Exploring the Latency-Resource Trade-off for the Discrete Fourier Transform on the FPGA

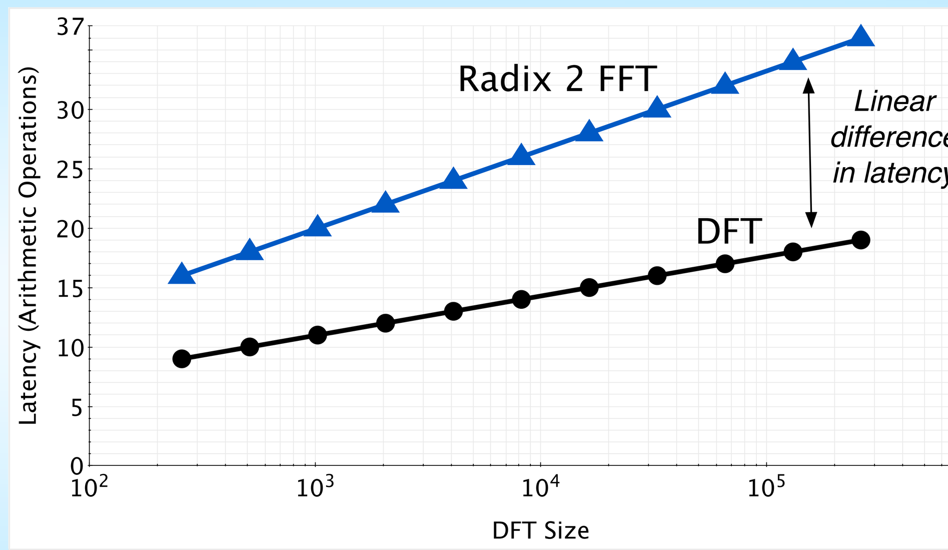
Gordon Inggs, David B. Thomas and Simon Winberg

Imperial College London and University of Cape Town

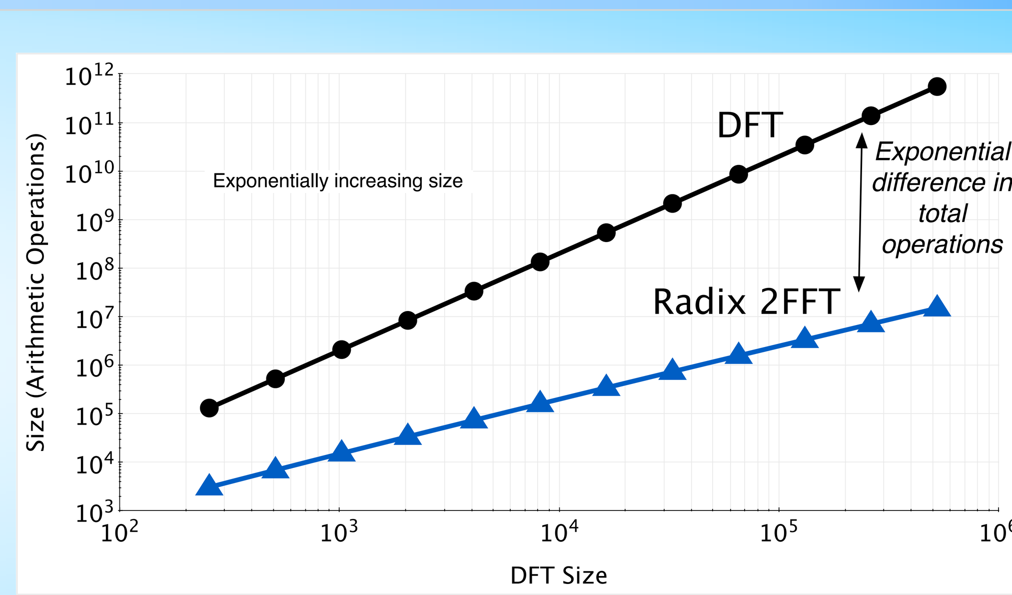
The DFT, FFT and Amadahl's Law

The Cooley-Tukey Radix 2 Fast Fourier Transform is a well-known optimisation of the Discrete Fourier Transform. The FFT trades a linear increase in the number of sequential operations for an exponential reduction in the arithmetic operations required.

As predicted by Amadahl's Law, increasing sequential operations results in an unavoidable increase in the absolute latency required to compute a single DFT. For certain applications such as real-time radar target tracking, the absolute latency of a DFT needs to be minimised.



Absolute Latency

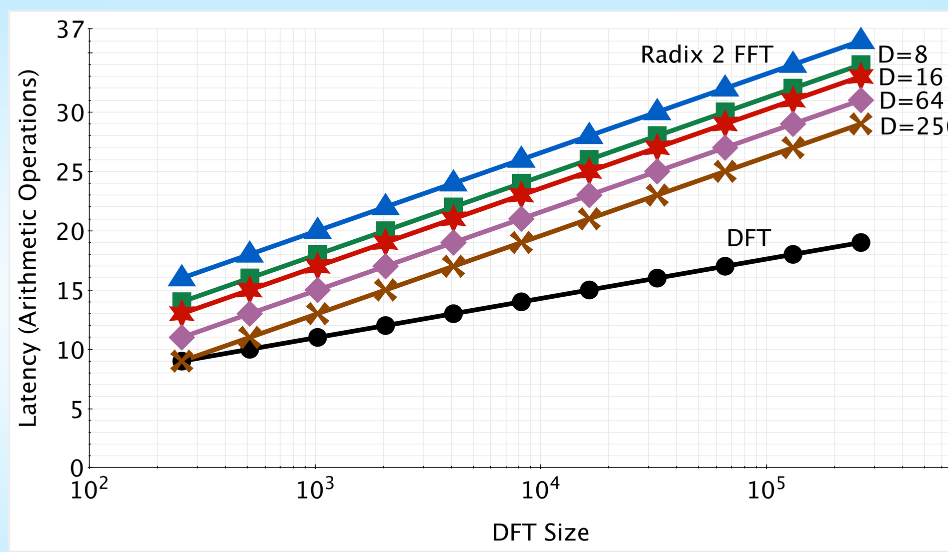


Total Operations

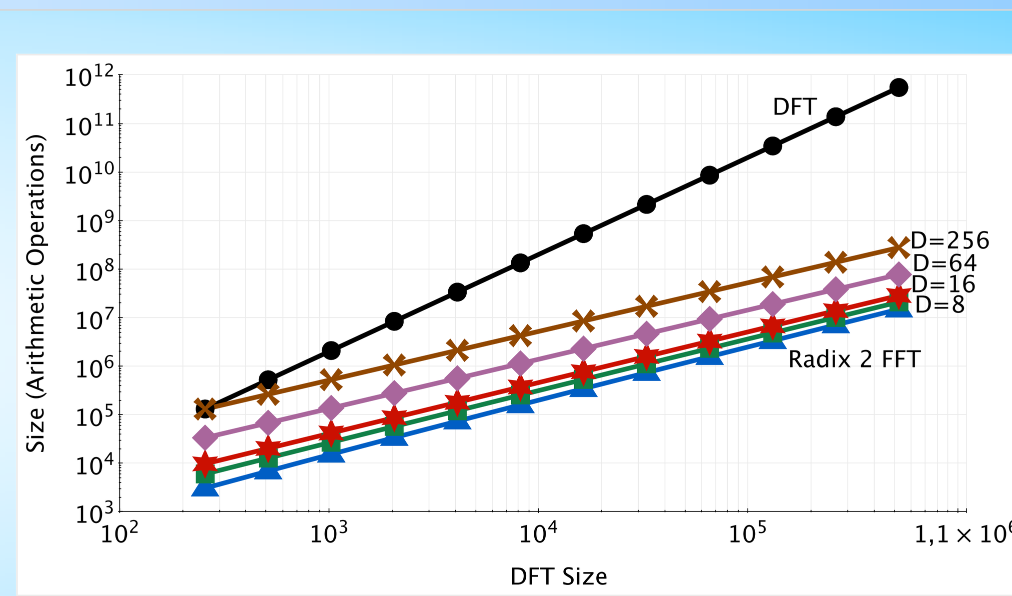
Proposed Flexible Radix FFT Algorithm

The key to the FFT optimisation is that a N point DFT is decomposed into $\log_2(N)$ 2 point DFTs, taking advantage of cyclical relationships in the complex coefficients by which the discrete signal is multiplied.

A flexible radix algorithm is proposed which decomposes a N point DFT down to a user-defined value, D, resulting in $\log_2(N/D)$ D point DFTs, providing control of the tradeoff between resources used and absolute latency in the DFT.



Absolute Latency

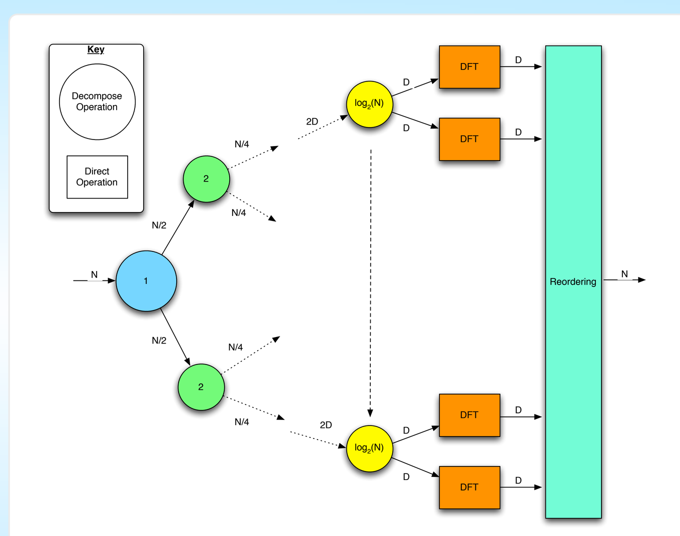


Total Operations

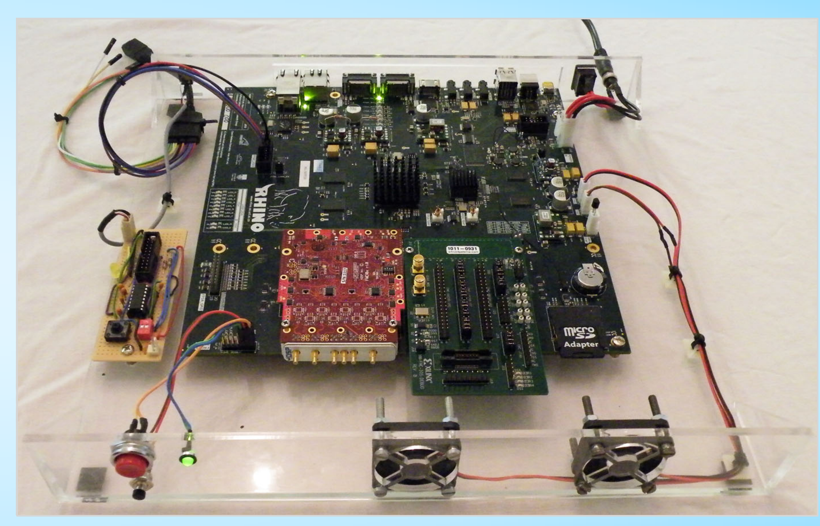
Experimental Implementation

A parameterisable implementation of the flexible radix algorithm was created using MyHDL (a Python-based HDL). The computing device targeted was the Rhino Platform, a Software Defined Radio research platform which has a Xilinx Spartan 6 SLX150T for compute purposes and Texas Instruments OMAP management processor.

To evaluate the algorithm, permutations of implementations were generated for DFTs of size 16-256 points, for the configuration parameter *D* set between *2* and *16*. In order to establish whether a tradeoff was established, the resource utilisation and absolute latency were measured for each permutation.



Operations Graph of Proposed Algorithm

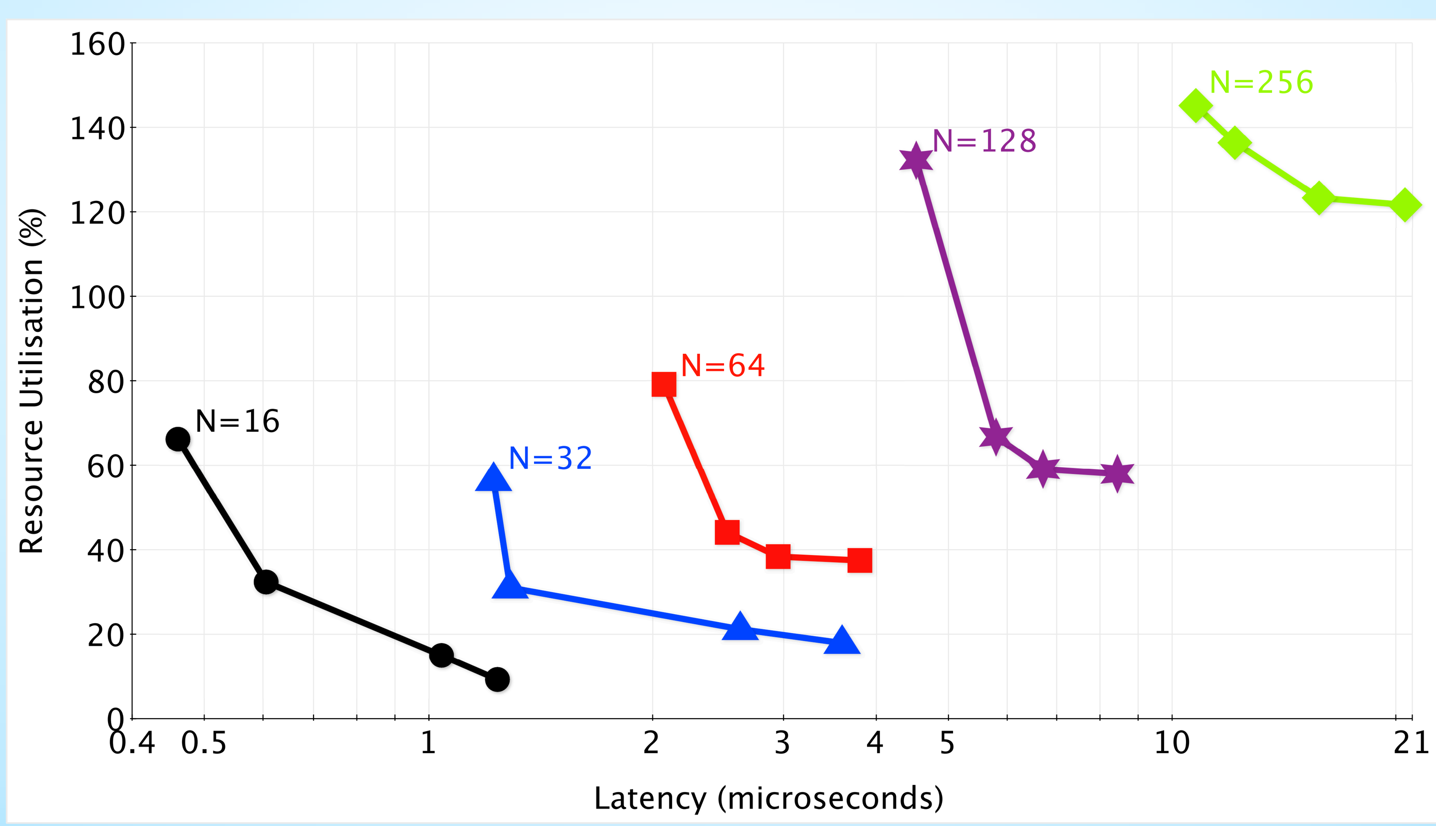


Rhino Platform in Test Rig

Results

With *D*=16, the *256* point implementation of the algorithm running at 142MHz on the Rhino platform is 14% faster in computing a single DFT than the corresponding radix 4 Xilinx LogiCore on the same FPGA. This implementation does however use 4 times the resources of the Xilinx LogiCore.

As illustrated below, the results across the experiment illustrate a regular and predictable trade off between absolute latency and resource utilisation, allowing for greater control of this tradeoff than a regular FFT.



Absolute Latency-Resource Tradeoff Curves

Future Work

The two main directions for future work:

1) Refining the MyHDL implementation framework, particularly with respect to resource utilisation.
2) Introducing finer control over optimisations such as operation pipelining

Git repository of the MyHDL-based Toolflow for the Rhino Platform, used to implement the algorithm.

Acknowledgements

The South African National Defense Force's Project Ledger, the South African National Research Foundation, the Oppenheimer Memorial Trust and the South African Square Kilometer Array