Tutorial 3 - Discrete Walsh Transform Processor (solution)

Philip Leong

July 10, 2017

1 INTRODUCTION

The discrete Walsh transform (DWT) takes $O(N^2)$ addition/subtraction operations to execute. This can be reduced to $O(N \log_2 N)$ using the fast Walsh transform (FWT). The algorithm, explained in reference [1] is analogous to the Cooley-Tukey algorithm for the fast Fourier transform.

2 LABORATORY QUESTIONS

1. Fast Walsh transform processor (30%). Make a combinatorial, parallel implementation of an $N = 64$ FWT processor for the Altera Cyclone V 5CSEMA5 FPGA used in the DE1-SoC board. Your inputs should be 16-bit integers in two's complement form, and your output represented as a two's complement fraction with sufficiently large wordlength that overflow cannot occur.

   Create a set of random test vectors and verify that your design is correct via simulation. The FPGA design tools report the maximum clock rate, f_{max} which can be achieved by your design. What is this value? The maximum throughput is thus $2N f_{max}$ bytes/sec. Calculate the throughput of your design.

2. Pipelined FWT processor (30%). Modify your FWT processor so that it is pipelined and verify via simulation. What is the new design's maximum throughput? What is the speedup compared with the non-pipelined design?

3. Multicycle execution (40%). It may not be feasible to supply the FWT processor with 64 high-speed, parallel inputs. Develop a modified version of the FWT processor which takes 2 inputs samples per cycle, i.e. it takes $N/2$ cycles to obtain a complete input vector. Redesign your processor so that it minimises the area-delay product (area being measured in LUTs) and can process streaming input data without stalling. What is the maximum performance in bytes/sec?

4. Comparison (bonus 20%). Integrate the FWT processor with a waveform generator input source. In real-time print out the FWT coefficients of the input signal. What is the maximum speed that you can achieve?

3 SOLUTIONS

REFERENCES

[1] J.L. Shanks. Computation of the fast walsh-fourier transform. *IEEE Transactions on Computers*, 18(5):457–459, 1969.