# Homework 3: Optimization of FIR Filter Design

Zach Neveu

February 21, 2020

## Designs

For this homework, 12 different HLS designs for an FIR filter were synthesized. These designs all contain variations of loop unrolling, loop fission, changing memory type, datatype optimization, and changing position of control flow in loops. The full list of experiments run is shown in Table 1.

Table 1: List of Experiments and Descriptions

| Design | Data/Coeffs. Bit Width | Unroll Factor | Fission | BRAM Ports | Control in Loop |
|---|---|---|---|---|---|
| basic | 10 | 1 | 0 | 0 | 1 |
| fission | 10 | 1 | 1 | 0 | 0 |
| remove_if | 10 | 1 | 0 | 0 | 0 |
| unroll_2 | 10 | 2 | 1 | 0 | 0 |
| unroll_all | 10 | 11 | 1 | 0 | 0 |
| unroll_all_2pbram | 10 | 11 | 1 | 2 | 0 |
| basic_1pbram | 10 | 1 | 1 | 1 | 1 |
| unroll_all_1pbram | 10 | 11 | 1 | 1 | 0 |
| 32b_unroll_all | 32 | 11 | 1 | 0 | 0 |
| 32b_unroll_2 | 32 | 2 | 1 | 0 | 0 |
| 32b_fission | 32 | 1 | 1 | 0 | 0 |
| 32b_basic | 32 | 1 | 0 | 0 | 1 |

## Results

Latency and Area were calculated for each synthesized filter design. For the purposes of this report, area was calculated according to formula 1. Latency and Area were then compared using a Pareto plot which can be seen in 1.

$$Area = 100 * BRAM + 100 * DSP + MAX(FF, LUT) \tag{1}$$

## Discussion

The lowest latency design was `unroll_all` with `unroll_all_2pbram` as a close second. This shows that unrolling loops can seriously improve latency when necessary. The lowest area design was `basic`. Only one other point, `remove_if` appears on the Pareto frontier. This design uses negligibly more

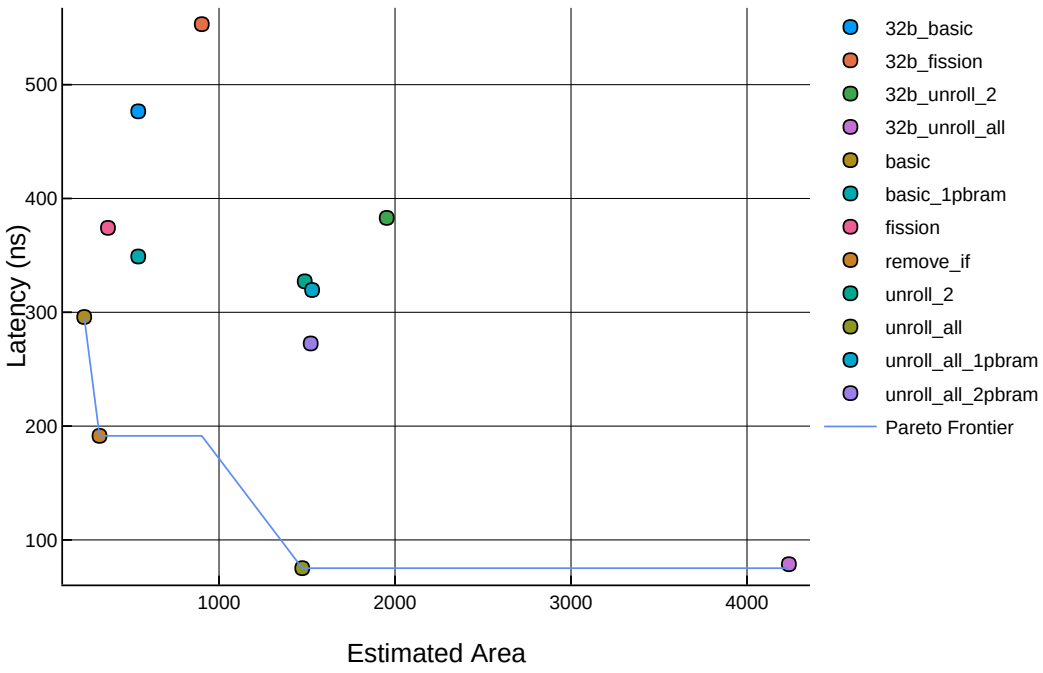


Figure 1: Pareto Chart of Designs

resources than `basic`, and decreases latency by more than 30%, so this design, in my opinion, is the best trade-off between area and latency. Based on these results, it appears that removing control flow from loops, and unrolling loops provide the largest performance trade-offs. While loop unrolling requires significantly more resources, removing control flow from loops comes at almost no cost. Finally, the cost of using non-optimized data types is quite significant. No 32 bit designs appear on the Pareto frontier and most lie quite far from it.