



FFT Cores for FPGA

Product Information Sheet

RF Engines Ltd (RFEL) provide a range of high specification FPGA-based Pipelined Fast Fourier Transform (FFT) cores. The pipelined or streaming architecture ensures that all cores can process back-to-back blocks of time-domain input data in real-time. FFT parameters such as radix, processing parallelism and bit growth can be tailored by RFEL to suit each application resulting in the most optimal design in terms of silicon, power and performance.

Complementary core options include polyphase and standard windowing, input buffering for single or multiple channel processing, input buffering for overlapped FFT processing, bit-reversal and 100% efficient real input FFT processing.

The designs are sold as custom Intellectual Property (IP) cores to equipment manufacturers and allow a low cost and reduced risk route to a faster design cycle. Additionally a selection of more than sixty HiSpeed and QuadSpeed FFT cores are available in RFEL's low cost Xilinx FFT Library.

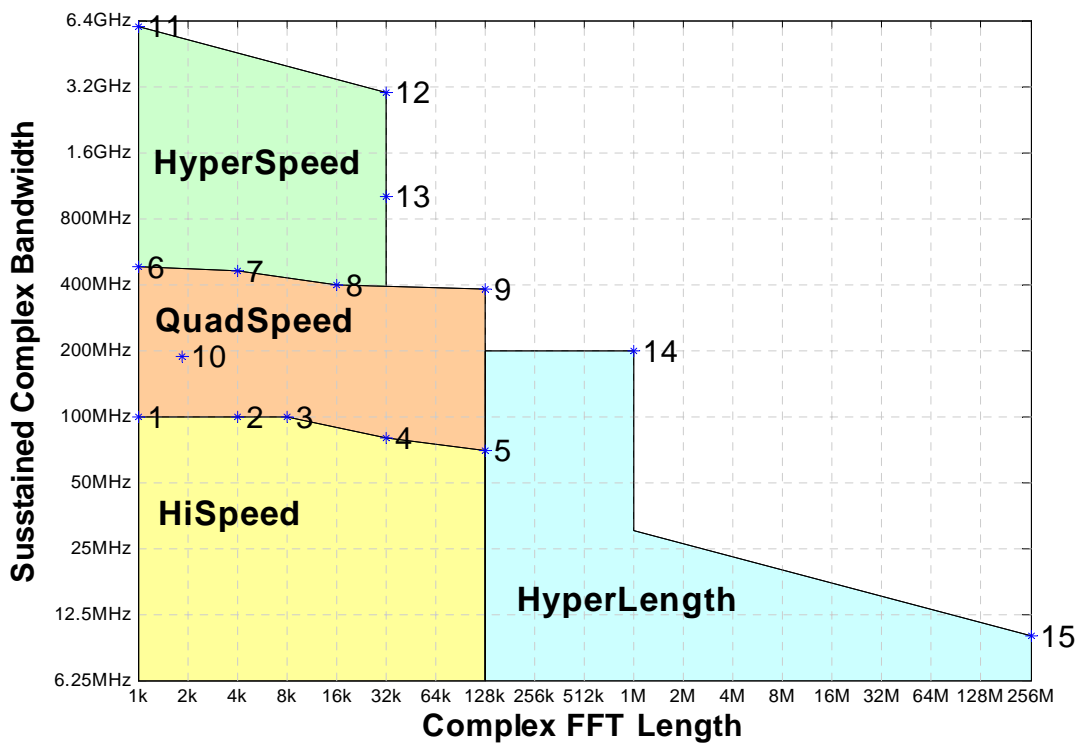


Figure 1: FFT Product Range Performance Envelopes

Figure 1 shows the performance envelopes for RFEL's FFT cores, implemented in a Xilinx Virtex-II PRO device (XC2VP70-7 FF1517). This graph is intended to illustrate the range of core specifications that can be implemented in a single large FPGA device, plus external memory where appropriate. Specific FFT examples (shown as numbered points) are described later.

Overview of the Range

The range includes the following types of FFT core:

- 1) **HiSpeed** FFT cores implement a Pipelined radix-2 Decimate In Frequency (DIF) algorithm using a single data path pipeline. This 'parallelism of 1' can typically accept complex time-domain data at a maximum continuous rate (and hence signal bandwidth) of 100MHz, depending on the FPGA target. The HiSpeed architecture is designed to be the smallest of the range in terms of logic and multiplier resource, and is matched in terms of real-time bandwidth to currently available 12-bit and 14-bit Analogue to Digital Converter (ADC) devices.
- 2) **QuadSpeed** FFT cores implement a Pipelined radix-2 DIF algorithm with a parallelism of 4. These cores can typically accept complex time-domain data at a maximum continuous rate of 400MHz. A QuadSpeed FFT core will consume roughly the same memory resource, two to three times the logic and three to four times the multiplier resource required by a HiSpeed core of the same transform length and arithmetic precision.
- 3) **Matrix** FFT cores provide non-power-of-two length FFTs and are implemented using a mixed-radix algorithm with a parallelism of 2. These cores can typically accept complex time-domain data at a maximum continuous rate of 200MHz. Transforms of any length can be implemented using prime factor FFTs, providing sufficient FPGA resource is available.
- 4) **HyperSpeed** FFT cores implement a mixed-radix algorithm with a parallelism of '2M' and a transform length of 'M x N'. 'M' and 'N' can be any integer, though integer powers of two provide the most silicon-efficient implementations. A HyperSpeed core with a parallelism of 8 can typically accept complex time-domain data at a maximum continuous rate of 800MHz.
- 5) **HyperLength** FFT cores implement a mixed-radix algorithm that requires RAM external to the FPGA to achieve large transform lengths. Both transform length and maximum sustained processing bandwidth are dependent on the type and configuration of external RAM.

Features

- Pipeline architecture allows continuous processing with no gaps in data
- Parameterisable at factory in terms of:
 - Processing parallelism (defines maximum complex data rate)
 - FFT length (powers of two / mixed-radix)
 - Arithmetic bit-widths
 - Twiddle-factor bit-width
 - Input buffering
 - Bit-reverser
 - Resource assignment (e.g. RAM and / or Multipliers versus Logic)
- Fully synchronous design
- IFFT control input
- HiSpeed and QuadSpeed cores available in the Xilinx FFT Library

Benefits

- Highly optimised design with minimum silicon usage
- Virtually any transform length possible
- Virtually any bandwidth possible for short transform lengths
- Parameterised HDL design may be modified by RFEL to meet specific requirements
- Fully tested Netlist delivery
- Fixed-point Matlab models provided to de-risk core integration process
- Reduced technical and timescale risk to project

Deliverables

Each delivery includes:

- Data Sheet and User Guide
- Netlist (Xilinx, Altera etc)
- Pre-compiled ModelSim libraries for behavioural and post-place and route VITAL models
- Bit-true Matlab Model
- VHDL Test Bench
- Constraints
- Place and Route reports

Arithmetic Precision and Scaling

RFEL's range of FFT cores use fixed-point arithmetic, which can be tailored to match the individual precision requirements of each application. Worst-case scaling is assumed in order to avoid arithmetic overflow, although specific scaling can be applied if required. Bit-growth can be defined for each processing stage so that an optimal core is realised in terms of both FPGA resource and arithmetic precision.

The cores are primarily intended for wide-band channelisation applications where the time-domain input data is sourced either directly from a fast ADC or from a Digital Down-Converter (e.g. a DHB core) that is connected to a fast ADC. In either case, data bit-widths and hence dynamic ranges, allow fixed-point arithmetic to be used without significant signal degradation.

Bit-true models are available for all FFT ranges that allow the system designer to assess arithmetic precision tradeoffs. RFEL can advise on processing precision for individual applications, considering effects such as SFDR (Spurious Free Dynamic Range), quantisation noise and low-level signal accuracy.

HiSpeed and QuadSpeed FFTs

HiSpeed and QuadSpeed FFT architectures are shown in Figure 2 and Figure 3 respectively.

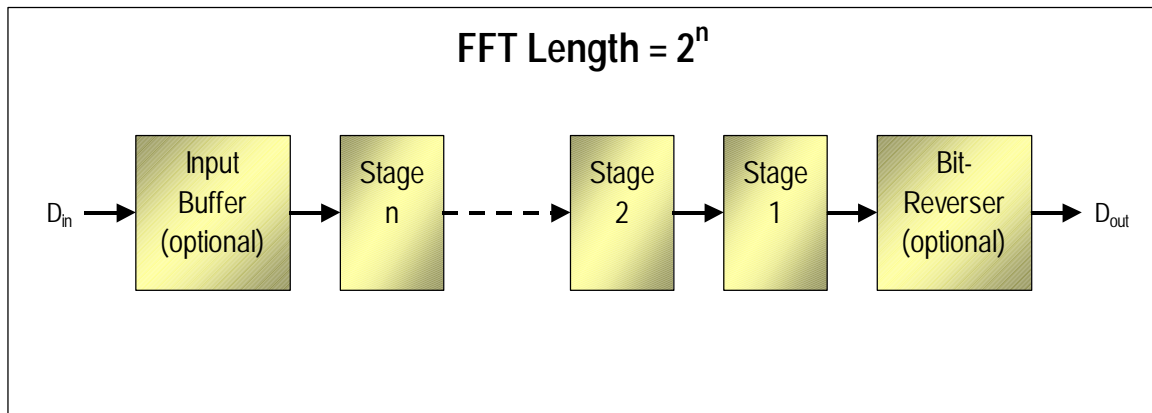


Figure 2: HiSpeed FFT Architecture

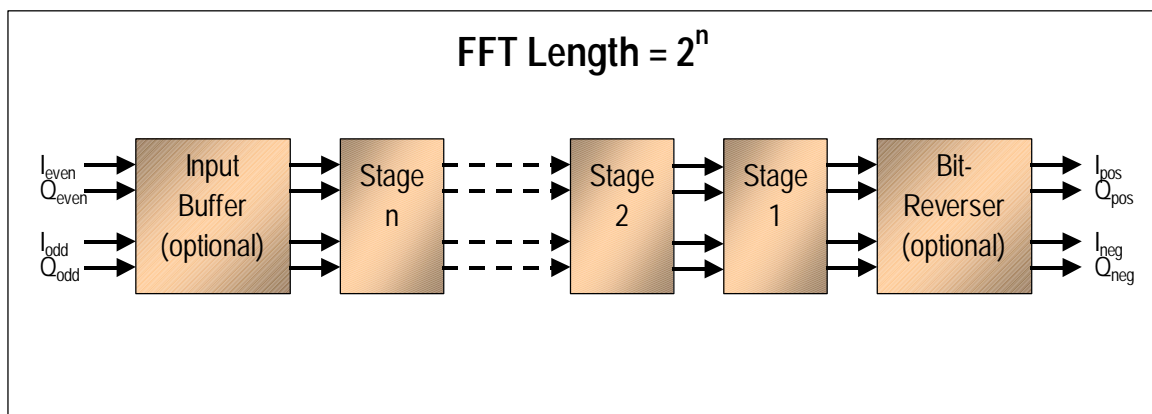


Figure 3: QuadSpeed FFT Architecture

The FFT pipeline is built from a number (n) of radix-2 processing stages, giving a complex FFT length of 2^n . Each pipelined stage implements a radix-2 butterfly, twiddle factor application and data re-ordering. Arithmetic overflow is avoided by always applying worst-case scaling. The position of the binary point can be chosen for each stage in order to achieve the desired arithmetic precision through the core.

The input data stream for both types of FFT core consists of normally ordered complex time-domain samples. The HiSpeed core requires In-phase (I) and Quadrature-phase (Q) components to be time interleaved to match the 'parallelism of 1' of the core. The QuadSpeed core requires odd and even complex samples to be de-multiplexed to match the parallelism of 4.

RFEL's range of Distributed Half Band Filter (DHBF) cores can be used to transform a real data stream into a complex data stream that can be fed directly into a HiSpeed or QuadSpeed FFT core. This combination of DHBF and complex FFT gives the most memory-efficient implementation of a real-input channeliser. If memory resource is not at a premium, then the complex FFT core can be used to implement a real input FFT using the 'different DIT' or 'two for the price of one' algorithms.

The optional input buffer function performs the first-stage of data re-ordering required by the FFT process. This block is optional since some applications may already have data storage prior to the FFT that can be used to perform the required re-ordering.

Frequency-domain samples are output from the final stage in bit-reversed frequency order. An optional bit-reverser block can be used to re-order the bit-reversed frequency samples into ascending frequency order.

The format of the HiSpeed FFT core's complex frequency-domain data output is time interleaved I and Q. The format of the QuadSpeed FFT core's complex frequency-domain data output is two parallel complex streams, one covering positive frequencies and the other covering negative frequencies.

Implementation resource tradeoffs are possible for each processing stage of the FFT. RFEL can trade the number of multipliers and / or RAMs with the amount of logic used for a specific build of the core. This can be very useful when one or more resource type is at more of a premium than another.

A selection of more than sixty HiSpeed and QuadSpeed FFT cores targeted at the most common requirements are available in the Xilinx FFT Library. Supplied on CD, this includes a range of FFT cores for different transform lengths, sample rates, and device types. Please refer to our website at www.rfel.com for more information about this product.

HiSpeed and QuadSpeed FFT Examples

A selection of HiSpeed and QuadSpeed FFT examples are shown in Table 1 and Table 2. 'Ref' numbers refer to the labels on Figure 1.

Ref	FFT Length	Input bit-width	Output bit-width	Twiddle bit-width	Input Buffer	Bit-Rev	B/W MHz	No. Slices	No. RAMs	No. Mults
1	1k	14	24	18	Yes	Yes	100	2661	12	12
2	4k	14	26	18	Yes	Yes	100	3373	36	16
3	8k	14	27	18	Yes	Yes	100	3732	68	18
4	32k	14	29	18	Yes	Yes	80	4550	263	22
5	128k	14	31	17	No	No	75	5078	324	26

Table 1: HiSpeed FFT Examples

Ref	FFT Length	Input bit-width	Output bit-width	Twiddle bit-width	Input Buffer	Bit-Rev	B/W MHz	No. Slices	No. RAMs	No. Mults
6	1k	12	22	18	Yes	Yes	480	4529	20	28
7	4k	12	24	18	Yes	Yes	460	6055	38	40
8	16k	12	26	18	Yes	Yes	400	8188	116	52
9	128k	12	29	18	No	No	380	11067	296	70

Table 2: QuadSpeed FFT Examples

Matrix FFT

The architecture of the Matrix FFT range is shown in Figure 4.

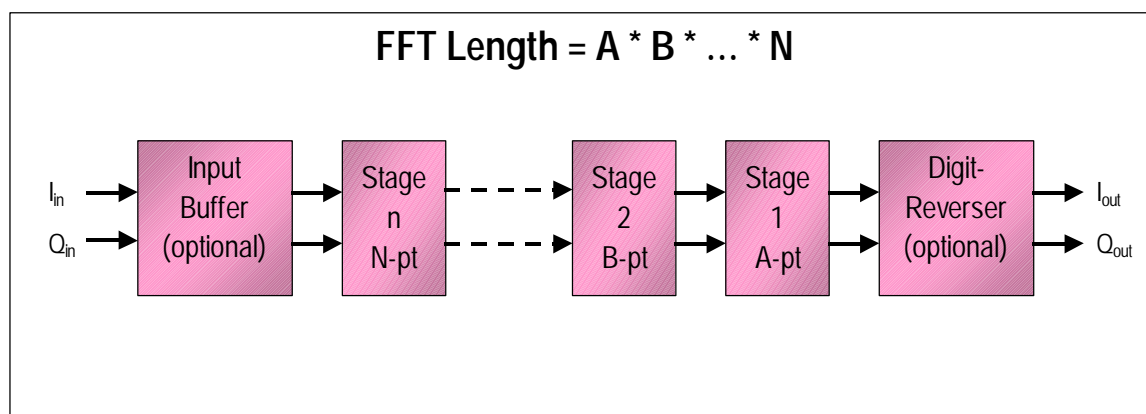


Figure 4: Matrix FFT Architecture

The Matrix FFT pipeline is built from a number of mixed-radix processing stages that give an FFT length equal to the product of all radices. Each pipelined stage implements DFT, twiddle factor application and data re-ordering. Careful choice of prime-length DFT implementation architectures and removal of twiddle factors on prime boundaries ensure minimal FPGA resource requirements. As with RFEL's other FFT ranges, arithmetic overflow is avoided by always applying worst-case scaling and the position of the binary point can be chosen for each stage in order to achieve the desired arithmetic precision through the core.

The input data stream consists of normally ordered parallel complex time-domain samples to match the pipeline parallelism of 2. RFEL's range of Distributed Half Band Filter (DHBF) cores can be used to transform a real data stream into a complex data stream that can be fed directly into a Matrix FFT core.

The optional input buffer function performs the first-stage of data re-ordering required by the FFT process. This block is optional since some applications may already have data storage prior to the FFT that can be used to perform the required re-ordering.

Frequency-domain samples are output from the final stage in digit-reversed frequency order. An optional digit-reverser block can be used to re-order the digit-reversed frequency samples into ascending frequency order.

RFEL have a growing library of pipelined prime-length DFT processing elements that can be cascaded to implement many unusual transform lengths. The Matrix architecture can implement any¹ transform length in an optimal silicon efficient manner.

Matrix FFT Example

Ref	FFT Length	Input bit-width	Output bit-width	Twiddle bit-width	Input Buffer	Bit-Rev	B/W MHz	No. Slices	No. RAMs	No. Mults
10	1872	16	18	18	Yes	Yes	190	9506	42	28

¹ Providing prime-length DFT processing elements and sufficient resource is available.

HyperSpeed FFT

The architecture of the HyperSpeed FFT is shown in Figure 5.

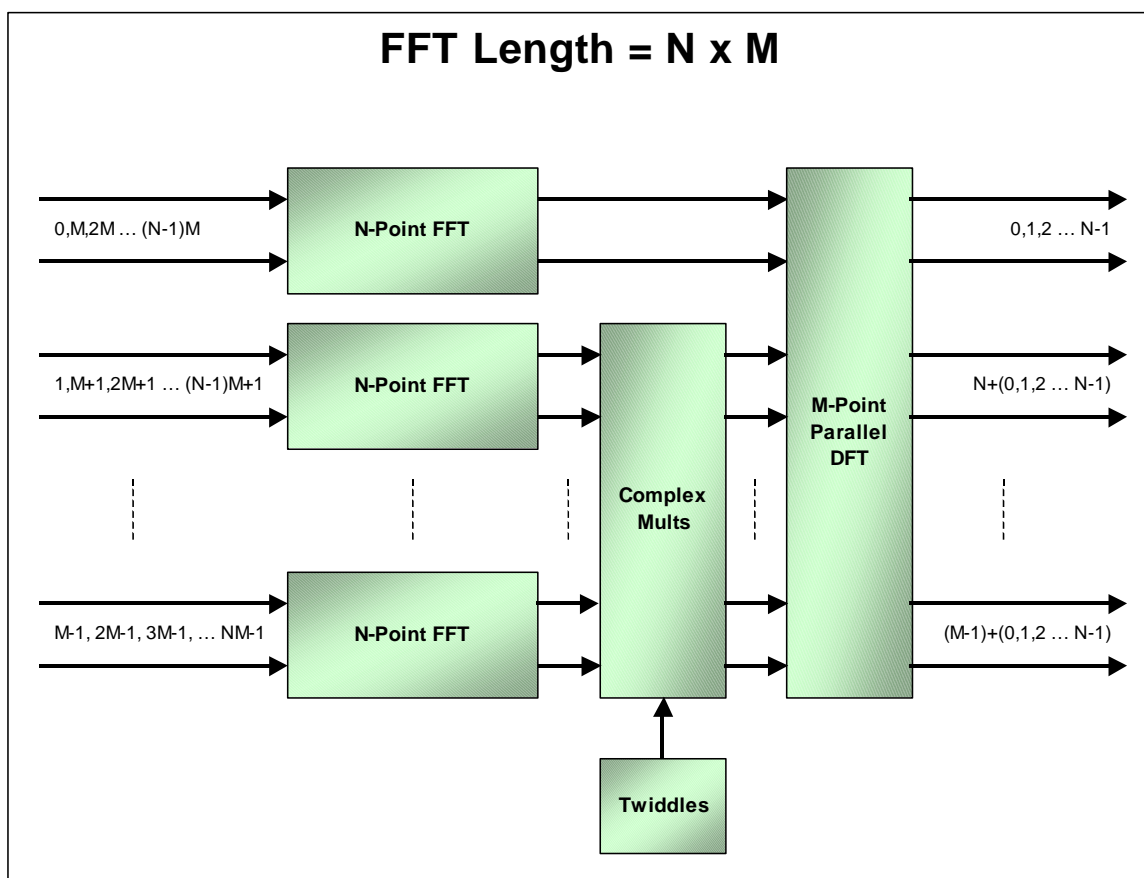


Figure 5: HyperSpeed FFT Architecture

The FFT pipeline is built from 'M' individual FFTs, each with a transform length of 'N'-points. Common resources are shared between the individual FFTs (such as twiddle factor generation, control etc). Frequency-domain outputs from the N-point FFTs are 'twiddled' and combined using an M-point DFT, which is implemented as a fully parallel pipelined FFT.

The HyperSpeed architecture can support transform lengths that are integer powers of two using existing 'off-the-shelf' building blocks. Other transform lengths can be built using Matrix FFT prime-length building blocks.

The input data stream consists of normally ordered parallel complex time-domain samples that have been de-multiplexed to match the pipeline parallelism of the core. This de-multiplex function is typically implemented in several stages. The first stages are normally implemented external to the FPGA using specialist devices that reduce the speed of the interface to a manageable rate (say 500MHz, or preferably less). The second stages of de-multiplexing are then implemented within the FPGA to reduce the data rate still further to a speed that can be processed using the architecture shown.

RFEL's range of highly parallel Distributed Half Band Filter (DHBF) cores can be used to transform a sample de-multiplexed real data stream into a sample de-multiplexed complex data stream that can be fed directly into a HyperSpeed FFT core.

The format of the HyperSpeed FFT core's complex frequency-domain data output is M x naturally ordered parallel complex streams, each covering a region of the spectrum that spans a frequency F_s/M .

HyperSpeed FFT Examples

Ref	FFT Length	Input bit-width	Output bit-width	Twiddle bit-width	Input Buffer	Bit-Rev	B/W GHz	No. Slices	No. RAMs	No. Mults
11	1k	6	20	14	N/A ¹	N/A ²	6	27634	116	285
12	32k	8	20	14	N/A	N/A	3	25596	220	285
13	32k	10	24	14	N/A	N/A	1	12663	240	144

¹ The HyperSpeed architecture always includes input buffer functionality

² The HyperSpeed architecture always includes bit-reversal functionality

HyperLength FFT

The architecture of the HyperLength FFT is shown in Figure 6.

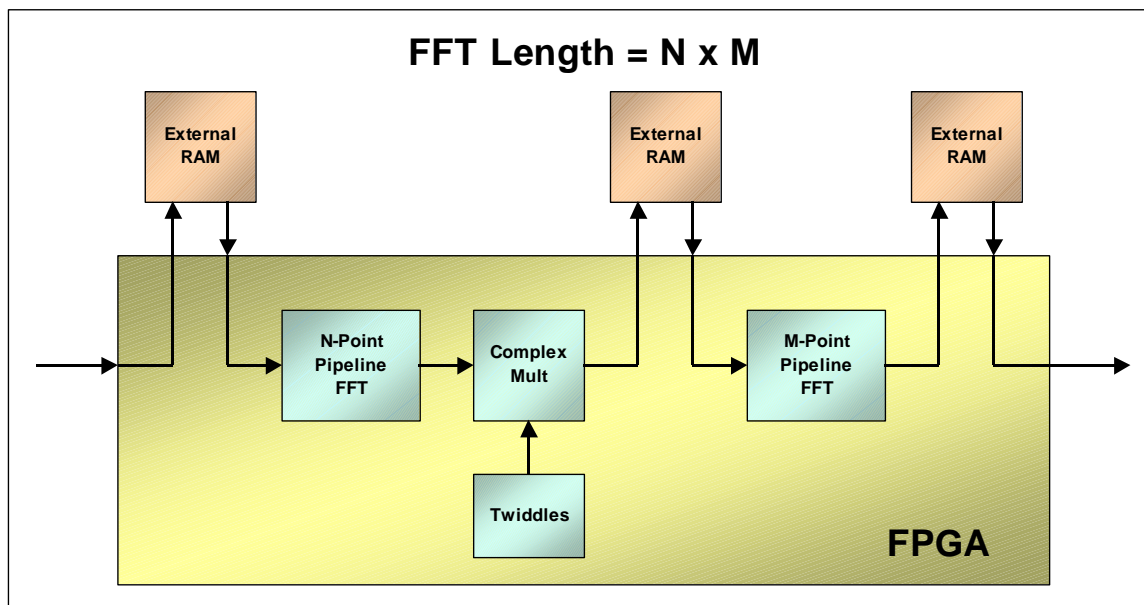


Figure 6: HyperLength FFT Architecture

The HyperLength FFT is designed to implement very long transforms with minimum memory bandwidth. Minimising memory bandwidth maximises processing bandwidth for a given FPGA + external RAM hardware configuration. The actual hardware configuration will dictate what transform lengths and continuous processing bandwidths are possible.

The architecture uses a pair of pipeline FFTs (usually HiSpeed, QuadSpeed or Matrix FFTs) to implement an N-point and an M-point DFT, that together with data re-ordering and twiddle factor application form an (N x M)-point FFT. External RAM is used to perform three data re-ordering stages as shown. Twiddle factors are generated arithmetically using logic within the FPGA rather than being stored in external memory.

Each data re-ordering stage performs a corner-turn type permutation, which can be efficiently implemented using SRAM type devices that have full bandwidth random address access (QDR, ZBT etc). Unfortunately SRAM devices are currently only available in sizes of a few MBytes, which limits a realistic SRAM-based HyperLength design to around a million points.

SDRAM type devices are available in sizes approaching GBytes, but they do not support high bandwidth random address accesses, limiting the maximum continuous bandwidth of the FFT. Implementing the data re-ordering as a two-stage process that uses some of the internal FPGA memory can improve the bandwidth, depending on the exact hardware configuration and the resources available. A relatively new SDRAM variant called RLDRAM may provide a good compromise between FFT length and bandwidth.

The input data stream consists of normally ordered complex time-domain samples, formatted into the correct form for the pipeline FFT input format. Frequency-domain samples are output from the M-point FFT in digit-reversed frequency order. The final data re-ordering stage is used to re-order the digit-reversed frequency samples into ascending frequency order.

HyperLength FFT Examples

Ref	FFT Length	Input bit-width	Output bit-width	Twiddle bit-width	B/W MHz	External RAM	No. Slices	No. RAMs	No. Mults
14	1M	16	32	18	200	5 x 36Mbit QDRII	11243	42	62
15	256M	16	32	18	10	3 x 4GB DDR DIMM	11644	278	42



rfengines limited

Innovation Centre
St Cross Business Park
Newport, Isle of Wight
PO30 5WB
United Kingdom

Tel: +44 1983 550330
Fax: +44 1983 550340
Email: info@rfel.com
Website: www.rfel.com