1. Архитектура решения

Приведённые выражения дисперсий для реализации вычислений квадратов сумм и сумм квадратов в потоке:

$$var\big(S(1,k)\big) = \left(\frac{\sum_{l=1}^{k}\big(S(l)\big)^2}{k} - \frac{\big(\sum_{l=1}^{k} S(l)\big)^2}{k^2}\right) \quad (3)$$

$$var\big(S(k+1,N)\big) = \left(\frac{\sum_{l=1}^{k+1}\big(S(l)\big)^2}{N-k+1} - \frac{\big(\sum_{l=1}^{k+1} S(l)\big)^2}{(N-k+1)^2}\right) \quad (4)$$

В формулах имеется деление и известно, что на ПЛИС такая операция трудно реализуется, в отличии от умножения. В таких случаях как правило, операция деления $K = \frac{A}{B}$ заменяется умножением на фиксированную константу по формуле $K = (A \cdot B) \gg n$, где $K = \frac{2^n}{B} + 1$, операция ($\gg$) – битовый сдвиг вправо.

Рассмотрим полученную в ходе работы архитектуру решения (приложение А).

В момент включения ПЛИС, происходит инициализация глобальных регистров и первых регистров массивов сумм из входного потока данных. Затем, начиная со следующего такта происходит выдача результата вычисления левой дисперсии. Это вычисление происходит каждый такт, в котором одновременно производятся суммирование и возведение в квадрат данных со входа, запись результатов в текущую ячейку массивов, умножение на фиксированную константу и квадрат константы, выдача результатов в выходной порт. По окончанию N тактов (равных ширине окна), происходит переключение на второй подобный блок, вычисляющий правую дисперсию.

В итоге, все вычисления происходят за 2*N+1 тактов.

2. Используемые инструменты

Для проектирования на ПЛИС в данной работе использовалась САПР Xilinx Vivado и язык программирования Hascol (Hardware Software Co-Design Language), который является альтернативой HDL (Hardware Description Language) и предоставляет программисту доступные средства для явного описания конвейерных и параллельных конструкций, чего нет в VHDL, но при этом наследуя от него возможность описания параллельно работающих процессов, обменивающихся между собой сообщениями. В языке имеется поддержка управления регистрами LUT и блоками BRAM, организация обработчиков в процессах, локальные и глобальные переменные, конвейерная семантика, а также возможность подключать и обмениваться сообщениями с внешними модулями HDL описаний. Язык расширен встроенным макропроцессором для генерации монотонно повторяющихся в коде параллельных конструкций.

Компилятор языка преобразует код, написанный на Hascol в код на HDL, который импортируется в проект Vivado, после чего компилируется и отлаживается средствами САПР в процессе моделирования (симуляции) спроектированной программы.

3. Организация считывания и записи данных

В работе стояла задача загрузки потока тестовых данных для симуляции и было найдено решение в виде подключаемой в HDL библиотеки TextIO. В связи с этим можно создать на языке VHDL блок с описанием процесса, в котором будет потактово (по изменению фронта сигнала clk) происходить считывание из сигнала типа file во внутренние регистры для дальнейшего использования.

Hascol поддерживает подключение внешних модулей с помощью создания их экземпляра в коде программы, в котором параметр external задаёт название файла с описанием архитектуры VHDL блока. В самом

экземпляре описываются входные и выходные порты, связываемые с основным (Top) модулем. Таким образом, программа на языке Hascol из Top модуля может обмениваться сообщениями с HDL модулем.

Также библиотека Textio позволяет записывать в файл результаты вычислений. В работе это использовалось для анализа результатов работы программы.

4. Описание реализации

Поток входных данных имеет ширину 16 бит, поэтому массив сумм имеет 24 бит а массив хранения квадратов сумм 32 бит. Ширина окна (всего поступивших точек) равна 64.

Из формулы (3) видно, что число делителей задано размером окна N, что позволяет предвычислить все фиксированные константы (коэффициенты) K и поместить их в регистры. То же самое делается и с квадратами коэффициентов. Параметром Points задаётся ширина окна данных и он определяет их количество. Размерность коэффициентов задана числом 2^{12} во избежание накопления ошибок деления. Инициализация коэффициентов реализована в первом такте, параллельно с записью из входного потока данных первых регистров массивов сумм.

Значение каждого коэффициента просчитывается предварительно препроцессором языка Hascol, который генерирует необходимый код (в данном случае инициализации регистров) прямо в тексте программы. Чтобы сделать деление, исходное число расширяется нулевыми значениями в старших битах, делается сдвиг вправо на 12 бит и умножается на соответствующий ему коэффициент операцией умножения с расширением.

В результате, исходя из формул (3) и (4), было реализовано потоковое вычисление выборочных дисперсий с помощью операций суммирования, умножения (с возведением в квадрат) и вычитания в соответствии с архитектурой, приведённой в приложении А. Вычисления производятся в

беззнаковом целочисленном формате данных, результат выдаётся шириной 64 бит.

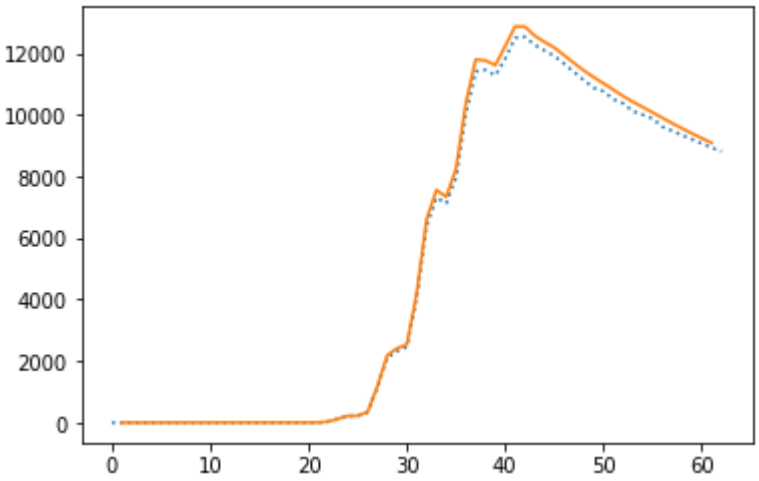

График значений левой дисперсии

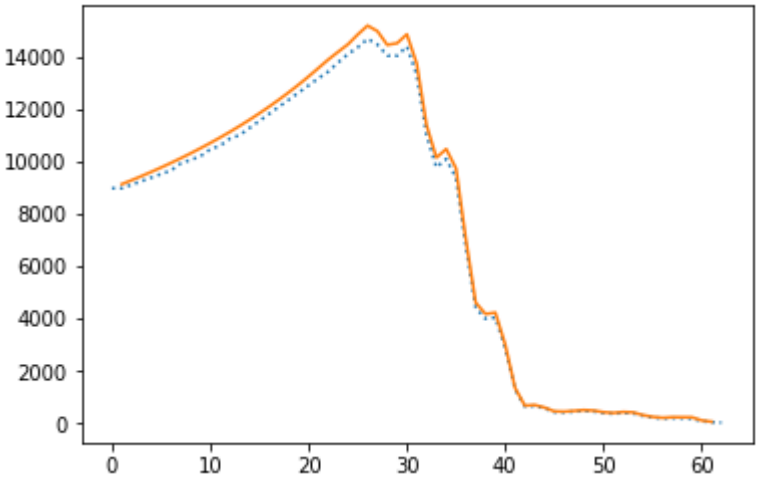

График значений правой дисперсии

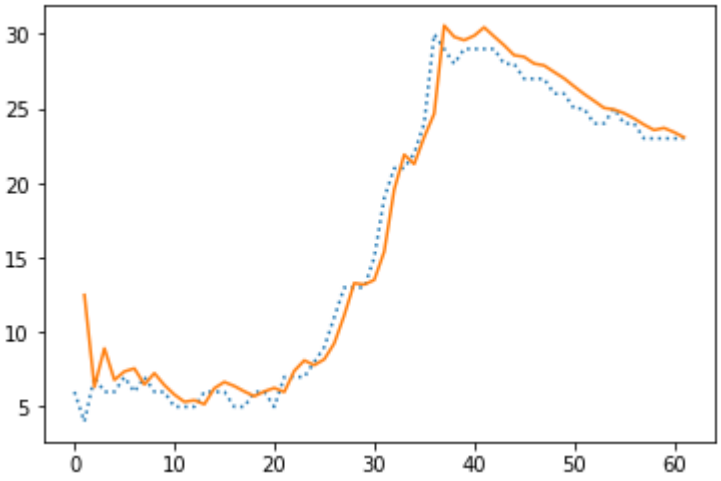

График значений левой дисперсии в условиях зашумлённых данных

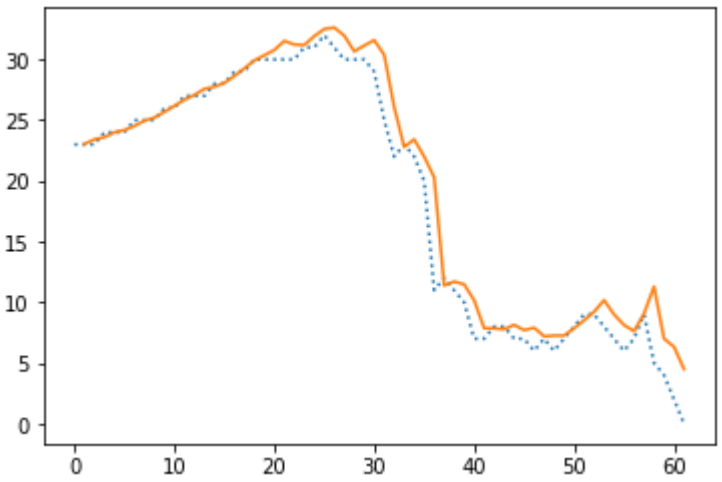

График значений правой дисперсии в условиях зашумлённых данных

Приложение А