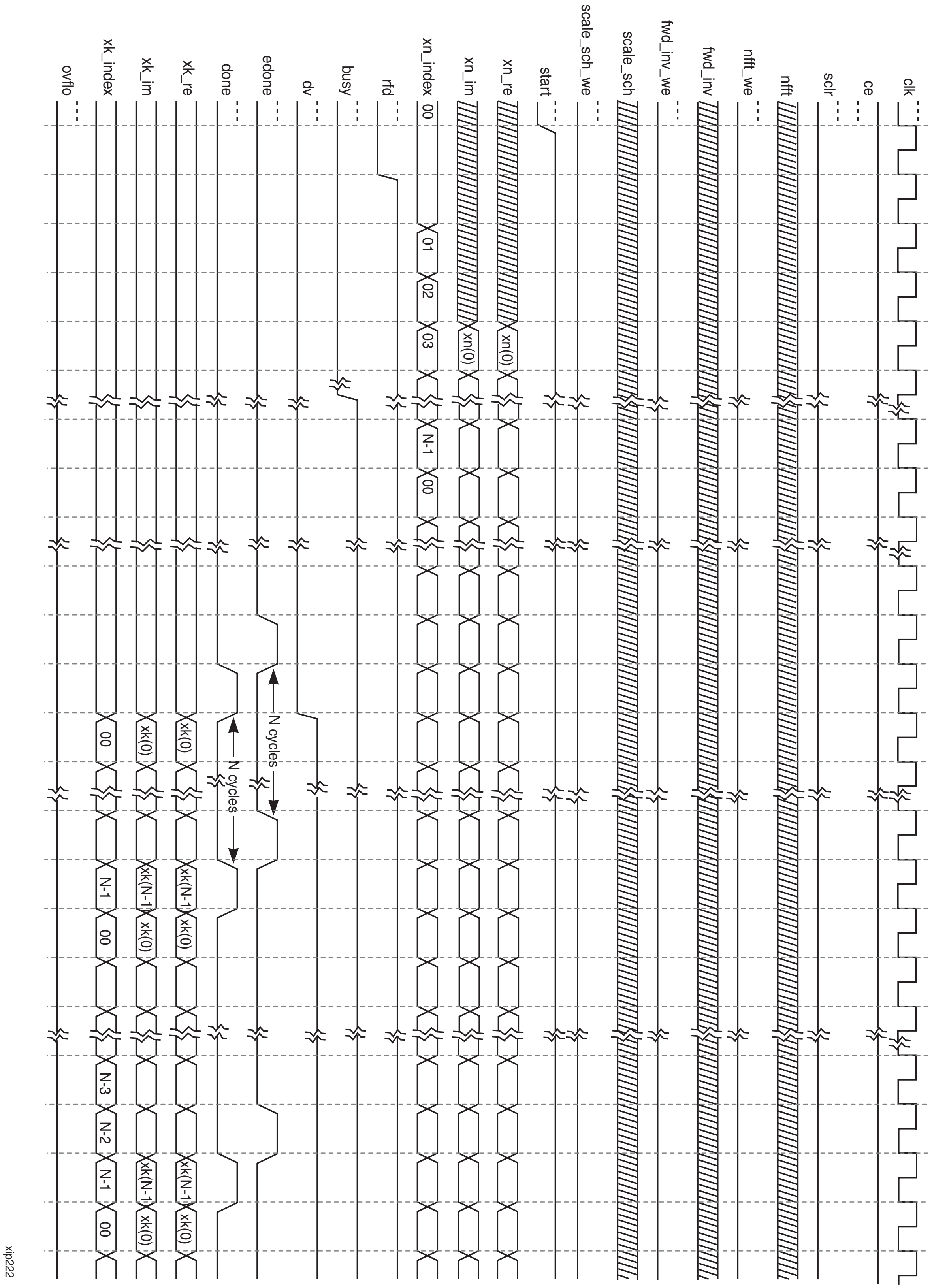


Figure 8: **Timing for Continuous Streaming Data**

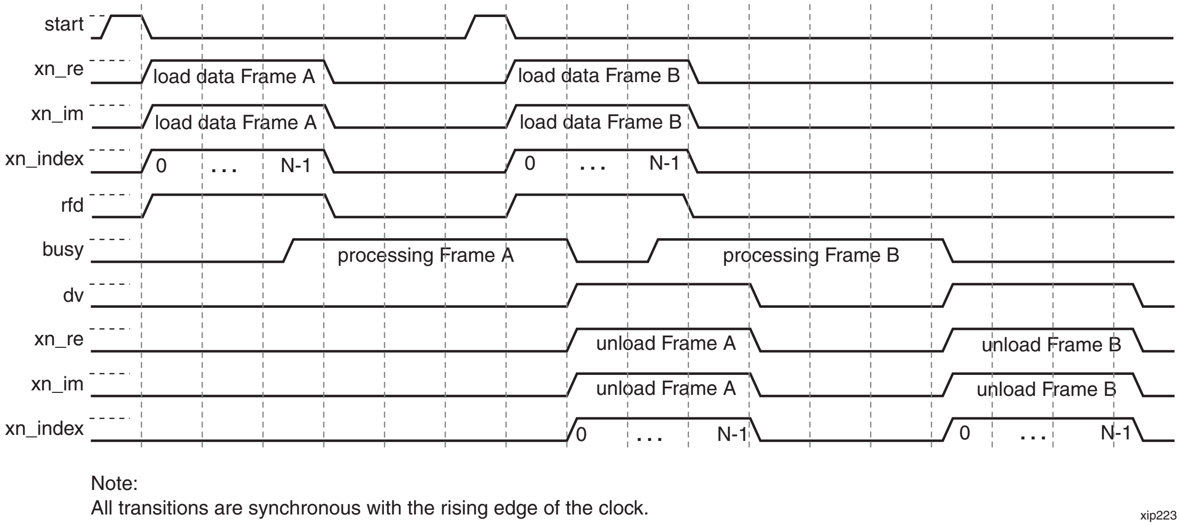


Figure 9: **Timing for Non-Continuous Data Stream**

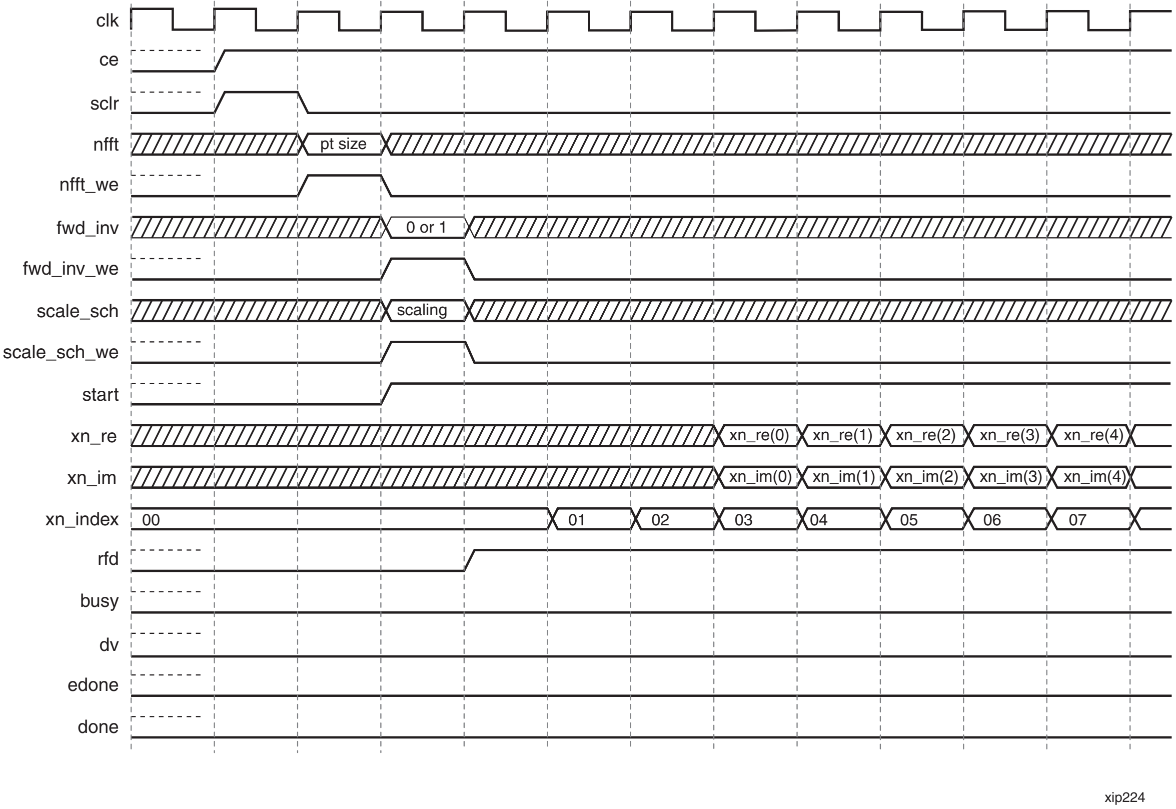


Figure 10: **Beginning of Data Frame**

Timing for Radix-4 Burst I/O and Radix-2 Minimum Resources

The START signal begins the data loading phase, which leads directly to the calculation phase. As with the Pipeline Streaming I/O architecture, START being pulsed at any time during the loading of the current frame (when XN_INDEX = 0 to N-1) signals the core that another frame will be processed after the current one.

Input data (XN_RE, XN_IM) corresponding to a certain XN_INDEX should arrive four clock cycles later than the XN_INDEX it matches (Figure 10). In this way, XN_INDEX can be used to address external memory or a frame buffer storing the input data. RFD will remain High with XN_INDEX during the loading phase when it is valid to input data.

BUSY will go High while the core is calculating the transform. DONE will go High during the last cycle of the calculation phase. EDONE will go High one cycle before that.

After START is asserted and the data is loaded and processed, two options are available to unload data:

- To output the data in natural order, UNLOAD should be asserted (Figure 11).
- To output data in bit / digit reversed order, the user can assert START again. While the next frame of data is loaded, the results will be presented in bit / digit reversed order at the same time (Figure 12).

DV remains High during data unloading in both cases.

START and UNLOAD can be tied High (Figure 13). In this case, the core will alternate between load / calculate and unload. The core will continuously load, process, and unload data.

There is a latency of *k* CLK cycles after triggering an unload before the output data XK_RE / XK_IM is presented (*k* = 7 for Radix-4 Burst I / O and *k* = 5 for Radix-2 Minimum Resources). If the unload process is triggered by pulsing UNLOAD after transform calculations are completed and DONE has gone High, then there is a delay of *k* clock cycles after UNLOAD before XK_RE(0) and XK_IM(0) appear on their ports. If UNLOAD is pulsed during transform calculation before DONE has gone High, then XK_RE and XK_IM will appear *k* clock cycles after DONE has gone High which means the calculation is complete. If bit / digit reversed unloading is triggered by pulsing START, then XK_RE and XK_IM will appear *k* clock cycles after START (Figure 14).

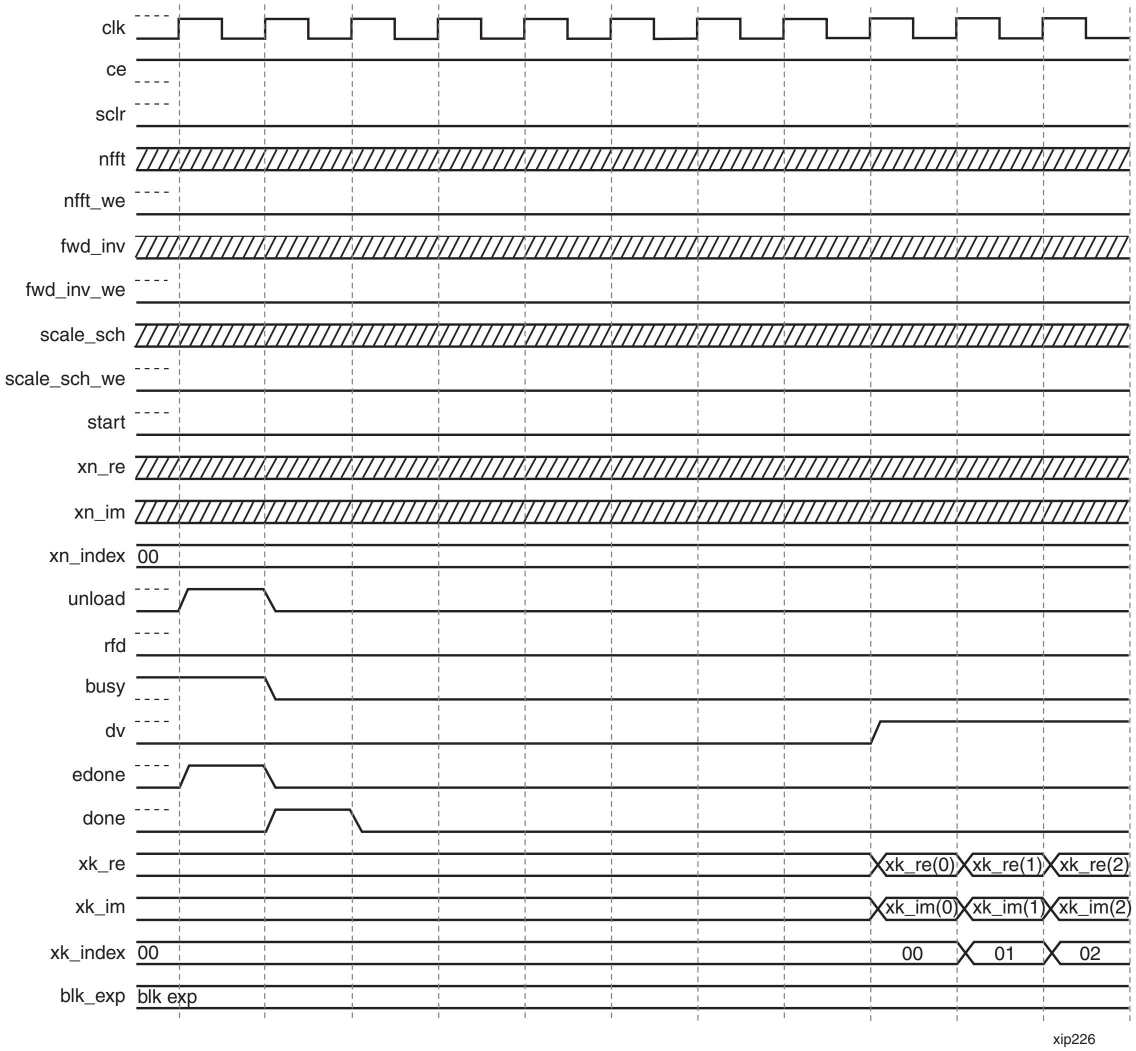


Figure 11: **Unload Output Results in Natural Order**

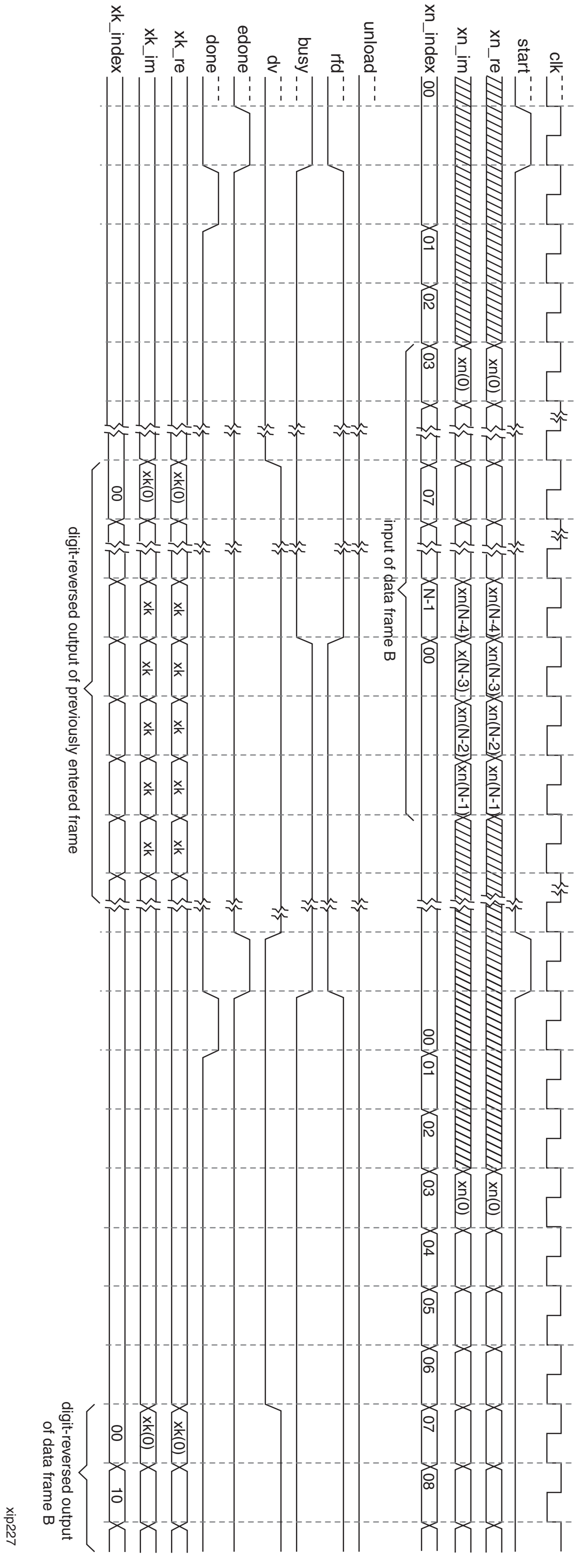
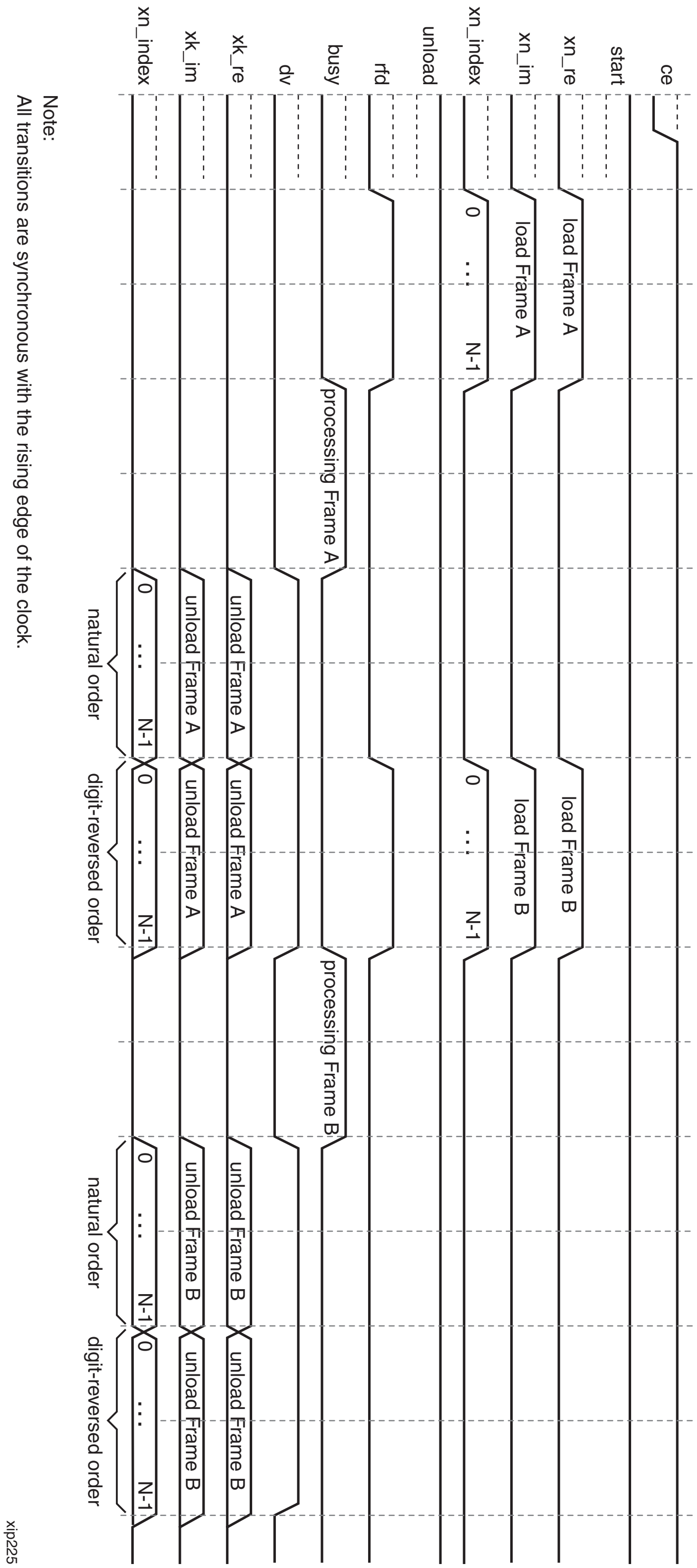


Figure 12: **Unload Results in Bit/Digit Reversed Order**



Note:
All transitions are synchronous with the rising edge of the clock.

Figure 13: **Timing for Burst I/O Solutions**

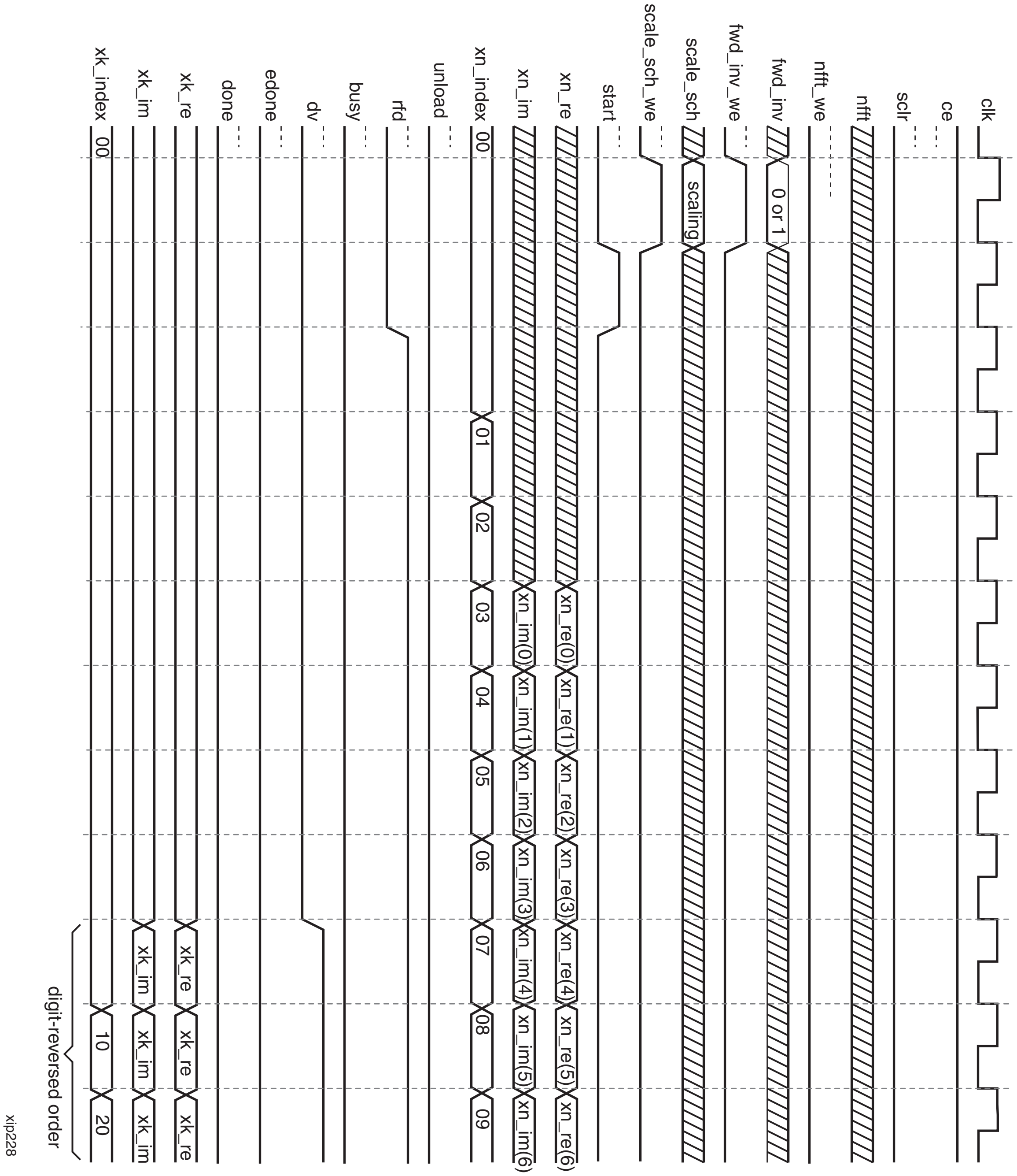


Figure 14: **Unloading Results in Bit/Digit Reversed Order**

Performance and Resource Usage

The following tables list the resource usage and transform time for a selected set of parameters. For a variety of point sizes and input data / phase factor bit widths, the slice count, Block RAM count, and XtremeDSP slice / embedded hardware multiplier count is listed. Also, the maximum clock frequency is listed next to the transform time. The transform time is described by the number of clock cycles or number of microseconds needed to perform the transform calculation. (Time for loading and unloading the data is not included in that value.) For the non-streaming I / O architectures, an additional column is provided, which contains the number of clock cycles and microseconds needed to load and calculate the transform.

These numbers were obtained using scaled fixed-point arithmetic with truncation after the butterflies. Data and phase factor storage is in Block RAM for the burst I / O architectures. For the streaming I / O architecture, the default value in the GUI is used for distributing data and phase factor storage between Block RAM and Distributed RAM. The slice counts are approximately the same if using block floating point arithmetic.

Table 5: **Performance and Resource Utilization for the Virtex-II Family: Pipelined, Streaming I/O**

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Device
								Clock Cycles	Time (µs)		
						-6	-5		-6	-5	
64	16	16	1280	1	8	188	169	64	0.34	0.38	2v250
256	16	16	1769	4	12	188	143	256	1.36	1.79	2v1000
1024	16	16	2294	7	16	188	169	1024	5.45	6.06	2v1500
2048	16	16	2545	10	20	188	143	2048	10.89	14.32	2v3000
8192	16	16	3141	24	24	188	143	8192	43.57	57.29	2v4000
65536	16	16	4285	150	28	n/a	169	65536	n/a	387.79	2v8000
64	24	24	2463	2	24	148	134	64	0.43	0.48	2v2000
1024	24	24	4526	12	48	148	134	1024	6.92	7.64	2v6000
8192	24	24	6252	37	72	148	134	8192	55.35	61.13	2v6000

Note: ISE 6.3i speed file - Production 1.118 2004-08-11.

Table 6: **Performance and Resource Utilization for the Virtex-II Family: Radix-4, Burst I/O**

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycles	Time (µs)		
						-6	-5		-6	-5		-6	-5	
64	16	16	1331	8	9	195	149	91	0.47	0.61	155	0.79	1.04	2v1000
256	16	16	1411	7	9	195	149	289	1.48	1.94	545	2.79	3.66	2v1000
1024	16	16	1498	7	9	195	149	1319	6.76	8.85	2343	12.02	15.72	2v1000
2048	16	16	1649	7	9	195	149	3117	15.98	20.92	5165	26.49	34.66	2v1000
8192	16	16	1753	22	9	195	149	14387	73.78	96.56	22579	115.79	151.54	2v3000
65536	16	16	2167	158	9	n/a	175	131129	n/a	749.31	196665	n/a	1123.80	2v8000
64	24	24	2669	12	36	152	138	97	0.64	0.70	161	1.06	1.17	2v3000

Table 6: **Performance and Resource Utilization for the Virtex-II Family: Radix-4, Burst I/O** ***(Continued)***

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
						-6	-5	Clock Cycles	Time (μs) -6	Time (μs) -5	Clock Cycles	Time (μs) -6	Time (μs) -5	
1024	24	24	2968	11	36	152	138	1321	8.69	9.57	2345	15.43	16.99	2v3000
8192	24	24	3107	33	36	152	138	14389	94.66	104.27	22581	148.56	163.63	2v6000

Note: ISE 6.3i speed file - Production 1.118 2004-08-11.

Table 7: **Performance and Resource Utilization for the Virtex-II Family: Radix-2, Minimum Resources**

Point Size	Input Data Width	Phase Factor Width	Slices(2)	Block RAM (2)	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
						-6	-5	Clock Cycles	Time (μs) -6	Time (μs) -5	Clock Cycle	Time (μs) -6	Time (μs) -5	
64	16	16	530/709	3/0	3	188	169	265	1.41	1.57	329	1.75	1.95	2v250
256	16	16	567/1365	3/0	3	192	166	1079	5.62	6.50	1335	6.95	8.04	2v250
1024	16	16	630/3053	3/0	3	176	150	5187	29.47	34.58	6211	35.29	41.41	2v250
2048	16	16	717	5	3	179	156	11338	63.34	72.68	13386	74.78	85.81	2v250
8192	16	16	808	18	3	166	142	53334	321.29	375.59	61526	370.64	433.28	2v1500
65536	16	16	1062	130	3	168	147	524497	3122.01	3568.01	590033	3512.10	4013.83	2v6000
64	24	24	973	5	12	152	138	277	1.82	2.01	341	2.24	2.47	2v1000
1024	24	24	1175	5	12	152	138	5190	34.14	37.61	6214	40.88	45.03	2v1000
8192	24	24	1276	25	12	152	138	53336	350.89	386.49	61528	404.79	445.86	2v3000

Notes:
1. ISE 6.3i speed file - Production 1.118 2004-08-11.
2. Second number is if input data and phase factors are stored in distributed memory.

Table 8: **Performance and Resource Utilization for the Virtex-II Pro Family: Pipelined, Streaming I/O**

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Device
						-7	-6	Clock Cycles	Time (μs) -7	Time (μs) -6	
64	16	16	1279	1	8	214	198	64	0.30	0.32	2vp4
256	16	16	1768	4	12	214	198	256	1.20	1.29	2vp7
1024	16	16	2284	7	16	214	198	1024	4.79	5.17	2vp20
2048	16	16	2545	10	20	214	198	2048	9.57	10.34	2vp20
8192	16	16	3120	24	24	214	198	8192	38.28	41.37	2vp30
65536	16	16	4252	150	28	214	198	65536	306.24	330.99	2vp70
64	24	24	2462	2	24	165	153	64	0.39	0.42	2vp20

Table 8: **Performance and Resource Utilization for the Virtex-II Pro Family: Pipelined, Streaming I/O *(Continued)***

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Device
								Clock Cycles	Time (µs)		
						-7	-6		-7	-6	
1024	24	24	4518	12	48	165	153	1024	6.21	6.69	2vp30
8192	24	24	6238	37	72	165	153	8192	49.65	53.54	2vp50

Note: ISE 6.3i speed file - Production 1.88 2004-08-11.

Table 9: **Performance and Resource Utilization for the Virtex-II Pro Family: Radix-4, Burst I/O**

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycles	Time (µs)		
						-7	-6		-7	-6		-7	-6	
64	16	16	1330	8	9	223	205	91	0.41	0.44	155	0.70	0.76	2vp7
256	16	16	1410	7	9	223	205	289	1.30	1.41	545	2.44	2.66	2vp7
1024	16	16	1492	7	9	223	205	1319	5.91	6.43	2343	10.51	11.43	2vp7
2048	16	16	1637	7	9	223	205	3117	13.98	15.20	5165	23.16	25.20	2vp7
8192	16	16	1723	22	9	223	205	14387	64.52	70.18	22579	101.25	110.14	2vp20
65536	16	16	2132	158	9	223	205	131129	588.02	639.65	196665	881.91	959.34	2vp70
64	24	24	2665	12	36	170	158	97	0.57	0.61	161	0.95	1.02	2vp30
1024	24	24	2948	11	36	170	158	1321	7.77	8.36	2345	13.79	14.84	2vp30
8192	24	24	3076	33	36	170	158	14389	84.64	91.07	22581	132.83	142.92	2vp50

Note: ISE 6.3i speed file - Production 1.88 2004-08-11.

Table 10: **Performance and Resource Utilization for the Virtex-II Pro Family: Radix-2, Minimum Resources**

Point Size	Input Data Width	Phase Factor Width	Slices(2)	Block RAM(2)	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycles	Time (µs)		
						-7	-6		-7	-6		-7	-6	
64	16	16	531/709	3/0	3	223	200	265	1.19	1.33	329	1.48	1.65	2vp4
256	16	16	567/1365	3/0	3	223	205	1079	4.84	5.26	1335	5.99	6.51	2vp4
1024	16	16	630/3053	3/0	3	207	186	5187	25.06	27.89	6211	30.00	33.39	2vp4
2048	16	16	716	5	3	216	193	11338	52.49	58.75	13386	61.97	69.36	2vp4
8192	16	16	774	18	3	194	174	53334	274.92	306.52	61526	317.14	353.60	2vp7
65536	16	16	1061	130	3	201	181	524497	2609.44	2897.77	590033	2935.49	3259.85	2vp70
64	24	24	971	5	12	170	158	277	1.63	1.75	341	2.01	2.16	2vp7

Table 10: **Performance and Resource Utilization for the Virtex-II Pro Family: Radix-2, Minimum Resources *(Continued)***

Point Size	Input Data Width	Phase Factor Width	Slices(2)	Block RAM(2)	MULT 18x18	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycles	Time (µs)		
						-7	-6		-7	-6		-7	-6	
1024	24	24	1154	5	12	170	158	5190	30.53	32.85	6214	36.55	39.33	2vp7
8192	24	24	1245	25	12	170	158	53336	313.74	337.57	61528	361.93	389.42	2vp20

Notes:

1. ISE 6.3i speed file - Production 1.88 2004-08-11.
2. Second number is if input data and phase factors are stored in distributed memory.

Table 11: **Performance and Resource Utilization for the Virtex-4 Family: Pipelined, Streaming I/O**

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	XtremeDSP Slices	Max. Clock Frequency (MHz)		Transform Time			Device
								Clock Cycles	Time (µs)		
						-11	-10		-11	-10	
64	16	16	1126	1	8	335	299	64	0.19	0.21	4vsx25
256	16	16	1539	4	12	335	299	256	0.76	0.86	4vsx25
1024	16	16	1975	7	16	335	299	1024	3.06	3.42	4vsx25
2048	16	16	2161	10	20	315	281	2048	6.50	7.29	4vsx25
8192	16	16	2664	24	24	315	281	8192	26.01	29.15	4vsx25
65536	16	16	3713	150	28	315	281	65536	208.05	233.22	4vsx35
64	24	24	2156	2	24	269	242	64	0.24	0.26	4vsx25
1024	24	24	3887	12	48	256	230	1024	4.00	4.45	4vsx25
8192	24	24	5320	37	72	256	230	8192	32.00	35.62	4vsx35

Note: ISE 6.3i speed file - Preview 1.47 2004-08-11.

Table 12: **Performance and Resource Utilization for the Virtex-4 Family: Radix-4, Burst I/O**

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	Xtreme DSP Slices	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycles	Time (µs)		
						-11	-10		-11	-10		-11	-10	
64	16	16	1291	8	9	303	273	100	0.33	0.37	164	0.54	0.60	4vsx25
64	16	16	893	8	28	410	360	100	0.24	0.28	164	0.40	0.46	4vsx25
256	16	16	1367	7	9	303	273	292	0.96	1.07	548	1.81	2.01	4vsx25
256	16	16	974	7	28	421	370	292	0.69	0.79	548	1.30	1.48	4vsx25
1024	16	16	1445	7	9	303	273	1322	4.36	4.84	2346	7.74	8.59	4vsx25
1024	16	16	1052	7	28	421	370	1322	3.14	3.57	2346	5.57	6.34	4vsx25
2048	16	16	1555	7	9	303	273	3120	10.30	11.43	5168	17.06	18.93	4vsx25
2048	16	16	1162	7	28	315	281	3120	9.90	11.10	5168	16.41	18.39	4vsx25

Table 12: **Performance and Resource Utilization for the Virtex-4 Family: Radix-4, Burst I/O** ***(Continued)***

Point Size	Input Data Width	Phase Factor Width	Slices	Block RAM	Xtreme DSP Slices	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycles	Time (µs)		
						-11	-10		-11	-10		-11	-10	
8192	16	16	1643	22	9	303	273	14390	47.49	52.71	22582	74.53	82.72	4vsx25
8192	16	16	1251	22	28	315	281	14390	45.68	51.21	22582	71.69	80.36	4vsx25
65536	16	16	2083	158	9	303	273	131132	432.78	480.34	196668	649.07	720.40	4vsx35
65536	16	16	1771	158	28	342	305	131132	383.43	429.94	196668	575.05	644.81	4vsx35
64	24	24	2306	12	36	248	224	124	0.50	0.55	188	0.76	0.84	4vsx25
64	24	24	1597	12	64	410	360	118	0.29	0.33	182	0.44	0.51	4vsx35
1024	24	24	2528	11	36	248	224	1330	5.36	5.94	2354	9.49	10.51	4vsx25
1024	24	24	1824	11	64	256	230	1328	5.19	5.77	2352	9.19	10.23	4vsx35
8192	24	24	2677	33	36	248	224	14398	58.06	64.28	22590	91.09	100.85	4vsx35
8192	24	24	1972	33	64	256	230	14398	56.23	62.59	22588	88.23	98.21	4vsx35

Note: ISE 6.3i speed file - Preview 1.47 2004-08-11.

Table 13: **Performance and Resource Utilization for the Virtex-4 Family: Radix-2, Minimum Resources**

Point Size	Input Data Width	Phase Factor Width	Slices(2)	Block RAM(2)	Xtreme DSP Slices	Max. Clock Frequency (MHz)		Transform Time			Data Load + Transform Time			Device
								Clock Cycles	Time (µs)		Clock Cycle	Time (µs)		
						-11	-10		-11	-10		-11	-10	
64	16	16	515/728	3/0	3	330	297	283	0.86	0.95	347	1.05	1.17	4vsx25
256	16	16	564/1270	3/0	3	330	297	1082	3.28	3.64	1338	4.05	4.51	4vsx25
1024	16	16	634/3462	3/0	3	308	269	5190	16.85	19.29	6214	20.18	23.10	4vsx25
2048	16	16	671	5	3	308	269	11341	36.82	42.16	13389	43.47	49.77	4vsx25
8192	16	16	733	18	3	276	243	53337	193.25	219.49	61529	222.93	253.21	4vsx25
65536	16	16	1023	130	3	290	256	524545	1808.78	2049.00	590081	2034.76	2305.00	4vsx35
64	24	24	860	5	12	266	240	331	1.24	1.38	395	1.48	1.65	4vsx25
1024	24	24	994	5	12	256	230	5199	20.31	22.60	6223	24.31	27.06	4vsx25
8192	24	24	1128	25	12	256	230	53345	208.38	231.93	61537	240.38	267.55	4vsx25

Notes:

1. ISE 6.3i speed file - Production 1.118 2004-08-11.
2. Second number is if input data and phase factors are stored in distributed memory.

The dynamic range characteristics are shown by performing *slot noise* tests. First, a frame of complex Gaussian noise data samples is created. An FFT is taken to acquire the spectrum of the data. To create the slot, a range of frequencies in the spectra is set to zero. To create the input slot noise data frame, the inverse FFT is taken, then the data is quantized to use the full input dynamic range. Because of the quantization, if a perfect FFT is done on the frame, the noise floor on the bottom of the slot will be nonzero. The Input Data figures, which basically represent the dynamic range of the input format, display this.

This slot noise input data frame is fed to the FFT core to see how shallow the slot becomes due to the finite precision arithmetic. The depth of the slot shows the dynamic range of the FFT.

Figures 15 through 24 show the effect of input data width on the dynamic range. All FFTs have the same bit width for both data and phase factors. Block floating point arithmetic is used with rounding after the butterfly. The figures show the input data slot and the output data slot for bit widths of 24, 20, 16, 12, and 8.

Figure Top x-ref 15

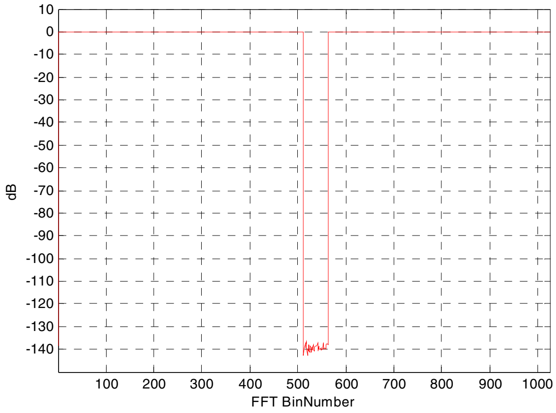


Figure 15: **Input Data: 24 Bits**

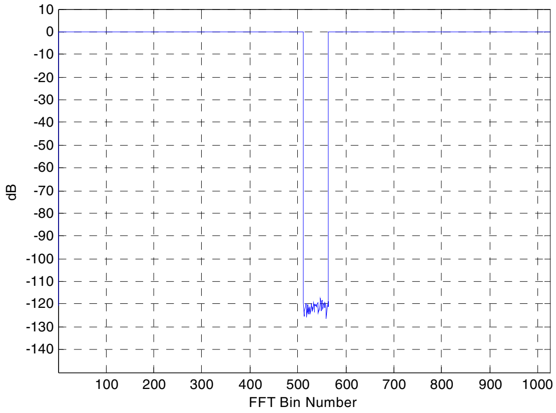


Figure 16: **FFT Core Results: 24 Bits**

Figure Top x-ref 17

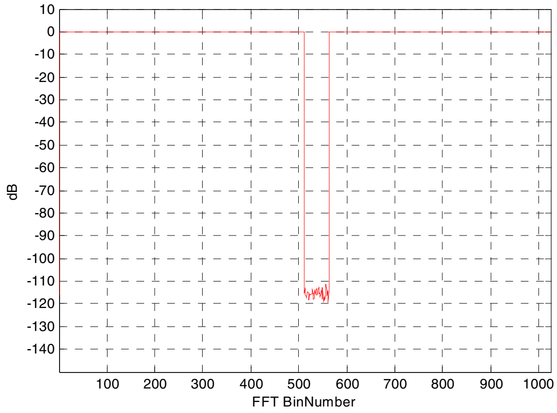


Figure 17: **Input Data: 20 Bits**

Figure Top x-ref 18

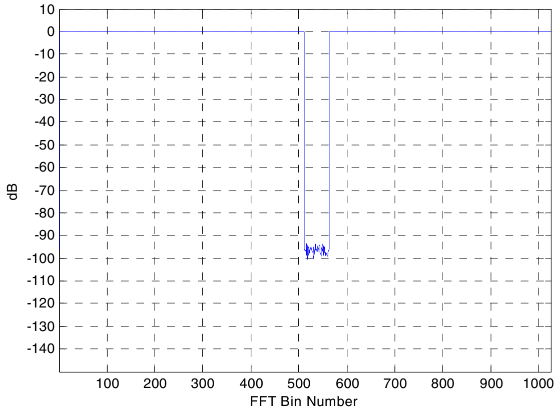


Figure 18: **FFT Core Results: 20 Bits**

Figure Top x-ref 19

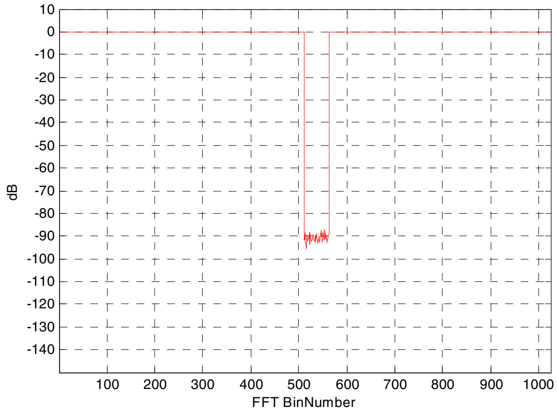


Figure 19: **Input Data: 16 Bits**

Figure Top x-ref 20

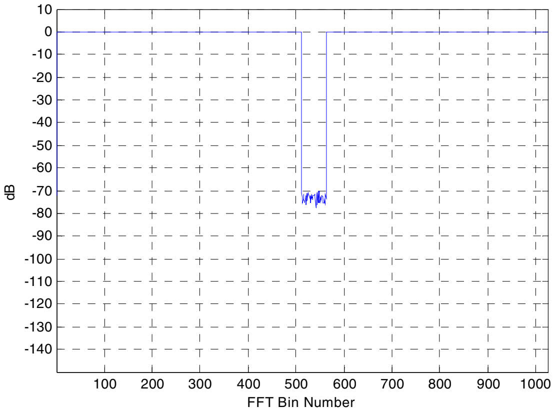


Figure 20: **FFT Core Results: 16 Bits**

Figure Top x-ref 21

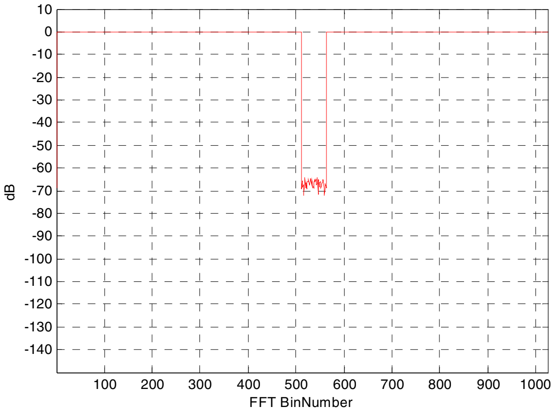


Figure 21: **Input Data: 12 Bits**

Figure Top x-ref 22

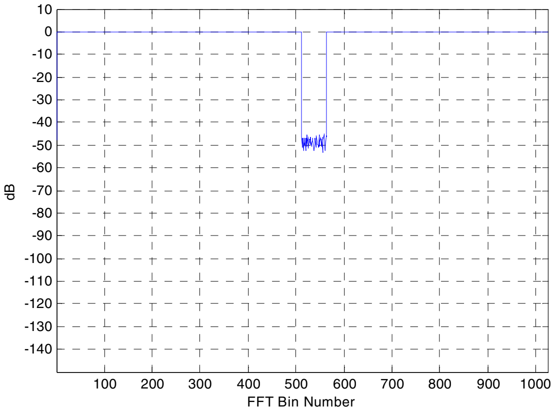


Figure 22: **FFT Core Results: 12 Bits**

Figure Top x-ref 23

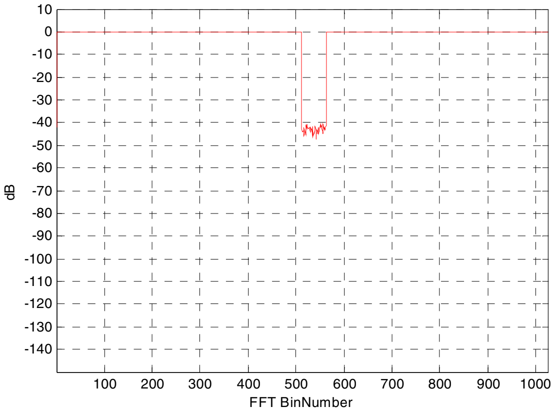


Figure 23: **Input Data: 8 Bits**

Figure Top x-ref 24

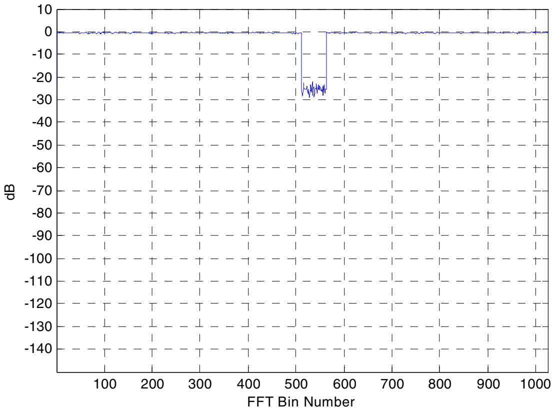


Figure 24: **FFT Core Results: 8 Bits**

There are several options available that also affect the dynamic range. Consider the arithmetic type used.

Figures 25, 26, and 27 display the results of using unscaled, scaled (scaling of 1 / 1024), and block floating point. All three FFTs are 1024 point, Radix-4 transforms with 16-bit input, 16-bit phase factors, and convergent rounding.

Figure Top x-ref 25

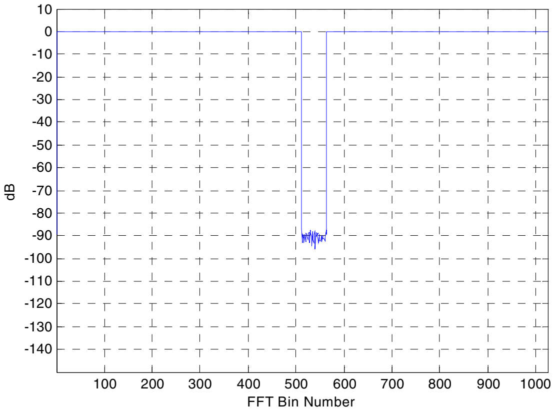


Figure 25: **Full-Precision Unscaled Arithmetic**

Figure Top x-ref 26

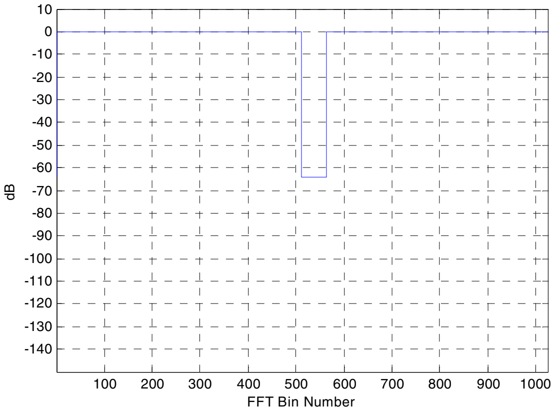


Figure 26: **Scaled (scaling of 1/N) Arithmetic**

Figure Top x-ref 27

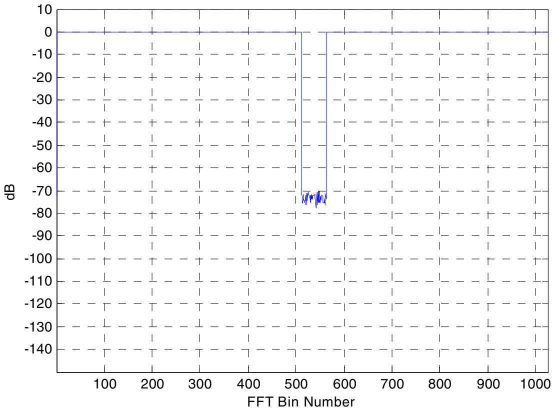


Figure 27: **Block Floating Point Arithmetic**

After the butterfly computation, the LSBs of the data path can be truncated or rounded. The effects of these options are shown below in Figures 28 and 29. Both transforms are 1024 points with 16-bit data and phase factors using block floating point arithmetic.

Figure Top x-ref 28

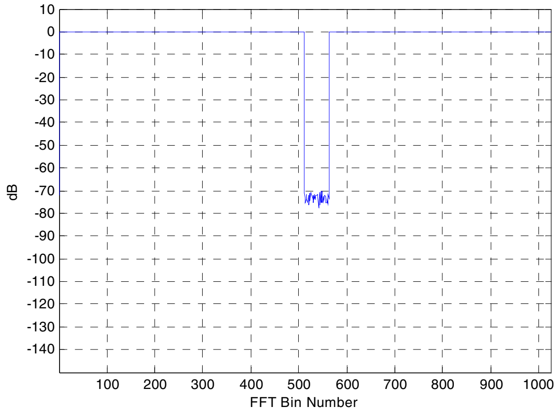


Figure 28: **Convergent Rounding**

Figure Top x-ref 29

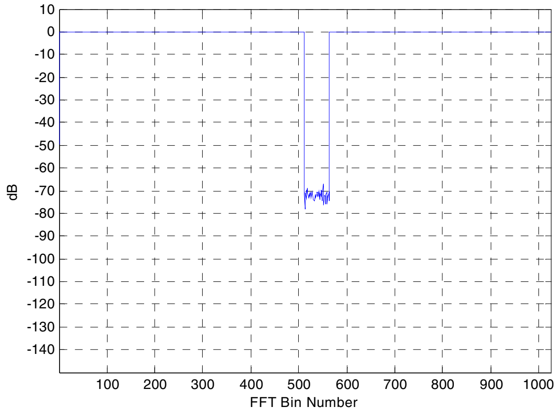


Figure 29: **Truncation**

For illustration purposes, the effect of point size on dynamic range is displayed Figures 30-32. The FFTs in these figures use 16-bit input and phase factors along with convergent rounding and block floating point arithmetic.

Figure Top x-ref 30

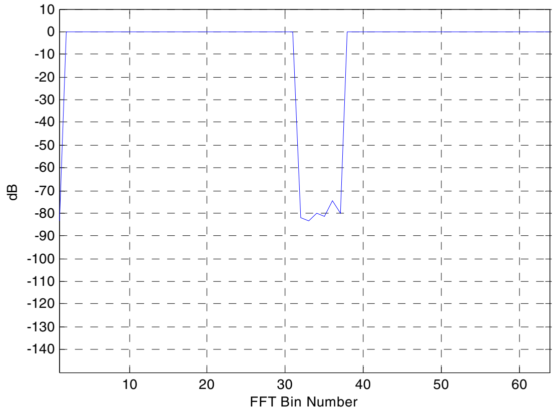


Figure 30: **64-point Transform**

Figure Top x-ref 31

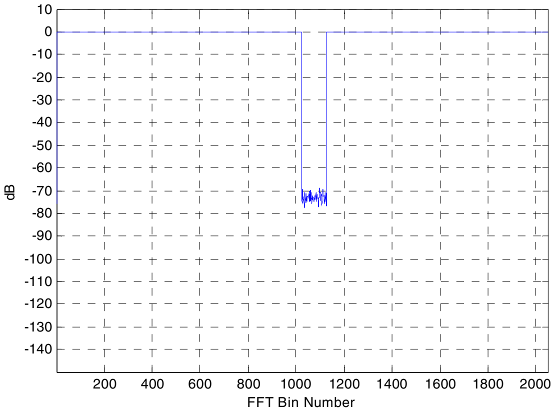


Figure 31: **2048-point Transform**

Figure Top x-ref 32

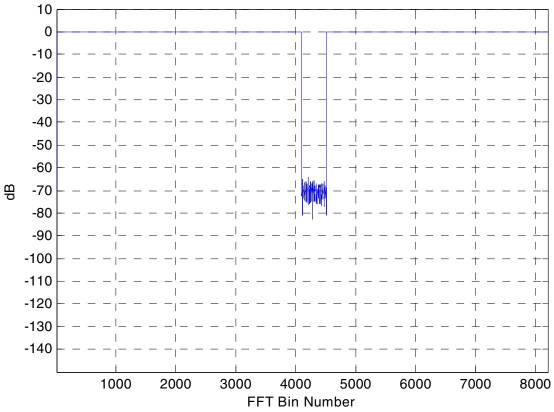


Figure 32: **8192-point Transform**

All of the above dynamic range plots show the result of a Radix-4 architecture. Figures 33-34 show two plots for the Radix-2 architecture. Both use 16-bit input and phase factors along with convergent rounding and block floating point.

Figure Top x-ref 33

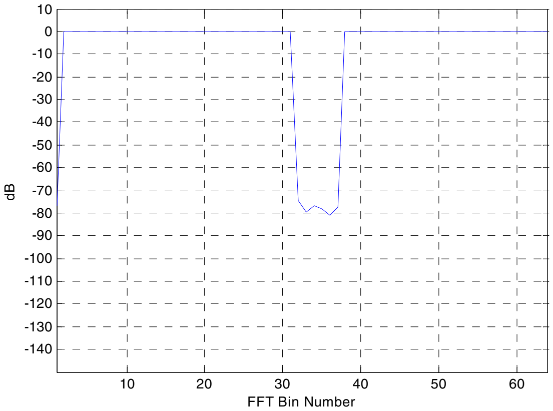


Figure 33: **64-point Radix-2 Transform**

Figure Top x-ref 34

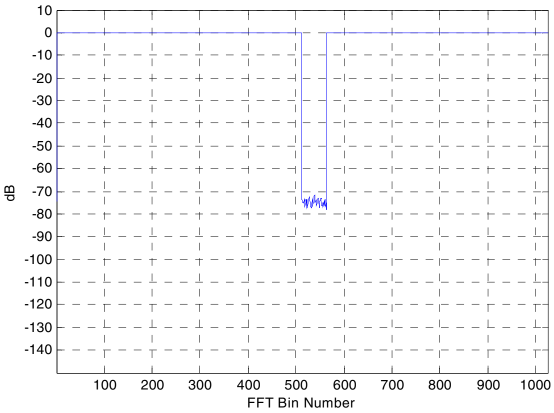


Figure 34: **1024-point Radix-2 Transform**

References

1. J. W. Cooley and J. W. Tukey, *An Algorithm for the Machine Computation of Complex Fourier Series, Mathematics of Computation*, Vol. 19, pp. 297-301, April 1965.
2. J. G. Proakis and D. G. Manolakis, *Digital Signal Processing Principles, Algorithms and Applications Second Edition*, Maxwell Macmillan International, New York, 1992.
3. W. R. Knight and R. Kaiser, *A Simple Fixed-Point Error Bound for the Fast Fourier Transform, IEEE Trans. Acoustics, Speech and Signal Proc.*, Vol. 27, No. 6, pp. 615-620, December 1979.
4. L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1975.

Ordering Information

The XFFT core may be downloaded from the Xilinx IP Center for use with the Xilinx CORE Generator v6.3i and later. The Xilinx CORE Generator system is bundled with all Alliance Series Software packages at no additional charge. Information about additional Xilinx LogiCORE modules is available on the Xilinx IP Center.

To order Xilinx software, please visit the Xilinx Silicon Xpresso Cafe or contact your local Xilinx sales representative.

Revision History

Date	Version	Revision
03/28/03	1.0	Xilinx release in new template.
07/14/03	2.0	Modified Figures 8 through 14, inclusive.
12/11/03	2.1	Updated to v2.1 release.
05/21/04	3.0	Updated to v3.0 release.
11/11/04	3.1	Updated document to support core v3.1 release - updated performance and resource utilization tables for Virtex-II and Virtex-II Pro. Also added performance and resource utilization tables for Virtex-4.