Данилов Дмитрий Александрович
студент, Казанский национальный исследовательский технический университет им А.Н. Туполева,
Россия, г. Казань
Шалагин Сергей Викторович
научный руководитель, канд. тех. наук,
Казанский национальный исследовательский технический университет им А.Н. Туполева,
Россия, г. Казань

РЕАЛИЗАЦИЯ ДИСКРЕТНОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ В СРЕДЕ ПЛИС КЛАССА FPGA

Аннотация: Статья посвящена модулю, выполняющему функцию дискретного преобразования Фурье на ПЛИС класса FPGA. Модуль будет реализован через VHDL код, а также будет использоваться встроенные элементы САПР. Для реализации модуля будет использована САПР Quartus Prime Lite Edition.

Ключевые слова: Быстрое преобразование Фурье, САПР, ПЛИС, Quartus Prime, VHDL.

Abstract: The article is devoted to a module performing the discrete Fourier transform function on FPGA-class FPGAs. The module will be implemented via VHDL code, and built-in CAD elements will also be used. The Quartus Prime Lite Edition CAD system will be used to implement the module.

Keywords: Fast Fourier transform, CAD, FPGA, Quartus Prime, VHDL.

В настоящее время актуальна задача обработки изображений в реальном масштабе времени. Возможности программной реализации алгоритмов, реализующих данную задачу на ЭВМ общего назначения, ограничены возможностями фон-неймановской архитектуры. Выходом из сложившейся ситуации является применение специализированных ЭВМ. В частности – ЭВМ,

включающих в свой состав аппаратные ускорители как ASIC, так и программируемые логические интегральные схемы класса FPGA (ПЛИС).

Дискретное преобразование Фурье представляет собой численный алгоритм представления функций разложением в ряд по тригонометрическим функциям.

Рассмотрим двумерное дискретное преобразование Фурье (ДПФ) для массива чисел $\{x_{n|n}\}=X$ размерности NxN, $N=2^a$. Оно представимо следующим образом:

$$G_{uv} = N^2 \sum_{n\,0}^{N\,1} \sum_{m\,0}^{N\,1} x_{mn} * W_N^{mu*nv},$$

где $W_N^{mu|nv} = \exp\left(-2\pi j \frac{mu+nv}{N}\right), u = \overline{0, N-1}, v = \overline{0, N-1}, \{G_{uv}\}_{\frac{N}{N}} = G$ – образ для X. По аналогии выполняется обратное двумерное быстрое преобразование Фурье (БПФ).

$$x_{n|n} = \sum_{u\,0}^{N\,1} \sum_{v\,0}^{N\,1} G_{uv} * W_N^{mu|nv}, m = \overline{0, N-1}, n = \overline{0, N-1}$$

Алгоритм 2-БПФ реализуется на основе конвейера, включающего L ступеней, $L = \log_2 N$. На каждой ступени l, $l = \overline{1, L}$, требуется вычислить N/2 однотипных преобразований, реализуемых согласно формуле вида

$$\begin{cases} s(k) = s_0(k) + s_1(k + N \cdot 2^{l-L-1}) \cdot W_B^k \\ s(k + N \cdot 2^{l-L-1}) = s_0(k) - s_1(k + N \cdot 2^{l-L-1}) \cdot W_B^k, \end{cases} \quad (1)$$

где: $W_B^k = \exp\left(-j \frac{2\pi k}{B}\right)$,

$B = N * 2^{l-L}$,

$k = \overline{0, \frac{N}{2} - 1}$,

$l = \overline{1, L}, N = 2^z$,

z – целое натуральное число.

На вход алгоритма, который включает два этапа, подан массив чисел X={x_i}, $i = \overline{1,N}$

Этап 1. Над *X* выполняется двоично-инверсная перестановка, результатом которой является новый массив $X = \left\{X_p^{(1)}\right\} : X_p^{(1)} = x_i, i = \overline{1,N}$ p получено согласно выражению:

$$p = \left(p(1)\dots p(n)\right) = \left(i(1)\dots i(n)\right) \cdot \begin{pmatrix} 0 & \dots & 0 & 1 \\ 0 & \dots & 1 & 0 \\ \dots & \dots & \dots & \dots \\ 1 & \dots & 0 & 0 \end{pmatrix}$$

Например, для N=256, i = (00010110) соответствует p = (01101000)

Этап 2. Применение на каждой из *L* ступеней, $L = \log_2 N$, N / 2 однотипных преобразований вида (1). На каждой ступени под номером *l*, $l = \overline{1,(L-1)}$, в качестве входных значений в формуле (1) выбираются два значения из массива $X^{(l)}$ под номерами *k* и $k + N * 2^{l-L-1}$, а результаты записываются в массив $X^{(l+1)}$ под теми же номерами, $k = \overline{0,\frac{N}{2}-1}$. На *L*-й ступени конвейера полученные результаты выбираются из массива $X^{(L)}$ и размещаются в массиве C={C_p}, $p = \overline{1,N}$. Массив *C* есть искомое значение – спектр цифрового сигнала.

Для реализации БПФ над массивом из *N* отсчетов требуется *N* / 2 однотипных блоков, реализующих операцию вида (1), а также не менее *L* регистров на *N*n* разрядов каждый для хранения $X^{(l)}$, $l = \overline{1,L}$, *n* – разрядность обрабатываемых двоичных чисел. Кроме того, требуются дополнительные регистры для хранения промежуточных результатов при конвейерном вычислении операции над комплексными числами согласно (1).

Можно представить систему в пункте 1.1, по аналогии с одномерным БПФ, в виде:

$$G_{uv} = N^2\left[S_{2n1*2m1} + W_N^u S_{2n1*1,2m1} + W_N^v S_{2n1*2m1*1} + W_N^{u|v} S_{2n1|1,2m1*1}\right] \text{ (2),}$$

где,

$$S_{2n1*2m1} = \sum_{n1\,0}^{N/2\,1} \sum_{m1\,0}^{N/2\,1} X_{2n1*2m1} * W_N^{2(n1u|m1w)}, u = \overline{0, N-1}, v = \overline{0, N-1}$$

В результате, двумерное БПФ реализуемо как комплекс однотипных преобразований «бабочка» (ПрБ), применяемых при вычислении одномерного БПФ. Для выполнения одной операции над четырьмя элементами массива чисел X, требуется четыре ПрБ. Указанные операции требуется выполнить над массивом из N^2 элементов $\log_2 N$ раз

Реализация модуля

Далее, мы рассмотрим реализацию модуля САПР в архитектуре ПЛИС класса FPGA.

На рисунке 1 изображена полная схема модуля быстрого преобразования Фурье. Она состоит из модуля умножения на константу ComplexConst, 3 регистра, 4 сумматора, два из которых выполняют функцию вычитания в обратном коде и 2 конвертера для увеличения разрядности сигнала с сохранением знака сигнала.

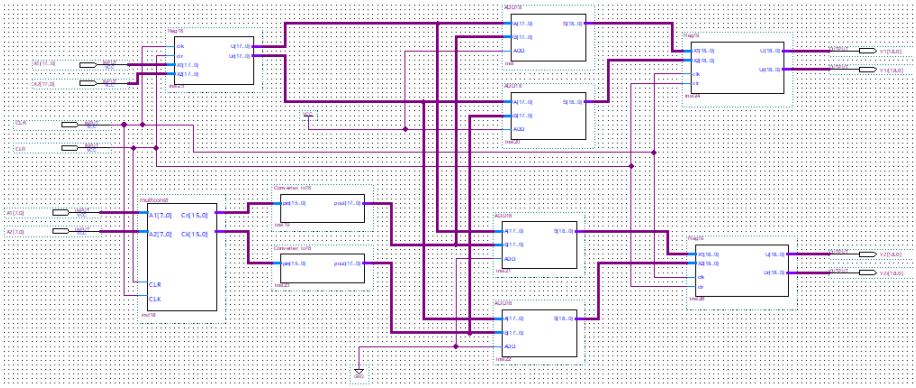

Рисунок 1. Модуль БПФ

На рисунке 2 изображен модуль ComplexConst, который служит для умножения поступающих чисел на константу.

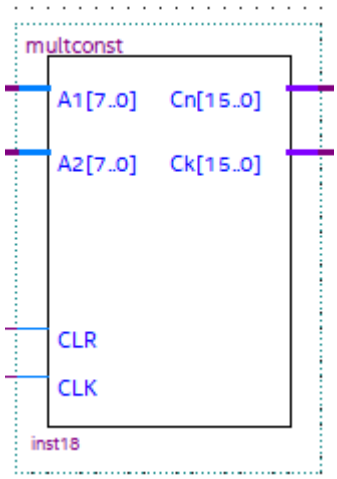

Рисунок 2. модуль ComplexConst

На выходе приходят два результата. Один реализует операцию сложения произведений чисел, другой – операцию вычитания.

На рисунке 3 изображена внутренняя составляющая модуля ComplexConst, состоящая из 4 умножителей, 6 промежуточных регистров и 2 сумматоров, один из которых выполняет функцию вычитания.

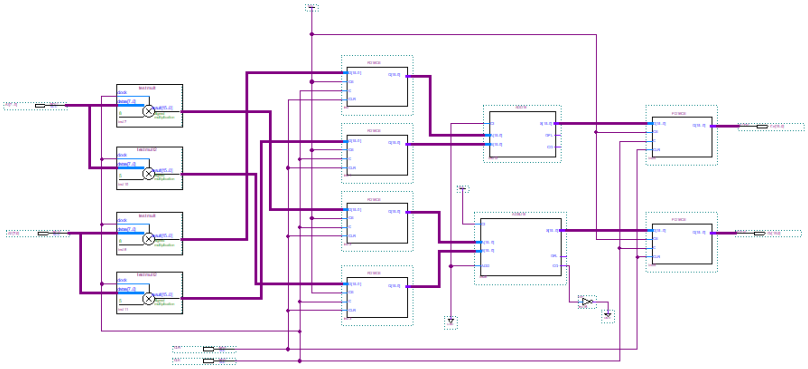

Рисунок 3. Внутреннее устройство ComplexConst

Список литературы:

1. Шалагин С.В. Распределённое вычисление быстрого преобразования Фурье в архитектуре FPGA / Шалагин С.В // Вестник Технологического университета. 2019. Т. 22. № 2. – С. 155-158 – EDN ZABNWH

2. Шалагин, С. В. Проектирование базовых IP-ядер для выполнения быстрого преобразования Фурье / С. В. Шалагин, Е. М. Кириллов // Актуальные проблемы математики и информационных технологий : Материалы I Всероссийской

конференции, Махачкала, 03–05 февраля 2020 года. – Махачкала: Дагестанский государственный университет, 2020. – С. 186-192. – EDN MYIXWH.
3. Егоров, И. А. Проблемы алгоритмов преобразования Фурье в обработке цифровых сигналов и изображений / И. А. Егоров, А. М. Кумратова // Информационное общество: современное состояние и перспективы развития: Сборник материалов XV международного форума, Краснодар, 10–14 июля 2023 года. – Краснодар: Кубанский государственный аграрный университет имени И.Т. Трубилина, 2023. – С. 293-297. – EDN HDNNUH