Unified FFT Intel FPGA IPs User Guide

Updated for Intel® Quartus® Prime Design Suite: **23.3**

IP Version: **1.0.7**

Online Version
Send Feedback

UG-20304

683366
2023.09.30

Contents

1. About the Unified FFT Intel FPGA IPs

The Unified FFT IPs comprise the Bit-reverse Intel FPGA IP, the FFT Intel FPGA IP, the Parallel FFT Intel FPGA IP, the Variable Size Bit-reverse Intel FPGA IP, and the Variable Size FFT Intel FPGA IP.

These IP use the same high-level synthesis technology as DSP Builder for Intel FPGAs. The high-level synthesis technology allows you to generate an IP that specifically targets the selected device family, speed grade, and frequency.

Intel recommends you use these Unified FFT IPs and not the FFT IP Core unless:

- The global enable-based flow control in Unified FFT IPs is not suitable for your requirements.
- You require 128K or 256K FFTs.
- You require bidirectional FFTs.
- You require an in-place or memory-based architecture for low-rate applications.

Related Information

FFT IP Core User Guide

1.1. Features of the Unified FFT Intel FPGA IPs

- Fixed, variable, and parallel streaming architectures.
- Fixed-point and multiple precision floating-point support.
- Optimized for selected frequency, device family, and speed grade using DSP Builder for Intel FPGAs technology.
- Software models in C++ language.
- VHDL testbenches.
- Optional global enable.

1.2. Unified FFT Intel FPGA IPs Device Family Support

The IP supports the following devices families:

- Intel Agilex® 7 devices
- Intel® Arria® 10 devices
- Intel Cyclone® 10 devices
- Intel Stratix® 10 devices

For the device support levels for Intel FPGA IP, refer to *Device Support and Pin-Out Status* in the latest version of the *Intel Quartus Prime Pro Edition: Software and Device Support Release Notes*.

Related Information

Device Support and Pin-Out Status

1.3. Release Information for the Unified FFT IPs

Intel FPGA IP versions match the Intel Quartus® Prime Design Suite software versions until v19.1. Starting in Intel Quartus Prime Design Suite software version 19.2, Intel FPGA IP has a new versioning scheme.

The Intel FPGA IP version (X.Y.Z) number can change with each Intel Quartus Prime software version. A change in:

- X indicates a major revision of the IP. If you update the Intel Quartus Prime software, you must regenerate the IP.
- Y indicates the IP includes new features. Regenerate your IP to include these new features.
- Z indicates the IP includes minor changes. Regenerate your IP to include these changes.

Table 1. Unified FFTs IP Release Information

Item	Description
Version	1.0.7
Release Date	September 2023

1.4. Supported Data Types

The Unified FFT IPs support various fixed- and floating-point data types.

Supported Fixed-Point Data Types

The fixed-point input and output data are in signed two’s complement representation. You determine the position of the binary point of the output by the selected pruning strategy and the position of the binary point of the input. For example, a 16 bit input grows to 25 bits in a 256 point FFT if you apply no pruning. If 8 bits of the input are fractional bits, 8-bits of the output are fractional bits. If you apply mild pruning the IP removes the three least significant bits (LSBs) and the output is 22 bits where 5 of the LSBs are fractional bits.

Supported Floating Point Data Types

You can independently set the mantissa and exponent widths within an allowed range to specify hundreds of floating-point formats to suit your requirements. The supported floating-point types are either IEEE 754 formats (half, single and double precision) or custom IEEE 754-like formats with user-specified exponent and fraction-field widths.

The Unifed FFT IPs represent the special values positive zero, negative zero, and non-numbers in the standard IEEE 754 manner, namely:

- Zero is mantissa=0 and exponent=0 with the sign-bit giving the sign.
- Infinity is mantissa=0 and exponent=all ones with the sign-bit giving the sign.
- Not a number (NaN) is mantissa != 0 and exponent=all ones.

Subnormal values are flushed to zero.

Except for the preceding special values, the numerical value of a float type is given in terms of its bit-wise representation by:

$$(-1)^{sign} \times 2^{exponent - bias} \times \left(1 + \left(mantissa / 2^{mantissa\_width}\right)\right)$$

where:

- Exponent, bias and mantissa are the base-10 equivalents of the respective bit sequences
- You specify the widths of exponent and mantissa, the width of sign bit is 1, and the value of the bias is given by:

$$bias = 2^{exponent\_width - 1} - 1$$

For example, for a 32-bit single precision floating point number with a bit-wise representation of 0x40300000:

- sign = 0b = 0
- exponent = 10000000b = 128
- mantissa = 01100000000000000000000b = 3145728

Then:

f = (-1)^0 × 2^(128-127) × (1+(3145728/(2^23))) = 1 × 2 × (1+0.375) = 2.75

1.5. FFT Spectrum

The Unified FFT IPs produce a double-sided frequency spectrum.

The frequency bin 0 (first) contains the DC signal. In natural order, frequency bins 1 to FFT_size/2 contain the positive frequencies in increasing order up to the Nyquist frequency. In natural order, the frequency bin (FFT_size/2) + 1 contains the most negative frequencies equal in absolute value to positive frequencies in bin (FFT_size/2) -1. The negative frequencies decrease in absolute value with increasing bin numbers. The frequency bin FFT_size -1 contains the least negative frequencies equal in absolute value to positive frequencies in bin 1.

For inverse FFTs the input spectrum is expected in this format.

1.6. Global Enable

You may specify an optional global enable signal for all Unified FFT IPs.

When a global enable signal is present, you enable the IPs when the signal is high and disable them when the signal is low. When you disable an IP, the externally visible state of the IP freezes in its last state when it was enabled. The IP ignores all inputs when it is disabled. When the IP is re-enabled it continues as if nothing happened while it was disabled.

2. Getting Started with the Unified FFT Intel FPGA IP

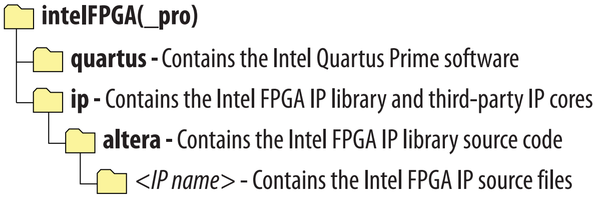
Describes installing, licensing, and compiling the IPs and running the testbench.

2.1. Installing and Licensing Intel FPGA IP Cores

The Intel Quartus Prime software installation includes the Intel FPGA IP library. This library provides many useful IP cores for your production use without the need for an additional license. Some Intel FPGA IP cores require purchase of a separate license for production use. The Intel FPGA IP Evaluation Mode allows you to evaluate these licensed Intel FPGA IP cores in simulation and hardware, before deciding to purchase a full production IP core license. You only need to purchase a full production license for licensed Intel IP cores after you complete hardware testing and are ready to use the IP in production.

The Intel Quartus Prime software installs IP cores in the following locations by default:

Figure 1. IP Core Installation Path



intelFPGA(_pro)

quartus - Contains the Intel Quartus Prime software

ip - Contains the Intel FPGA IP library and third-party IP cores

altera - Contains the Intel FPGA IP library source code

<IP name> - Contains the Intel FPGA IP source files

Table 2. IP Core Installation Locations

Location	Software	Platform
`<drive>:\intelFPGA_pro\quartus\ip\altera`	Intel Quartus Prime Pro Edition	Windows*
`<drive>:\intelFPGA\quartus\ip\altera`	Intel Quartus Prime Standard Edition	Windows
`<home directory>:/intelFPGA_pro/quartus/ip/altera`	Intel Quartus Prime Pro Edition	Linux*
`<home directory>:/intelFPGA/quartus/ip/altera`	Intel Quartus Prime Standard Edition	Linux

2.2. Generating a Unified FFT IP

To include the IP in a design, generate the IP in the Intel Quartus Prime software. Or optionally, you can generate a design example.

1. Create a New Intel Quartus Prime project.
2. Open IP Catalog.
3. Select **DSP ➤ Transforms ➤ Unified FFT** and click **Add**

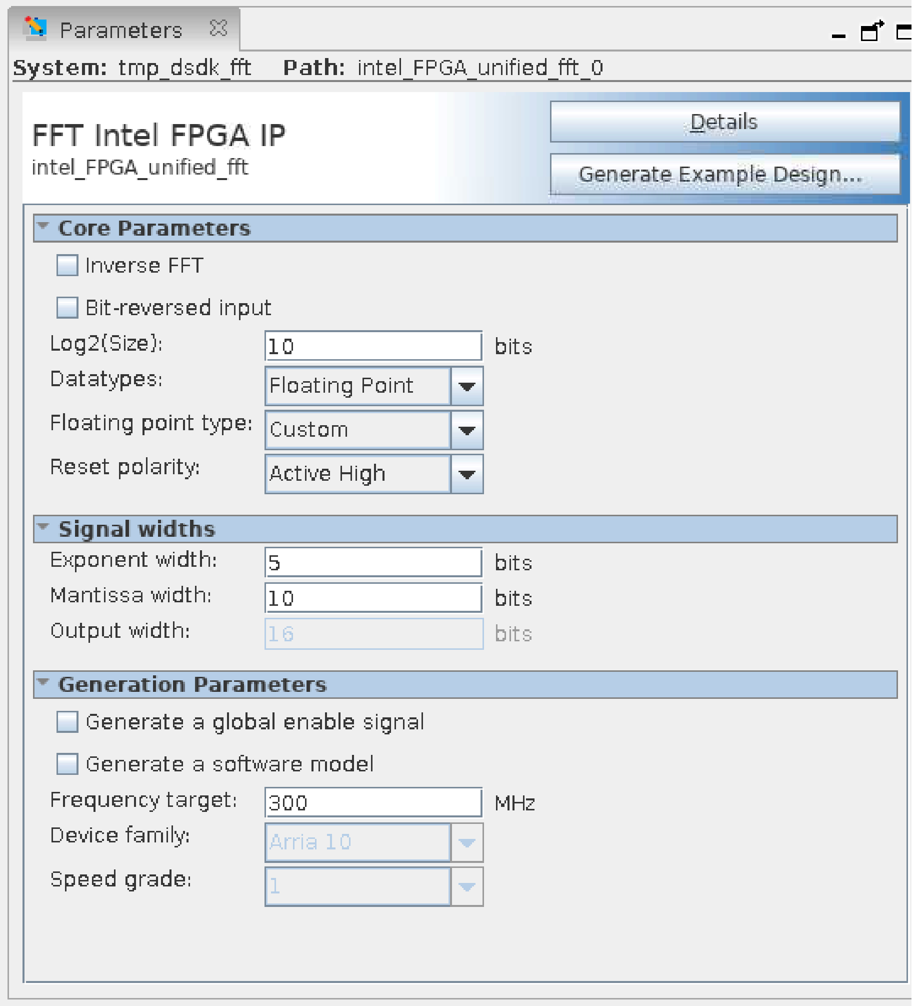
4. Enter a name for your IP variant and click **Create**.

   The name is for both the top-level RTL module and the corresponding `.ip` file.

   The parameter editor for this IP appears.

5. Choose your parameters.

Figure 2. Parameter Editor



Parameters

System: tmp_dsdk_fft Path: intel_FPGA_unified_fft_0

FFT Intel FPGA IP
intel_FPGA_unified_fft

Core Parameters

□ Inverse FFT

□ Bit-reversed input

Log2(Size): 10 bits

Datatypes: Floating Point

Floating point type: Custom

Reset polarity: Active High

Signal widths

Exponent width: 5 bits

Mantissa width: 10 bits

Output width: 16 bits

Generation Parameters

□ Generate a global enable signal

□ Generate a software model

Frequency target: 300 MHz

Device family: Arria 10

Speed grade: 1

6. Click **Generate HDL**.

Related Information

Generating IP Cores
   Use this link for the Quartus Prime Pro Edition Software.

2.3. Compiling the Software Model for the Unified FFT IPs

When you turn on **Generate a software model**, Intel Quartus Prime generates a software model of the IP in C++ language.

The software models are in the following directories depending on the IP:

- `intel_FPGA_unified_fft_10/sim/cmodel`
- `intel_FPGA_unified_vfft_10/sim/cmodel`
- `intel_FPGA_unified_pfft_10/sim/cmodel`
- `intel_FPGA_unified_bitrev_10/sim/cmodel`
- `intel_FPGA_unified_vbitrev_10/sim/cmodel`

You need a gcc version from 7.2 to 9.2 on Linux and Microsoft Visual Studio 2017 or 2019 on Windows to compile the C++ model. Later versions and other compilers may require minor modifications to the static or generated C++ source code to work correctly.

The `cmodel` directory contains the following files:

- `csl.h/cpp` files containing utility functions and implementation details for the generated models.
- *`instance_name`*`.h` or `.cpp`) files containing the simulation model of the IP and *`instance_name`*`_atb.cpp` containing the testbench of the IP.
- `atb_app.cpp` file containing the `main()` function.
- `CMakeFiles.txt/CMakeLists.txt` file containing `CMake` build scripts for building the ATB executable and model files.

1. Generate the project or makefiles by running the CMake build scripts using CMake 3.13 or greater. For example, to generate Visual Studio 2017 projects, from the `cmodel/build` subdirectory, run:

   ```
   cmake -G "Visual Studio 15 2017 Win64" -DWRITE_STM_FILES=1
   ```

   Or to generate a makefile for the release build with symbols on Linux, run:

   ```
   cmake -G "Unix Makefiles" -DCMAKE_BUILD_TYPE=RelWithDebInfo -
   DWRITE_STM_FILES=1
   ```

   Defining `WRITE_STM_FILES=1` is necessary for the test executable (`atb_app`).

   You may need to set the CC and CXX variables to the location of gcc and g++ executables on Linux for proper operation of CMake. For example, `export CC=/path/to/gcc/7.2.0/1/linux64/bin/gcc export CXX=/path/to/gcc/7.2.0/1/linux64/bin/g++`.
   Refer to the CMake documentation for more options.

2. Set the `MPIR_INC_PATH`, `MPIR_LIB_PATH`, `MPFR_INC_PATH`, `MPFR_LIB_PATH` options to the include and library directories of builds of the mpfr or mpir libraries if the build scripts require them.

   You require the libraries if the internal bit widths in the software model are larger than 64-bits or when modeling certain floating-point configurations. The internal bit widths may be different to the bit widths that you choose. Build instructions and prebuilt binaries are on the mpfr or mpir websites.

3. On Windows, open the generated solution file and run the compilation. On Linux, run `make`.

Alternatively, run CMake again with the `-build` option to compile on both Windows and Linux, e.g:`cmake –build . –config Release`

4. Run the `atb_app` executable in the `cmodel` directory as the working directory so that the generated stimulus file paths are correct. If simulation is successful, the executable produces the following output to `stdout`:

```
Opening stimulus files...
Simulating...
Simulation has completed.
```

5. Refer to the testbench to see how you can integrate the generated models into an existing system.

   The basic application programming interface (API) of the C++ model is:

```
 // Create an instance of the C model class
unified_fft_c_model_t inst0;
 // Reset the model
 inst0.reset();
 // Note: this is a C Model of the RTL pipeline, so needs to be flushed to
    get all the output just like RTL
 while(there_is_input_or_expecting_output) {
   // Write input
   inst0.write(io_chanIn_cunroll_x_t0);
   // Execute for all chanIn/Out structures
   inst0.execute(io_chanIn_cunroll_x_t0);
   // Execute for all chanIn/Out structures
   inst0.execute(io_chanOut_cunroll_x_t0);
   // Read output
   inst0.read(io_chanOut_cunroll_x_t0);
 }
```

2.4. Running the Unified FFT IPs Automatic Testbench

Intel provides an automatic testbench as part of the simulation fileset.

1. Navigate to the appropriate simulator directory in the `sim` folder. For example, `sim/mentor` for ModelSim

2. Run the appropriate setup script within the simulator, for example, `msim_setup.tcl` for ModelSim.

3. Follow the instructions and set the TOP_LEVEL_NAME to the name of the testbench.

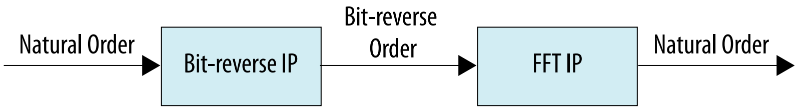
   The name of the testbench is in the information window when the simulation fileset is generated. For example, you may see:

```
Info: intel_FPGA_unified_fft_0: To run testbench, generate the Qsys
testbench system and set TOP_LEVEL_NAME to
intel_FPGA_unified_fft_10.my_fft_intel_FPGA_unified_fft_10_4xf52ey_atb to
simulate the IP core
```

3. Bit-reverse Intel FPGA IP

This IP performs radix-2 FFT bit reverse operation. You may use this IP to bit-reverse the input or the output of FFT Intel FPGA IP.

Figure 3. Example use of the Bit-reverse Intel FPGA IP



If the input data width is smaller than the output data width, bit-reversing the input reduces memory usage in the Bit-reverse Intel FPGA IP. However, when the input is bit-reversed, the memory usage of the FFT Intel FPGA IP increases. You should experimentally determine the best configuration for memory usage.

Table 3. Bit-reverse Intel FPGA IP Input Signals

Name	Required	Description
`clk`	Yes	All input signals must be synchronous to this clock.
`rst`	Yes	Reset signal. The reset signal is asynchronous for Intel Arria 10 and Intel Cyclone 10 GX device; synchronous for Intel Agilex 7 and Intel Stratix 10. If the reset is asynchronous, deassert the reset signal synchronously to the input clock to avoid metastability issues. Select the reset polarity with the **Reset polarity** parameter.
`validIn`	Yes	Data valid signal. Assert this signal when input data is valid. This signal must not deassert during an FFT. Keep it asserted from the first input to last input of an FFT.
`channelIn`	Yes	Not used. Connect it to ground.
`d`	Yes	Data input signal. This signal contains the optional enable and complex or real input. The signal order is from least significant bit (LSB) to most significant bit (MSB): 1. Real data input 2. Imaginary data input (if complex) 3. Global enable input (if requested)

Table 4. Bit-reverse Intel FPGA IP Output Signals

Name	Required	Description
`validOut`	Yes	Data valid signal. The IP asserts this signal for valid output data.
`channelOut`	Yes	Not used.
`q`	Yes	Data output signal.

continued...

Name	Required	Description
		The signals are in the following order from least significant bit (LSB) to most significant bit (MSB): 1. Real output 2. Imaginary output (if complex)

Table 5. Bit-reverse Intel FPGA IP Parameters

Parameter	Value	Description
Core Parameters		
Complex input	-	Select if input should be treated as a complex value.
Log2(Size)	2 to 16	Specify the size of the FFT. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT.
Reset polarity	Active High Active Low	Select the reset polarity.
Signal Widths		
Input width	4 to 64	Specify the width of the input in bits. If the input is complex, the IP applies the input width separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter.
Generation Parameters		
Generate a global enable signal	-	Select to generate a global enable signal, which you can use to enable and disable the IP. This enable doesn't force the output valid signal to go low.
Generate a software model	-	Select to generate a software model of the IP in C++ language.
Generate a cycle-accurate software model	-	Select to generate a cycle-accurate software model. If this option is not selected the generated software model is not cycle accurate with respect to RTL of the IP. Only available when you select **Generate a software model** .
Generate HLD external function wrappers	-	Select to generate wrapper files so that you can use the IP as an external function with HLD tools such as HLS and OpenCL. Only available if generating a software model.
Frequency target	20 to 2000	Specify the frequency at which the IP is required to operate in MHz. The **Frequency target** affects the RTL generation for that IP.
Device family	-	Specifies the device family the IP is targeting. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Device family** affects the RTL generation for that IP.
Speed grade	-	Specifies the speed grade of the device the IP is required to operate on. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Speed grade** affects the RTL generation for that IP.

4. FFT Intel FPGA IP

The IP performs fixed-size streaming FFT and inverse FFT operations.

Table 6. FFT Intel FPGA IP Input Signals

Name	Description
clk	All input signals must be synchronous to this clock.
rst	Reset signals. The reset signal is asynchronous for Intel Arria 10 and Intel Cyclone 10 GX device; synchronous for Intel Agilex 7 and Intel Stratix 10. If the reset is asynchronous, deassert the reset signal synchronously to the input clock to avoid metastability issues. Select the reset polarity with the **Reset polarity** parameter.
validIn	Data valid signal. Assert this signal when input data is valid. This signal must not deassert during an FFT. Keep it asserted from the first input to last input of an FFT.
channelIn	Not used. Connect it to ground.
d	Data input signal. This signal contains the optional enable and complex input. The signal order is from least significant bit (LSB) to most significant bit (MSB): 1. Real data input 2. Imaginary data input 3. Global enable input (if requested)

Table 7. FFT Intel FPGA IP Output Signals

Name	Description
validOut	Data valid signal. The IP asserts this signal for valid output data.
channelOut	Not used.
q	Data output signal. This signal contains the size and complex outputs. The signals are in the following order from LSB to MSB: 1. Real output 2. Imaginary output

Table 8. FFT Intel FPGA IP Parameters

Parameter	Value	Description
Core Parameters		
Inverse FFT	-	Specify if the operation is an FFT or inverse FFT. Turn on for an inverse FFT; turn off to compute a normal FFT.
Bit-reversed input		Specify if the input is in natural or bit-reversed order. Turn on for the IP to receive its input in bit-reversed order and to produce its output in natural order. Turn off for the IP to receive its input in natural order and to produce its output in bit-reversed order.

continued...

Parameter	Value	Description
		You can use Bit-reverse Intel FPGA IP to bit-reverse the input or the output.
Log2(Size)	2 to 16	Specify the size of the FFT. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT.
Datatypes	Fixed Point Floating Point	Select a fixed-point or floating-point FFT.
Floating point types	Single Double Custom	Select a floating-point format. Only available when **Datatypes** is **Floating Point**.
Reset polarity	Active High Active Low	Select the reset polarity.
Signal Widths		
Input width	4 to 32	Specify the width of the input in bits. The input is a complex value. The IP applies the **Input width** separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter. Only available when **Datatypes** is **Fixed Point**.
Twiddle width	4 to 32	Specify the width of the twiddle constants. The twiddle constants are complex values. The IP applies the **Twiddle width** separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter.
Exponent width	5 to 11	Specify the width of the floating point exponent. Only available when **Floating point type** is **Custom**.
Mantissa width	7 to 52	Specify the width of the floating-point mantissa. Only available when **Floating point type** is **Custom**.
Twiddle Parameters		
Pruning scheme	Full Wordgrowth Mild Pruning Prune To Width	Select the pruning scheme. Each pruning scheme has a different effect on the datapath: • **Full wordgrowth** allows the datapath to grow by one bit at each adder, and at the first multiplier. • **Mild pruning** allows the datapath to grow by one bit at each adder, and at the first multiplier. It also reduces its width by one bit immediately before each multiplier. • **Prune to width** allows the datapath to grow by one bit at each adder, and at the first multiplier. It also reduces its width to be no more than the specified **Pruning width** immediately before each multiplier. Whenever pruning is applied to the datapath only the least significant bit(s) of the datapath are removed. Only available when **Datatypes** is **Fixed Point**.
Pruning width	4 to 32	Specify the pruning width. The IP applies the **Pruning width** separately to the real and imaginary components of the complex signal. Only available when **Pruning scheme** is set to **Prune To Width**.
Generation Parameters		
Generate a global enable signal	-	Select to generate a global enable signal, which you can use to enable and disable the IP. This enable doesn't force the output valid signal to go low.
Generate a software model	-	Select to generate a software model of the IP in C++ language.

continued...

Parameter	Value	Description
Generate a cycle-accurate software model	-	Select to generate a cycle-accurate software model. If this option is not selected the generated software model is not cycle accurate with respect to RTL of the IP. Only available when you select **Generate a software model** .
Generate HLD external function wrappers	-	Select to generate wrapper files so that you can use the IP as an external function with HLD tools such as HLS and OpenCL. Only available if generating a software model.
Frequency target	20 to 2000	Specify the frequency at which the IP is required to operate in MHz. The **Frequency target** affects the RTL generation for that IP.
Device family	-	Specifies the device family the IP is targeting. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Device family** affects the RTL generation for that IP.
Speed grade	-	Specifies the speed grade of the device the IP is required to operate on. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Speed grade** affects the RTL generation for that IP.

5. Parallel FFT Intel FPGA IP

The IP performs parallel streaming FFT and inverse FFT operations. The IP accepts a user specified number of FFT inputs on parallel wires. This IP uses a hybrid architecture using parallel and serial FFT stages.

Table 9. Parallel FFT Intel FPGA IP Input Signals

Name	Description
`clk`	All input signals must be synchronous to this clock.
`rst`	Reset signals. The reset signal is asynchronous for Intel Arria 10 and Intel Cyclone 10 GX device; synchronous for Intel Agilex 7 and Intel Stratix 10. If the reset is asynchronous, deassert the reset signal synchronously to the input clock to avoid metastability issues. Select the reset polarity with the **Reset polarity** parameter.
`validIn`	Data valid signal. Assert this signal when input data is valid. This signal must not deassert during an FFT. Keep it asserted from the first input to last input of an FFT.
`channelIn`	Not used. Connect it to ground.
`d`	Data input signal. This signal contains the optional enable and complex inputs. The signal order is from least significant bit (LSB) to most significant bit (MSB): 1. First real data input 2. First imaginary data input 3. ... 4. Last real data input 5. Last imaginary data input 6. Global enable input (if requested)

Table 10. Parallel FFT Intel FPGA IP Output Signals

Name	Description
`validOut`	Data valid signal. The IP asserts this signal for valid output data.
`channelOut`	Not used.
`q`	Data output signal. This signal contains the size and complex or real outputs. The signals are in the following order from LSB to MSB: 1. First real output 2. First imaginary output 3. ... 4. Last real output 5. Last imaginary output

Table 11. Parallel FFT Intel FPGA IP Parameters

Parameter	Value	Description
Core Parameters		
Inverse FFT	-	Specify if the operation is an FFT or inverse FFT. Turn on for an inverse FFT; turn off to compute a normal FFT.
Bit-reversed input		Specify if the input is in natural or bit-reversed order. Turn on for the IP to receive its input in bit-reversed order and to produce its output in natural order. Turn off for the IP to receive its input in natural order and to produce its output in bit-reversed order.
Log2(Size)	2 to 16	Specify the size of the FFT. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT.
Log2(Wires)	1 to 5	Specify the number of parallel inputs. For example, a value of 1 means 2 parallel data inputs and a value of 5 means 32 parallel data inputs.
Number of serial stages	0 to (Log2(Size)-Log2(Wires))	Specify the number of serial stages of the hybrid architecture. The hybrid architecture consists of zero-or-more serial stages combined with one-or-more parallel stages. This number controls the architecture that implements the FFT and affects its resource usage and latency.
Datatypes	Fixed Point Floating Point	Select a fixed-point or floating-point FFT.
Floating point types	Single Double Custom	Select a floating-point format. Only available when **Datatypes** is **Floating Point**.
Reset polarity	Active High Active Low	Select the reset polarity.
Signal Widths		
Input width	4 to 32	Specify the width of the input in bits. The input is a complex value. The IP applies the **Input width** separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter. Only available when **Datatypes** is **Fixed Point**.
Twiddle width	4 to 32	Specify the width of the twiddle constants. The twiddle constants are complex values. The IP applies the **Twiddle width** separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter.
Exponent width	5 to 11	Specify the width of the floating point exponent. Only available when **Floating point type** is **Custom**.
Mantissa width	7 to 52	Specify the width of the floating-point mantissa. Only available when **Floating point type** is **Custom**.
Twiddle Parameters		
Pruning scheme	Full Wordgrowth Mild Pruning Prune To Width	Select the pruning scheme.

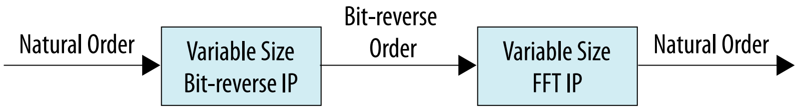
continued...

Parameter	Value	Description
		Each pruning scheme has a different effect on the datapath: • **Full wordgrowth** allows the datapath to grow by one bit at each adder, and at the first multiplier. • **Mild pruning** allows the datapath to grow by one bit at each adder, and at the first multiplier. It also reduces its width by one bit immediately before each multiplier. • **Prune to width** allows the datapath to grow by one bit at each adder, and at the first multiplier. It also reduces its width to be no more than the specified **Pruning width** immediately before each multiplier. Whenever pruning is applied to the datapath only the least significant bit(s) of the datapath are removed. Only available when **Datatypes** is **Fixed Point**.
Pruning width	4 to 32	Specify the pruning width. The IP applies the **Pruning width** separately to the real and imaginary components of the complex signal. Only available when **Pruning scheme** is set to **Prune To Width**.
Generation Parameters		
Generate a global enable signal	-	Select to generate a global enable signal, which you can use to enable and disable the IP. This enable doesn't force the output valid signal to go low.
Generate a software model	-	Select to generate a software model of the IP in C++ language.
Generate a cycle-accurate software model	-	Select to generate a cycle-accurate software model. If this option is not selected the generated software model is not cycle accurate with respect to RTL of the IP. Only available when you select **Generate a software model** .
Generate HLD external function wrappers	-	Select to generate wrapper files so that you can use the IP as an external function with HLD tools such as HLS and OpenCL. Only available if generating a software model.
Frequency target	20 to 2000	Specify the frequency at which the IP is required to operate in MHz. The **Frequency target** affects the RTL generation for that IP.
Device family	-	Specifies the device family the IP is targeting. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Device family** affects the RTL generation for that IP.
Speed grade	-	Specifies the speed grade of the device the IP is required to operate on. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Speed grade** affects the RTL generation for that IP.

6. Variable Size Bit-reverse Intel FPGA IP

This IP performs radix-2 FFT bit reverse operation. The size of the input stream may be variable. You may use this IP to bit-reverse the input or the output of FFT Intel FPGA IP.

Figure 4. Example use of the Variable Size Bit-reverse Intel FPGA IP



If the input data width is smaller than the output data width, bit-reversing the input reduces memory usage in the Bit-reverse Intel FPGA IP. However, when the input is bit-reversed, the memory usage of the FFT Intel FPGA IP increases. You should experimentally determine the best configuration for memory usage.

Table 12. Variable Size Bit-reverse Intel FPGA IP Input Signals

Name	Description
`clk`	All input signals must be synchronous to this clock.
`rst`	Reset signals. The reset signal is asynchronous for Intel Arria 10 and Intel Cyclone 10 GX device; synchronous for Intel Agilex 7 and Intel Stratix 10. If the reset is asynchronous, deassert the reset signal synchronously to the input clock to avoid metastability issues. Select the reset polarity with the **Reset polarity** parameter.
`validIn`	Data valid signal. Assert this signal when input data is valid. This signal must not deassert during an FFT. Keep it asserted from the first input to last input of an FFT.
`channelIn`	Not used. Connect it to ground.
`d`	Data input signal. This signal contains the real and complex data, size of the FFT, and the optional enable. The size of the FFT is represented as a logarithm to base 2. For example, a value of 2 represents a 4-point FFT and 16 represents a 64K-point FFT The signal order is from least significant bit (LSB) to most significant bit (MSB): 1. Real data input 2. Imaginary data input (if complex) 3. Size of the current FFT 4. Global enable input (if requested) Do not set the `size` input (which dynamically controls the current FFT size) to a value larger than the value specified by **Log2(Maximum Size)**.

Table 13. Variable Size Bit-reverse Intel FPGA IP Output Signals

Name	Description
`validOut`	Data valid signal. The IP asserts this signal for valid output data.
`channelOut`	Not used.
`q`	Data output signal. This signal contains real and complex data and size of the FFT. The size of the FFT is represented as a logarithm to base 2. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT. The signals are in the following order from LSB to MSB: 1. Real output 2. Imaginary output (if complex) 3. Size of the current FFT

Table 14. Variable Size Bit-reverse Intel FPGA IP Parameters

Parameter	Value	Description
Core Parameters		
Complex input	-	Select if input should be treated as a complex value.
Log2(Maximum Size)	2 to 16	Specify the maximum size of the FFT. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT.
Reset polarity	Active High Active Low	Select the reset polarity.
Signal Widths		
Input width	4 to 64	Specify the width of the real or complex data input in bits. If the input is complex, the IP applies the input width separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter.
Generation Parameters		
Generate a global enable signal	-	Select to generate a global enable signal, which you can use to enable and disable the IP. This enable doesn't force the output valid signal to go low.
Generate a software model	-	Select to generate a software model of the IP in C++ language.
Generate a cycle-accurate software model	-	Select to generate a cycle-accurate software model. If this option is not selected the generated software model is not cycle accurate with respect to RTL of the IP. Only available when you select **Generate a software model** .
Generate HLD external function wrappers	-	Select to generate wrapper files so that you can use the IP as an external function with HLD tools such as HLS and OpenCL. Only available if generating a software model.
Frequency target	20 to 2000	Specify the frequency at which the IP is required to operate in MHz. The **Frequency target** affects the RTL generation for that IP.
Device family	-	Specifies the device family the IP is targeting. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Device family** affects the RTL generation for that IP.
Speed grade	-	Specifies the speed grade of the device the IP is required to operate on. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Speed grade** affects the RTL generation for that IP.

7. Variable Size FFT Intel FPGA IP

The IP performs variable size streaming FFT and inverse FFT operations. You can dynamically specify the size of the current FFT. You must completely flush the previous FFT out of the FFT pipeline before changing the size of the FFT.

Table 15. Variable Size FFT Intel FPGA IP Input Signals

Name	Description
`clk`	All input signals must be synchronous to this clock.
`rst`	Reset signals. The reset signal is asynchronous for Intel Arria 10 and Intel Cyclone 10 GX device; synchronous for Intel Agilex 7 and Intel Stratix 10. If the reset is asynchronous, deassert the reset signal synchronously to the input clock to avoid metastability issues. Select the reset polarity with the **Reset polarity** parameter.
`validIn`	Data valid signal. Assert this signal when input data is valid. This signal must not deassert during an FFT. Keep it asserted from the first input to last input of an FFT.
`channelIn`	Not used. Connect it to ground.
`d`	Data input signal. This signal contains the complex data, size of the FFT, and the optional enable. The size of the FFT is represented as a logarithm to base 2. For example, a value of 2 represents a 4-point FFT and 16 represents a 64K-point FFT. The signal order is from least significant bit (LSB) to most significant bit (MSB): 1. Real data input 2. Imaginary data input 3. Size of the current FFT 4. Global enable input (if requested)

Table 16. Variable Size FFT Intel FPGA IP Output Signals

Name	Description
`validOut`	Data valid signal. The IP asserts this signal for valid output data.
`channelOut`	Not used.
`q`	Data output signal. This signal contains complex data and size of the FFT. The size of the FFT is represented as a logarithm to base 2. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT. The signals are in the following order from LSB to MSB: 1. Real output 2. Imaginary output 3. Size of the current FFT The size output is the specified input size that the IP delays appropriately to synchronize with the output data.

Table 17. Variable Size FFT Intel FPGA IP Parameters

Parameter	Value	Description
Core Parameters		
Inverse FFT	-	Specify if the operation is an FFT or inverse FFT. Turn on for an inverse FFT; turn off to compute a normal FFT.
Bit-reversed input		Specify if the input is in natural or bit-reversed order. Turn on for the IP to receive its input in bit-reversed order and to produce its output in natural order. Turn off for the IP to receive its input in natural order and to produce its output in bit-reversed order. You can use the Variable Size Bit-reverse Intel FPGA IP to bit-reverse the input or the output.
Log2(Minimum Size)	0 to 16	Specify minimum the size of the FFT. For example, a value of 0 represents a 1 point FFT and 16 represents a 64K point FFT.
Log2(Maximum Size)	2 to 16	Specify the maximum size of the FFT. For example, a value of 2 represents a 4 point FFT and 16 represents a 64K point FFT.
Datatypes	Fixed Point Floating Point	Select a fixed-point or floating-point FFT.
Floating point types	Single Double Custom	Select a floating-point format. Only available when **Datatypes** is **Floating Point**.
Reset polarity	Active High Active Low	Select the reset polarity.
Signal Widths		
Input width	4 to 32	Specify the width of the complex data input in bits. The input is a complex value. The IP applies the **Input width** separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter. Only available when **Datatypes** is **Fixed Point**.
Twiddle width	4 to 32	Specify the width of the twiddle constants. The twiddle constants are complex values. The IP applies the **Twiddle width** separately to the real and imaginary components of the complex signal. The total width of the complex signal is double the value of this parameter.
Exponent width	5 to 11	Specify the width of the floating point exponent. Only available when **Floating point type** is **Custom**.
Mantissa width	7 to 52	Specify the width of the floating-point mantissa. Only available when **Floating point type** is **Custom**.
Twiddle Parameters		
Pruning scheme	Full Wordgrowth Mild Pruning Prune To Width	Select the pruning scheme. Each pruning scheme has a different effect on the datapath: • **Full wordgrowth** allows the datapath to grow by one bit at each adder, and at the first multiplier. • **Mild pruning** allows the datapath to grow by one bit at each adder, and at the first multiplier. It also reduces its width by one bit immediately before each multiplier. • **Prune to width** allows the datapath to grow by one bit at each adder, and at the first multiplier. It also reduces its width to be no more than the specified **Pruning width** immediately before each multiplier. Whenever pruning is applied to the datapath only the least significant bit(s) of the datapath are removed.

continued...

Parameter	Value	Description
		The IP determines the resultant scaling of the output by the maximum size of the FFT and not the current dynamic size. All FFT sizes within the range you specify pass through the same (maximum) number of pruning stages. Only available when **Datatypes** is **Fixed Point**.
Pruning width	4 to 32	Specify the pruning width. The IP applies the **Pruning width** separately to the real and imaginary components of the complex signal. Only available when **Pruning scheme** is set to **Prune To Width**.
Generation Parameters		
Generate a global enable signal	-	Select to generate a global enable signal, which you can use to enable and disable the IP. This enable doesn't force the output valid signal to go low.
Generate a software model	-	Select to generate a software model of the IP in C++ language.
Generate a cycle-accurate software model	-	Select to generate a cycle-accurate software model. If this option is not selected the generated software model is not cycle accurate with respect to RTL of the IP. Only available when you select **Generate a software model** .
Generate HLD external function wrappers	-	Select to generate wrapper files so that you can use the IP as an external function with HLD tools such as HLS and OpenCL. Only available if generating a software model.
Frequency target	20 to 2000	Specify the frequency at which the IP is required to operate in MHz. The **Frequency target** affects the RTL generation for that IP.
Device family	-	Specifies the device family the IP is targeting. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Device family** affects the RTL generation for that IP.
Speed grade	-	Specifies the speed grade of the device the IP is required to operate on. Usually, the Intel Quartus project sets this value. If using other tools such as `qsys-generate`, refer to those tools' documentation. The **Speed grade** affects the RTL generation for that IP.

8. Unified FFT Intel FPGA IPs User Guide Archive

For the latest and previous versions of this document, refer to: Unified FFT Intel FPGA IP User Guide. If an IP or software version is not listed, the user guide for the previous IP or software version applies.

9. Document Revision History for the Unified FFT Intel FPGA IPs User Guide

Document Version	IP Version	Intel Quartus Prime Version	Changes
2023.09.30	1.0.7	23.3	• Updated the product family name to "Intel Agilex 7." • Updated *Compiling the Software Model for the Unified FFTs*
2021.04.05	-	21.1	Added: • *Release Information* • *User Guide Archive* • *Supported Datatypes* • *FFT Spectrum* • *Global Enable* • **Generate a cycle-accurate software model** parameter • *Example use of the Variable Size Bit-reverse Intel FPGA IP* figure
2020.10.05	1.0.0	20.3	Initial release.