An Efficient Architecture for Ultra Long FFTs in FPGAs and ASICs

Tom Dillon
Dillon Engineering, Inc.

This presentation outlines an architecture for efficient Ultra Long FFTs for use in FPGAs and ASICs. Analysis of accuracy, performance, cost and power consumption are presented.

FFTs are at the heart of many real time signal processing applications and Ultra Long FFTs are quite often used for frequency analysis and communications applications. As the processing requirements increase, the use of FPGAs and ASICs become the logical choice for implementing real time FFTs.

This presentation describes and efficient framework for implementing the Cooley-Tukey algorithm for Ultra Long FFTs using minimal external memory. Typically for lengths over 16K the memory resources of the FPGA or ASIC are exhausted and external memory is required. The architecture is implemented using two shorter length FFTs (lengths N_1 and N_2) to calculate an FFT of length $N=N_1 \times N_2$. This architecture is optimized for continuous data FFTs, minimizing the external memory requirements and offering flexibility so that it can be used for many different applications.

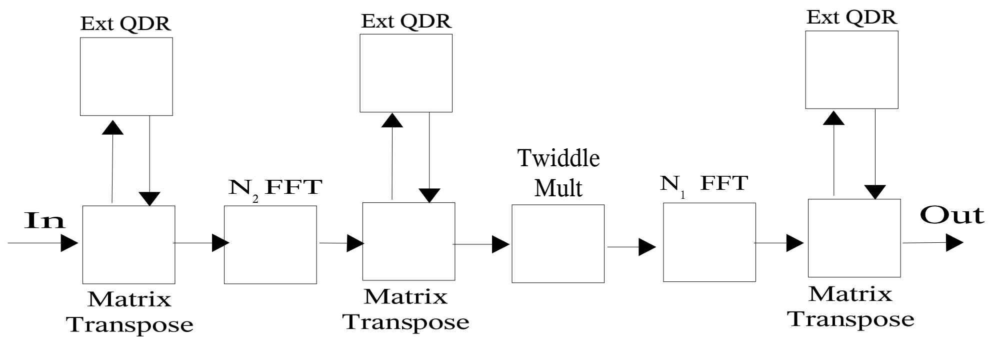
The $(N_1 \times N_2)$ -point FFT can be computed as

$$X[k_1 N_2 + k_2] = \sum_{n_1=0}^{N_1-1} [e^{-j\frac{2\pi n_1 k_2}{N}} (\sum_{n_2=0}^{N_2-1} x[n_2 N_1 + n_1] e^{-j\frac{2\pi n_2 k_2}{N_2}})] e^{-j\frac{2\pi n_1 k_1}{N_1}} .$$

Computing this for $0 \leqslant k_1 \leqslant N_1 - 1$ and $0 \leqslant k_2 \leqslant N_2 - 1$ results in

$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi nk}{N}}$ for $0 \leqslant k \leqslant N-1$, as desired. This leads to the following high-level architecture:

The input data is re-ordered by performing the equivalent of a matrix transposition. The next step is to compute N_1 FFTs, each of length N_2, followed by the second matrix transposition. The re-ordered data set is multiplied by the twiddle factors, and N_2 FFTs, each of length N_1, are computed. The final step is to perform the third matrix transposition so the output data is in the correct order.



The external memory requirements can be reduced by using QDR synchronous SRAM and an addressing sequence to allow a single bank of memory to be used for each matrix transposition. This presentation describes details of this addressing sequence. SRAM is a requirement because data needs to be read and written on every clock cycle and the addresses are usually not consecutive. QDR allows writing and reading different locations simultaneously, thereby removing the requirement for two banks of memory at each matrix transposition.

The potential data growth in longer length FFTs makes numerical analysis a necessity. A finite word length analysis will be presented for both fixed and floating point FFTs which will show that either floating point or fairly wide fixed point FFTs are required to maintain the precision required for most applications. These wider word lengths affect the memory architecture because wider word lengths require more memory bandwidth for the matrix transpositions. The trade-offs between word length requirements and memory architecture are discussed in the presentation.

This architecture can also function as 2D FFT by simply bypassing the twiddle multiply and removing the first Matrix Transpose.

A variable length FFT engine can be built form the same architecture by using variable length N_1 and N_2 FFTs and modifying the Matrix Transpose blocks. Often a run time length selection is desired so that the resolution can be adjusted.

Accuracy, component cost, and power consumption data will be presented for a system implemented in a single FPGA and three QDR SRAM ICs computing 512K FFTs on continuous data at 200MSPS.