Алгоритм быстрого преобразования Фурье, реализация и оценка трудоемкости

Трудоемкость вычисления амплитуд комплексных составляющих $X(j)$ весьма значительна: N^2 операций комплексного сложения и умножения (без учета вычисления тригонометрических составляющих, для определения которых можно применить табличный способ). Поэтому для вычисления $X(j)$ применяют алгоритм быстрого преобразования Фурье (БПФ).

В основе алгоритмов, обеспечивающих снижение трудоемкости вычисления ДПФ лежат следующие соотношения:

$$W_N^{k\cdot n} = W_N^{k\cdot(n+N)} = W_N^{(k+N)\cdot n}; \qquad W_N^{k\cdot(N-n)} = W_N^{-k\cdot n}. \tag{4.36}$$

Алгоритмы, в которых исходная, размерности N, дискретная последовательность, представляющая сигнал, разбивается на меньшие последовательности, называются алгоритмами с прореживанием по времени. Сущность такого преобразования выглядит следующим образом.

$$X(j) = \frac{1}{N}\sum_{k=0}^{N-1} x(k)\cdot W_N^{jk} = \frac{1}{N}\sum_{l=0}^{\frac{N}{2}-1} x(2l)\cdot W_N^{2l\cdot j} + \frac{1}{N}\sum_{l=0}^{\frac{N}{2}-1} x(2l+1)\cdot W_N^{(2l+1)\cdot j} =$$

$$= \frac{1}{N}\sum_{l=0}^{\frac{N}{2}-1} x(2l)\cdot W_{\frac{N}{2}}^{l\cdot j} + \frac{1}{N}W_N^{j}\sum_{l=0}^{\frac{N}{2}-1} x(2l+1)\cdot W_{\frac{N}{2}}^{l\cdot j}. \tag{4.37}$$

В выражении (4.37) первая сумма является $N/2$ -точечным дискретным преобразованием Фурье для четных элементов, а вторая - для нечетных, последовательности $\{x(k)\}$. Суммы вычисляются по отдельности, а затем полученные результаты объединяются и дают N - точечное ДПФ. Трудоемкость этого вычисления $2\left(\frac{N}{2}\right)^2$ операций комплексного умножения и сложения плюс N комплексных умножений и сложений. Разбиение последовательностей можно продолжать до тех пор, пока в них не останется по два элемента. Тогда с учетом того, что

$$W_{N/2} = W_N^2; \qquad W_N^{N/2} = -1; \qquad W_N^{m+N/2} = -W_N^m$$

вычислительный алгоритм можно представить граф-схемой (пример для N=16), изображенной на рисунке 4.1.

Эту реализацию называют алгоритмом БПФ Кули-Тьюки с прореживанием по времени. Она состоит из последовательности ступеней, на каждой из которых выполняются однотипные действия над массивом комплексных данных, поступающим на вход ступени, а результаты, полученные в ходе преобразований, выполненных на текущей ступени, обратно перезаписываются в исходный массив, который является выходным для текущей ступени и входным для следующей ступени.

Основной вычислительной процедурой этого алгоритма является, так называемая, операция "бабочка", которая в комплексном виде представляется как

$$\begin{aligned} X_{m+1}(p) &= X_m(p) + W_N^r \cdot X_m(q) \\ X_{m+1}(q) &= X_m(p) - W_N^r \cdot X_m(q) \end{aligned}, \tag{4.38}$$

где m - номер ступени преобразования исходного массива данных.

Значения элементов входного массива на входе алгоритма соответствуют нулевой итерации.

Следует отметить, что массив данных на входе алгоритма представляет собой преобразованный массив исходных данных, а именно, выполнена перестановка местами большинства элементов.

Если массив исходных данных состоит из N элементов, то для представления номера элемента в двоичной системе счисления требуется $m = \log_2 N$ двоичных разрядов

$x(\alpha_{m-1}\alpha_{m-2}\ldots\alpha_2\alpha_1\alpha_0)$, здесь α_i - i -ый двоичный разряд номера.

При перестановке меняются местами элемент $x(\alpha_{m-1}\alpha_{m-2}\ldots\alpha_2\alpha_1\alpha_0)$ и элемент $x(\alpha_0\alpha_1\alpha_2\ldots{}_0\alpha_{m-2}\alpha_{m-1})$.

Такая перестановка называется двоичной инверсией, так как в преобразованной последовательности исходных данных двоичные номера элементов являются зеркальным отображением номеров элементов исходной последовательности.

Массив данных на выходе алгоритма БПФ получается упорядоченным и не требует дополнительных преобразований.

Для реализации алгоритма БПФ требуется $\log_2 N$ ступеней, а на каждой ступени выполняется $\frac{N}{2}$ операций "бабочка". Таким образом, трудоемкость всего алгоритма - $\frac{N}{2} \cdot \log_2 N$ операций "бабочка".

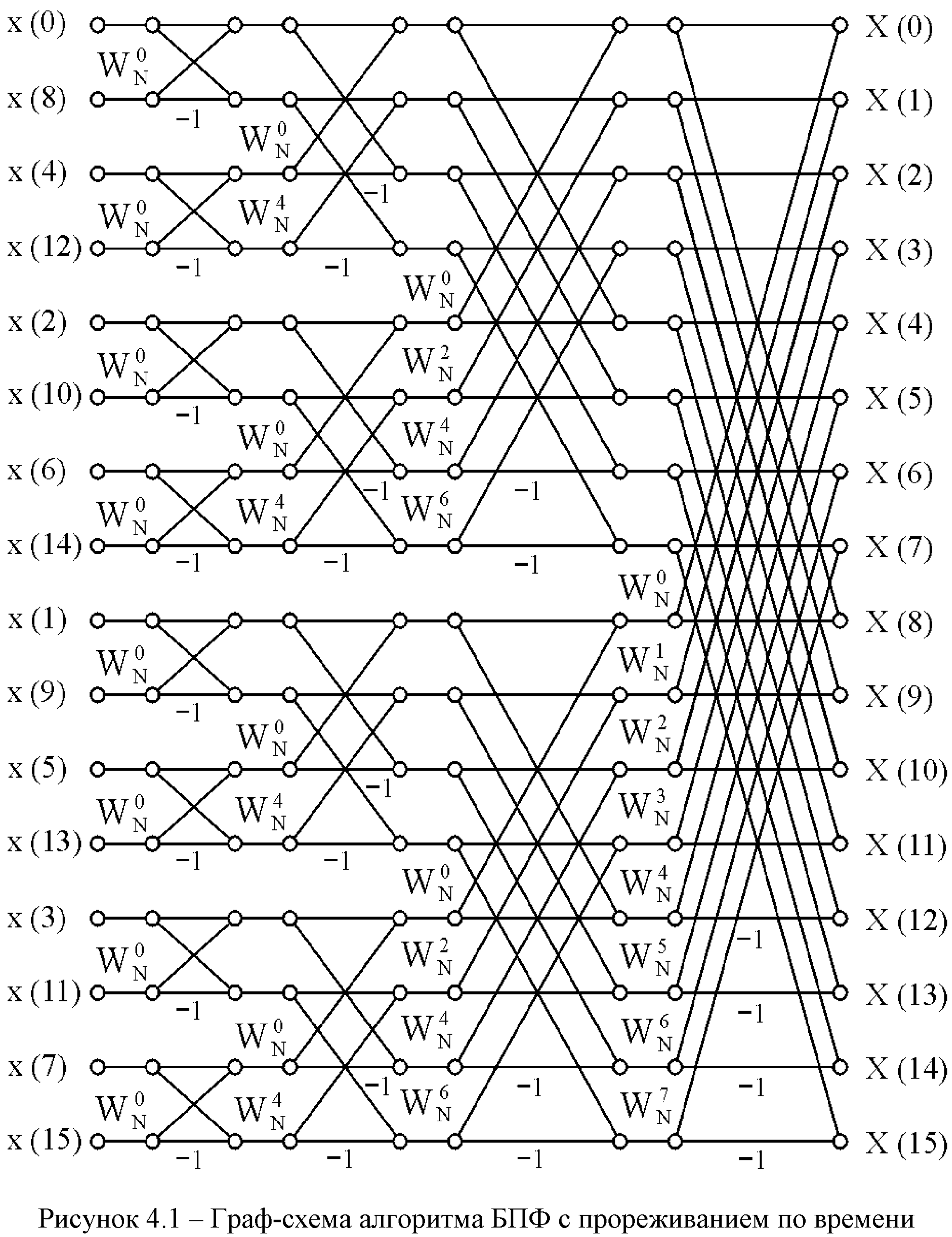

Рисунок 4.1 – Граф-схема алгоритма БПФ с прореживанием по времени

При переходе от комплексного представления выражений (4.33) к действительному получим:

$$
\begin{aligned}
&\operatorname{Re}\left[X_{m+1}(p)\right]=\operatorname{Re}\left[X_{m}(p)\right]+A;\\
&\operatorname{Im}\left[X_{m+1}(p)\right]=\operatorname{Im}\left[X_{m}(p)\right]+B;\\
&\operatorname{Re}\left[X_{m+1}(q)\right]=\operatorname{Re}\left[X_{m}(p)\right]-A;\\
&\operatorname{Im}\left[X_{m+1}(q)\right]=\operatorname{Im}\left[X_{m}(p)\right]-B,
\end{aligned}
\tag{4.39}
$$

где

$$
\begin{aligned}
&A=\operatorname{Re}\left[X_{m}(q)\right]\cdot\cos\left(\frac{2\pi\cdot r}{N}\right)-\operatorname{Im}\left[X_{m}(q)\right]\cdot\sin\left(\frac{2\pi\cdot r}{N}\right);\\
&B=\operatorname{Im}\left[X_{m}(q)\right]\cdot\cos\left(\frac{2\pi\cdot r}{N}\right)+\operatorname{Re}\left[X_{m}(q)\right]\cdot\sin\left(\frac{2\pi\cdot r}{N}\right).
\end{aligned}
\tag{4.40}
$$

Для реализации операции "бабочка" необходимо выполнить четыре операции действительного умножения и шесть операций действительного сложения, а трудоемкость всего алгоритма БПФ равна:

$3N\cdot\log_2 N$ операций сложения и
$2N\cdot\log_2 N$ операций умножения.

Вычисление функций sin и cos целесообразно осуществлять табличным способом. Так как современные ПЭВМ имеют достаточно большие размеры оперативной памяти, то для упрощения алгоритма выбора значений функции, соответствующих определенному аргументу, из таблицы, размер таблицы целесообразно выбрать равным максимальной размерности обрабатываемых массивов данных и в нее записать один период функции синуса.

Можно заметить, что на первой, второй и третьей ступенях БПФ реализация операции "бабочка" значительно упрощается.

Так для "бабочек" с индексом равным нулю (W_N^0) выражения (4.39) принимают вид (первая модификация):

$$
\begin{aligned}
&\operatorname{Re}\left[X_{m+1}(p)\right]=\operatorname{Re}\left[X_{m}(p)\right]+\operatorname{Re}\left[X_{m}(q)\right];\\
&\operatorname{Im}\left[X_{m+1}(p)\right]=\operatorname{Im}\left[X_{m}(p)\right]+\operatorname{Im}\left[X_{m}(q)\right];\\
&\operatorname{Re}\left[X_{m+1}(q)\right]=\operatorname{Re}\left[X_{m}(p)\right]-\operatorname{Re}\left[X_{m}(q)\right];\\
&\operatorname{Im}\left[X_{m+1}(q)\right]=\operatorname{Im}\left[X_{m}(p)\right]-\operatorname{Im}\left[X_{m}(q)\right],
\end{aligned}
\tag{4.41}
$$

для "бабочек" с индексом N/4 ($W_N^{\frac{N}{4}}$)(вторая модификация) - вид:

$$
\begin{aligned}
&\operatorname{Re}\left[X_{m+1}(p)\right]=\operatorname{Re}\left[X_{m}(p)\right]-\operatorname{Im}\left[X_{m}(q)\right];\\
&\operatorname{Im}\left[X_{m+1}(p)\right]=\operatorname{Im}\left[X_{m}(p)\right]+\operatorname{Re}\left[X_{m}(q)\right];\\
&\operatorname{Re}\left[X_{m+1}(q)\right]=\operatorname{Re}\left[X_{m}(p)\right]+\operatorname{Im}\left[X_{m}(q)\right];\\
&\operatorname{Im}\left[X_{m+1}(q)\right]=\operatorname{Im}\left[X_{m}(p)\right]-\operatorname{Re}\left[X_{m}(q)\right],
\end{aligned}
\tag{4.42}
$$

для "бабочек" с индексом N/8 ($W_N^{\frac{N}{8}}$)(третья модификация) - вид:

$$\begin{aligned} &\operatorname{Re}\left[X_{m+1}(p)\right] = \operatorname{Re}\left[X_m(p)\right] + C; \\ &\operatorname{Im}\left[X_{m+1}(p)\right] = \operatorname{Im}\left[X_m(p)\right] + D; \\ &\operatorname{Re}\left[X_{m+1}(q)\right] = \operatorname{Re}\left[X_m(p)\right] - C; \\ &\operatorname{Im}\left[X_{m+1}(q)\right] = \operatorname{Im}\left[X_m(p)\right] - D, \end{aligned} \tag{4.43}$$

где

$$\begin{aligned} C &= \left(\operatorname{Re}\left[X_m(q)\right] - \operatorname{Im}\left[X_m(q)\right]\right) \cdot \sqrt{2}; \\ D &= \left(\operatorname{Im}\left[X_m(q)\right] + \operatorname{Re}\left[X_m(q)\right]\right) \cdot \sqrt{2}, \end{aligned} \tag{4.44}$$

а для "бабочек" с индексом 3N/8 ($W_N^{\frac{3N}{8}}$)(четвертая модификация) - вид:

$$\begin{aligned} &\operatorname{Re}\left[X_{m+1}(p)\right] = \operatorname{Re}\left[X_m(p)\right] - E; \\ &\operatorname{Im}\left[X_{m+1}(p)\right] = \operatorname{Im}\left[X_m(p)\right] - F; \\ &\operatorname{Re}\left[X_{m+1}(q)\right] = \operatorname{Re}\left[X_m(p)\right] + E; \\ &\operatorname{Im}\left[X_{m+1}(q)\right] = \operatorname{Im}\left[X_m(p)\right] + F, \end{aligned} \tag{4.45}$$

где

$$\begin{aligned} E &= \left(\operatorname{Re}\left[X_m(q)\right] + \operatorname{Im}\left[X_m(q)\right]\right) \cdot \sqrt{2}; \\ F &= \left(\operatorname{Im}\left[X_m(q)\right] - \operatorname{Re}\left[X_m(q)\right]\right) \cdot \sqrt{2}. \end{aligned} \tag{4.46}$$

На первой ступени БПФ требуется выполнить $\frac{N}{2}$ "бабочек" первой модификации ($2N$ операций сложения), на второй ступени по $\frac{N}{4}$ "бабочек" первой и второй модификаций ($2N$ операций сложения), на третьей ступени по $\frac{N}{8}$ "бабочек" первой, второй, третьей и четвертой модификации ($\frac{5N}{2}$ операций сложения и $\frac{N}{2}$ операций умножения). На каждой следующей ступени имеется соответственно по $\frac{N}{16}$, $\frac{N}{32}$, ... , 2 "бабочек" каждой из модификаций.

Следовательно, для реализации модифицированного алгоритма БПФ требуется выполнить:

$N \cdot \left(2\log_2 N - 7\right) + 12$ операций умножения и
$3N \cdot \left(2\log_2 N - 1\right) + 4$ операций сложения.

Так как при решении задач обработки реальных сигналов приходиться иметь дело с действительными данными, то можно еще использовать особенности БПФ для действительных последовательностей. В этом случае исходная последовательность данных размерности N разбивается на две последовательности размерности $\frac{N}{2}$, одна из которых, состоящая из четных элементов исходной последовательности, интерпретируется как действительная часть, а вторая, состоящая из нечетных элементов исходной последовательности, интерпретируется как мнимая часть комплексной последовательности. В совокупности эти две последовательности представляют комплексную последовательность данных размерности $\frac{N}{2}$.

Далее для комплексной последовательности размерности $\frac{N}{2}$ выполняется БПФ. Но после его реализации требуется выполнить два действия по переходу к действительной последовательности размерности N. Первое действие, условно называемое разведением спектров, определяется выражениями:

$$\begin{aligned}
&\operatorname{Re}[X(k)] = p1; && \operatorname{Re}\left[X\left(\frac{N}{2}-k\right)\right] = p1;\\
&\operatorname{Im}[X(k)] = p2; && \operatorname{Im}\left[X\left(\frac{N}{2}-k\right)\right] = -p2;\\
&\operatorname{Re}[X(N-k)] = p3; && \operatorname{Re}\left[X\left(\frac{N}{2}+k\right)\right] = p3;\\
&\operatorname{Im}[X(N-k)] = p4; && \operatorname{Im}\left[X\left(\frac{N}{2}+k\right)\right] = -p4;
\end{aligned} \tag{4.47}$$

где

$$\begin{aligned}
p1 &= \frac{\operatorname{Re}[X(k)] + \operatorname{Re}\left[X\left(\frac{N}{2}-k\right)\right]}{2};\\
p2 &= \frac{\operatorname{Im}[X(k)] - \operatorname{Im}\left[X\left(\frac{N}{2}-k\right)\right]}{2};\\
p3 &= \frac{\operatorname{Im}[X(k)] + \operatorname{Im}\left[X\left(\frac{N}{2}-k\right)\right]}{2};\\
p4 &= \frac{\operatorname{Re}[X(k)] - \operatorname{Re}\left[X\left(\frac{N}{2}-k\right)\right]}{2},
\end{aligned} \tag{4.48}$$

а k изменяется от 0 до $\frac{N}{4}$.

Для реализации одной такой группы требуется выполнить четыре операции сложения, четыре операции умножения на 0.5, и две операции изменения знака. А для всей последовательности операций по разведению спектра - по N операций сложения и умножения на 0.5 и $\frac{N}{2}$ операций изменения знака.

Вторая действие, называемое объединением спектра, состоит из следующих операций:

$$\begin{aligned}
&\operatorname{Re}[X(k)] = \operatorname{Re}[X(k)] + A;\\
&\operatorname{Re}\left[X\left(\frac{N}{2}+k\right)\right] = \operatorname{Im}[X(k)] - A;\\
&\operatorname{Im}[X(k)] = \operatorname{Im}[X(k)] + B;\\
&\operatorname{Im}\left[X\left(\frac{N}{2}+k\right)\right] = \operatorname{Im}[X(k)] - B,
\end{aligned} \tag{4.49}$$

где

$$\begin{aligned}
A &= \operatorname{Re}\left[X\left(\frac{N}{2}+k\right)\right]\cdot\cos\left(\frac{2\pi}{N}\right) - \operatorname{Im}\left[X\left(\frac{N}{2}+k\right)\right]\cdot\sin\left(\frac{2\pi}{N}\right);\\
B &= \operatorname{Im}\left[X\left(\frac{N}{2}+k\right)\right]\cdot\cos\left(\frac{2\pi}{N}\right) + \operatorname{Re}\left[X\left(\frac{N}{2}+k\right)\right]\cdot\sin\left(\frac{2\pi}{N}\right),
\end{aligned} \tag{4.50}$$

а k изменяется от 0 до $\frac{N}{2}$.

Выражения (4.49) и (4.50), по сути дела, представляют собой обычную "бабочку", состоящую из четырех операций умножения и шести операций сложения.

А для всего действия по объединению спектра требуется выполнить $2N$ умножений и $3N$ сложений.

Таким образом, реализация обычного БПФ для действительной последовательности требует выполнения

$N\cdot\left(\log_2\left(\frac{N}{2}\right)+3\right)$ операций умножения и

$N\cdot\left(\frac{3}{2}\log_2\left(\frac{N}{2}\right)+4\right)$ операций сложения,

а модифицированного БПФ для действительной последовательности

$N\cdot\left(\log_2\left(\frac{N}{2}\right)-0.5\right)+12$ операций умножения и

$N\cdot\left(\frac{3}{2}\left(\log_2\left(\frac{N}{2}\right)+2.5\right)\right)+4$ операций сложения.

Во многих практических приложениях БПФ в недавнем прошлом реализовывался для данных, представленных в формате с фиксированной запятой с ограниченной разрядностью, как правило, равной шестнадцати. Такой подход объясняется требованием уменьшения времени выполнения этого алгоритма в системах реального времени. Соотношение шум/сигнал для такой реализации БПФ определяется величиной $4N \cdot 2^{-2b}$, где N - размерность массива входных данных, а b - число значащих двоичных разрядов.

Современные ПЭВМ оснащены мощными математическими сопроцессорами, которые осуществляют выполнение арифметических операций и вычисление основных функций в формате с плавающей запятой с высокой скоростью. Применение математического сопроцессора позволяет практически пренебречь шумом, возникающим в выходных данных из-за ограниченной разрядной сетки. Соотношение шум/сигнал для данных с плавающей запятой определяется выражением $2\log_2 N \cdot 2^{-2b}$. Так как для формата с плавающей запятой b обычно равно 24, то это соотношение имеет очень малое значение даже для больших N.

При применении математического сопроцессора ПЭВМ при оценке трудоемкости БПФ следует учитывать и операции пересылки данных из памяти в регистры сопроцессора и обратно, так как они сопоставимы по времени с арифметическими операциями сопроцессора. Для математического сопроцессора i486, если трудоемкость операции умножения принять за 1, то трудоемкость операции сложения будет примерно 0.625, операции пересылка память-регистр сопроцессора (П→R) - 0.1875, операции пересылка регистр сопроцессора память (R→П) - 0.4375, операции пересылка память-память (П→П) - 0.4375, операции инверсия знака 0.375. Для математического сопроцессора Pentium, если трудоемкость операции умножения принять за 1, то трудоемкость операции сложения будет примерно 1, операции пересылка память-регистр сопроцессора (П→R) - 0.333, операции пересылка регистр сопроцессора память (R→П) - 0.666, операции пересылка память-память (П→П) - 0.333.

Программная реализация на языке высокого уровня операции "бабочка" может иметь вид:

```
TSIN = mass[is];     // mass - массив в котором записан период синуса
TCOS = mass[ic];     // is - индекс для выбора из массива функции синуса
                     // ic - индекс для выбора из массива функции косинуса
ReA = ar_r[p];       // ar_r  - массив, в котором хранится действительная
                     // часть преобразуемой последовательности
ImA = ar_im[p];      // ar_im  - массив, в котором хранится мнимая
                     // часть преобразуемой последовательности
ReB = ar_r[q];
ImB = ar_im[q];
p1 = Reb * TCOS;     // p1,p2,p3 - переменные,  используемые
p2 = ImB * TSIN;     // в промежуточных вычислениях
p1 = p1 - p2;
p2 = Imb * TCOS;
p3 = ReB * TSIN;
p2 = p2 + p3;
ar_r[q] = ReA - p1;// в массивах ar_r и ar_im значения элементов
ar_r[p] = ReA + p1;        // изменяются от ступени к ступен
```

```
ar_im[q] = ImA - p2;
ar_im[p] = ImA + p2;
```

С учетом того, что сопроцессор поддерживает выполнение команд типа регистр-регистр и регистр-память, легко определить, что для выполнения этой программной реализации необходимо по 10 операций пересылки П→R и R→П и 6 операций пересылки П→П.

Проведя аналогичные рассуждения, увидим, что для реализации "бабочек" первой и второй модификации требуется по 4 операции пересылок П→R, R→П и П→П, а для "бабочек" третьей и четвертой модификации - по 6 операций пересылок П→R и R→П и 4 операции пересылки П→П, для операции разведения спектров - по 6 операций пересылок П→R и R→П и 8 операций пересылок П→П и для операции объединения спектра - по 10 операций пересылок П→R и R→П и 6 операций пересылок П→П.

С учетом этих данных, можно получить выражения для определения суммарного числа различных операций пересылок для полной реализации алгоритма БПФ.

Для реализации обычного алгоритма БПФ требуется выполнить

$5N \cdot \log_2 N$ пересылок типа П→R,

$5N \cdot \log_2 N$ пересылок типа R→П,

$5N \cdot \log_2 N$ пересылок типа П→П.

Для реализации модифицированного алгоритма БПФ требуется выполнить

$N \cdot (5\log_2 N - 11) + 20$ пересылок типа П→R,

$N \cdot (5\log_2 N - 11) + 20$ пересылок типа R→П,

$N \cdot (3\log_2 N - 4) + 8$ пересылок типа П→П.

Для реализации обычного алгоритма БПФ для действительных последовательностей требуется выполнить

$N \cdot \left(2.5\log_2\left(\frac{N}{2}\right) + 6.5\right)$ пересылок типа П→R,

$N \cdot \left(2.5\log_2\left(\frac{N}{2}\right) + 6.5\right)$ пересылок типа R→П,

$N \cdot \left(1.5\log_2\left(\frac{N}{2}\right) + 5\right)$ пересылок типа П→П.

Для реализации модифицированного алгоритма БПФ для действительных последовательностей требуется выполнить

$$N \cdot \left(2.5 \log_2 \left(\frac{N}{2} \right) + 1 \right) + 20$$ пересылок типа П→R,

$$N \cdot \left(2.5 \log_2 \left(\frac{N}{2} \right) + 1 \right) + 20$$ пересылок типа R→П,

$$N \cdot \left(1.5 \log_2 \left(\frac{N}{2} \right) + 3 \right) + 8$$ пересылок типа П→П.

Сопроцессор ПЭВМ имеет восемь внутренних регистров, доступ к которым осуществляется по принципу стека. Наличие этих регистров дает возможность в начале выполнения базовых операций БПФ ("бабочка", разведение спектров, сведение спектра) загрузить в регистры сопроцессора операнды, участвующие в операциях, а затем выгружать из регистров получаемые результаты в требуемые ячейки оперативной памяти. Такая организация вычислительного процесса БПФ позволяет значительно сократить число различных пересылок. Ниже представлена реализация операции "бабочка" на математическом сопроцессоре процессоров Intel.

```
    TSIN = mass[is];     // mass - массив в котором записан период синуса
    TCOS = mass[ic];     // is - индекс для выбора из массива функции синуса
                         // ic - индекс для выбора из массива функции косинуса
 asm  push si
 asm  mov  ax,word ptr i_p // i_p - значение индекса элемента X(p)
 asm  mov  dx,word ptr i_q // i_q - значение индекса элемента X(q)
 asm  shl  ax,1
 asm  shl  ax,1
 asm  shl  dx,1
 asm  shl  dx,1
 asm  les  bx,dword ptr [bp+10]  // загрузить адрес массива мнимой части
                                 //  исходных данных в регистр bx
 asm  mov  si,bx
 asm  add  bx,dx   //определить  адрес элемента массива мнимой
                   // части X(q)
 asm  FLD  dword ptr es:[bx]    // загрузить  значение элемента массива
                   // мнимой части X(q) (ImB) в вершину стека сопроцессора
 asm  FST  ST(2)        // создать копию ImB  в регистре ST(2) сопроцессора
 asm  les  bx,dword ptr [bp+6]  // загрузить адрес массива действительной
                                //  части исходных данных в регистр bx
 asm  mov  cx,bx
 asm  add  bx,dx        //определить  адрес элемента массива действительной
                        // части X(q)
 asm  FLD  dword ptr es:[bx] // загрузить  значение элемента массива
          // действительной части X(q) (ReB) в вершину стека сопроцессора
 asm  FST  ST(2) // Создать копию ReB в регистре ST(2) сопроцессора
 asm  FLD  dword ptr TSIN    //  загрузить TSIN в вершину стека сопроцессора
```

```
asm  FLD   dword ptr TCOS  // загрузить TCOS в вершину стека сопроцессора
asm  FMUL  ST(5),ST(0)          // ImB*TCOS , результатв ST(5)
asm  FMULP ST(4),ST(0)          // ReB*TCOS , результатв ST(4)
asm  FMUL  ST(2),ST(0)          // ImB*TSIN
asm  FMULP ST(1),ST(0)          // ReB*TSIN
asm  FADDP ST(3),ST(0)          // ImB*TCOS+ReB*TSIN (ii2), результатв ST(3)
asm  FSUBP ST(1),ST(0)          // ReB*TCOS-ImB*TSIN (ii1) , результатв ST(1)
asm  mov   bx,cx
asm  add   bx,ax        //определить  адрес элемента массива действительной
                        // части X(p)
asm  FLD   dword ptr es:[bx] // загрузить  значение элемента массива
          // действительной части X(p) (ReA) в вершину стека сопроцессора
asm  FLD   ST(0) // Создать копии ReA в регистрах ST(0) и ST(1)
asm  FSUB  ST(0),ST(2)          // ReA-ii1
asm  mov   bx,cx
asm  add   bx,dx        //определить  адрес элемента массива действительной
                        // части X(q)
asm  FSTP  dword ptr es:[bx] // выгрузить из вершины стека сопроцессора
        //вычисленное значение в элемент массива действительной части X(q)
asm  FADDP ST(1),ST(0)          // ReA+ii1
asm  mov   bx,cx
asm  add   bx,ax       //определить  адрес элемента массива действительной
                       // части X(p)
asm  FSTP  dword ptr es:[bx] // выгрузить из вершины стека сопроцессора
        //вычисленное значение в элемент массива действительной части X(p)
asm  mov   bx,si
asm  add   bx,ax       //определить  адрес элемента массива мнимой
                       // части X(p)
asm  FLD   dword ptr es:[bx] // загрузить  значение элемента массива
                     // мнимой части X(p) (ImA) в вершину стека сопроцессора
asm  FLD   ST(0) // создать копии ImA в регистрах ST(0) и ST(1)
asm  FSUB  ST(0),ST(2)          // ImA-ii2
asm  mov   bx,si
asm  add   bx,dx       //определить  адрес элемента массива мнимой
                       // части X(q)
asm  FSTP  dword ptr es:[bx] // выгрузить из вершины стека сопроцессора
               //вычисленное значение в элемент массива мнимой части X(q)
asm  FADDP ST(1),ST(0)          // ImA+ii2
asm  mov   bx,si
asm  add   bx,ax             //определить  адрес элемента массива мнимой
                             // части X(p)
asm  FSTP  dword ptr es:[bx] // выгрузить из вершины стека сопроцессора
               //вычисленное значение в элемент массива мнимой части X(p)
asm  pop   si
```

При таком подходе для вычисления одной операции "бабочка" требуется выполнить 6 пересылок типа П→R, 4 пересылки типа R→П и 2 пересылки типа П→П, для вычисления "бабочек" третьей и четвертой модификаций 5 пересылок типа П→R и 4 пересылки типа R→П, для операции разведения спектров 5 пересылок типа П→R и 4 пересылки R→П.

А в целом при такой организации вычислений для всего обычного алгоритма БПФ требуется выполнить

$3N \cdot \log_2 N$ пересылок типа П→R,
$2N \cdot \log_2 N$ пересылок типа R→П,
$N \cdot \log_2 N$ пересылок типа П→П,

для реализации модифицированного алгоритма БПФ -

$N \cdot (3\log_2 N - 3.5) + 6$ пересылок типа П→R,
$2N \cdot \log_2 N$ пересылок типа R→П,
$N \cdot (\log_2 N - 3)$ пересылок типа П→П.

для реализации обычного алгоритма БПФ для действительных последовательностей-

$N \cdot \left(1.5\log_2\left(\frac{N}{2}\right) + 4.25\right)$ пересылок типа П→R,
$N \cdot \left(\log_2\left(\frac{N}{2}\right) + 4\right)$ пересылок типа R→П,
$\frac{N}{2} \cdot \left(\log_2\left(\frac{N}{2}\right) - 3\right)$ пересылок типа П→П.

для реализации модифицированного алгоритма БПФ для действительных последовательностей -

$N \cdot \left(1.5\log_2\left(\frac{N}{2}\right) + 2.5\right)$ пересылок типа П→R,
$N \cdot \left(\log_2\left(\frac{N}{2}\right) + 4\right)$ пересылок типа R→П,
$\frac{N}{2} \cdot \left(\log_2\left(\frac{N}{2}\right) - 3\right)$ пересылок типа П→П.

Оценка трудоемкости рассмотренных алгоритмов показала, что использование стековых регистров сопроцессора для хранения промежуточных данных позволяет значительно снизить трудоемкость реализации алгоритма БПФ в целом. К примеру, для модифицированного алгоритма БПФ для действительных последовательностей при размере выборки в 1024 отсчета для математического сопроцессора i486 трудоемкость уменьшается примерно в 1.385 раз, а для математического сопроцессора Pentium - в 1.338 раз.