Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Томский государственный университет систем управления и радиоэлектроники»

Кафедра электронных приборов

Основы оптоинформатики

БЫСТРЫЙ АЛГОРИТМ ВЫЧИСЛЕНИЯ ДИСКРЕТНОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

Методические указания к лабораторной работе
для студентов направления «Фотоника и оптоинформатика»

2012

Слядников, Евгений Евгеньевич

Быстрый алгоритм вычисления дискретного преобразования Фурье = Основы оптоинформатики: методические указания к лабораторной работе для студентов направления «Фотоника и оптоинформатика» / Е.Е. Слядников; Министерство образования и науки Российской Федерации, Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования Томский государственный университет систем управления и радиоэлектроники, Кафедра электронных приборов. - Томск : ТУСУР, 2012. - 28 с.

Предназначено для студентов очной и заочной форм, обучающихся по направлению «Фотоника и оптоинформатика» по курсам «Основы оптоинформатики».

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Томский государственный университет систем управления и
радиоэлектроники»

Кафедра электронных приборов

УТВЕРЖДАЮ
Зав.кафедрой ЭП
_____С.М. Шандаров
«___» _____ 2012 г.

Основы оптоинформатики

БЫСТРЫЙ АЛГОРИТМ ВЫЧИСЛЕНИЯ ДИСКРЕТНОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

Методические указания к лабораторной работе
для студентов направления «Фотоника и оптоинформатика»

Разработчик

д-р физ.-мат. наук, проф. каф.ЭП
_____Е.Е. Слядников
«____»_____2012 г

2012

Содержание

1 Введение

Имеется несколько способов вычисления дискретного преобразования Фурье (ДПФ), такие как решение системы линейных уравнений, или *корреляционный* метод. Быстрое преобразование Фурье (БПФ) есть другой метод для вычисления ДПФ. Этот способ дает тот же результат, как и другие способы, но он невероятно более эффективен, зачастую он уменьшает время вычисления в *сотни* раз. Это все равно, что сравнивать полет в самолете с пешей прогулкой! Если бы не было БПФ, то многие технические решения, не могли бы быть реализованы. Хотя БПФ требует всего несколько дюжин линий в коде программы, он наиболее сложный алгоритм в ЦОС. Но не отчаивайтесь! Вы можете легко использовать опубликованные программы БПФ без полного понимания их внутренней работы.

2 Теоретическая часть

2.1 Общие требования

Реальное ДПФ, использующее комплексное ДПФ

Колей (J.W. Cooley) и Туки (J.W. Turkey) открыли БПФ двери в мир своей работой «Алгоритмы для машинного вычисления комплексных рядов Фурье» (*Математические вычисления*, том 19, страницы 297-301). В ретроспективном плане, много раньше эту технику открыли другие. Например, великий германский математик Карл Фридрих Гаусс (1777-1855) использовал этот метод более чем за столетие раньше. Его ранняя работа была сильно забыта потому, что отсутствовал инструмент для выполнения этого на практике, *цифровой компьютер*. Колей и Туки оказались героями потому, что они открыли БПФ в нужное время, в начале компьютерной революции.

БПФ базируется на комплексном ДПФ, более изощренной версии реального ДПФ. Эти преобразования названы по способу, которым представляются данные, то есть, используются для этого реальные числа или используются комплексные числа. Термин «комплексный» не означает, что это представление трудно или сложно, оно обозначает специфический тип используемой математики. Комплексная математика часто является трудной и сложной, но имя пришло не оттуда. Цель данной работы показать, как используется БПФ для вычисления реального ДПФ, без погружения в глубины математики.

Поскольку БПФ есть алгоритм для вычисления комплексного ДПФ, важно понимать, как получаются данные *реального* ДПФ на входе и на выходе из формата *комплексного* ДПФ. Рисунок 2.1 дает сравнение того, как хранятся данные при реальном ДПФ и при комплексном ДПФ. Реальное ДПФ преобразует N точек сигнала временной области в два набора из N/2 + 1 точек частотной области. Сигнал во временной области так и называется: *сигнал временной области*. Два сигнала частотной области называются *реальная часть* и *мнимая часть*, в них содержатся амплитуды косинусных волн и синусных волн, соответственно.

Сравнение реального и комплексного ДПФ

Реальное ДПФ берет N точек сигнала временной области и создает два набора из N/2+1 точек сигнала частотной области. Комплексное ДПФ берет два набора по N точек сигнала временной области и создает два набора из N точек сигала частотной области. Штриховкой показаны области величин общие для двух преобразований.

По сравнению с реальным ДПФ, комплексное ДПФ преобразует два набора по N точек сигнала временной области в два набора по N точек сигнала частотной области. Два набора сигнала временной области называются *реальной частью* и *мнимой частью*, так же как и сигналы частотной области. Не смотря на их названия, все величины в этих массивах так же обычные числа. [Если вы знакомы с комплексными числами, то j не включено в массивы этих величин, они часть *математического* способа; вспомните, что оператор Im( ) возвращает реальные числа].

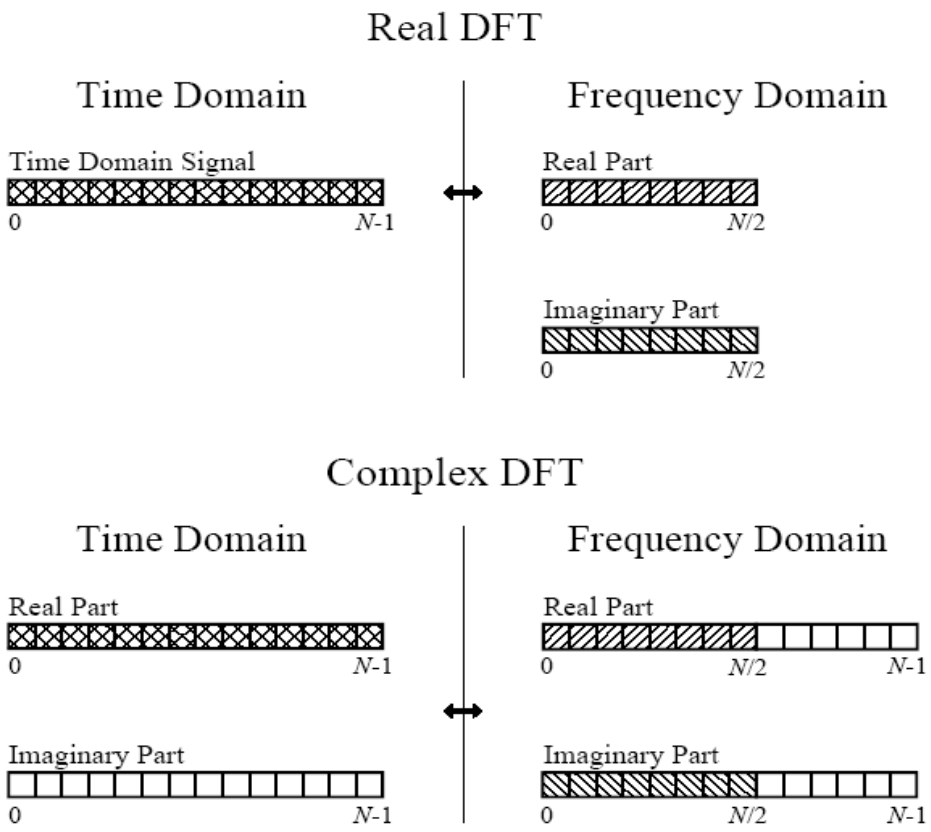

Рисунок 2.1 - Сравнение того, как хранятся данные при реальном ДПФ и при комплексном ДПФ

Предположим, что вы имеете N точек сигнала и вам необходимо вычислить *реальное* ДПФ с помощью *комплексного* ДПФ (как в случае использования алгоритма БПФ). Первое, помещаете N точек сигнала в реальную часть временной области комплексного БПФ и затем устанавливаете все отсчеты мнимой части в *нули*. В результате вычисления

комплексного ДПФ в реальной и мнимой частях частотной области получается N точек. Отсчеты от 0 до N/2 этих сигналов соответствуют спектру реального ДПФ.

Частотная область ДПФ периодическая, и в нее включены негативные частоты. Выбор сигнального периода произволен; он может быть выбран между -1,0 и 0, между -0,5 и 0,5, между 0 и 1 или между каким либо другим единичным интервалом, определяемым частотой дискретизации. Частотный спектр комплексного ДПФ включает негативные частоты в интервале 0-1,0. Другими словами, один полный период от отсчета 0 до отсчета N-1 соответствует интервалу от 0 до 1 частоты дискретизации. Положительные частоты находятся между отсчетами 0 и N/2, что соответствует интервалу 0 – 0.5 частоты дискретизации. Отсчеты между N/2+1 и N-1 содержат величины негативных частот, которые обычно игнорируются**.**

Для вычисления *реального инверсного ДПФ* используют *комплексное инверсное ДПФ* с небольшими изменениями. Это вызвано тем, что вам необходимо убедится, что негативные частоты загружены в правильном формате. Помните, точки от 0 до N/2 в комплексном ДПФ те же самые, что и в реальном ДПФ для реальной части и мнимой части. Для реальной части точка N/2+1 та же самая, что и точка N/2-1, точка N/2+2 та же, что и точка N/2-2 и так далее. Это продолжается до точки N-1, которая то же, что и точка 1. Тот же подход используется и для мнимой части, за исключением изменения знака. Так точка N/2+1 есть негативная точка N/2-1, точка N/2+2 есть негативная точка N/2-2 и так далее. Заметим, что точки 0 и N/2 не имеют пар при этом подходе. На практике вы загружаете спектр реального ДПФ в отсчеты от 0 до N/2 массива комплексного ДПФ, и затем используете подпрограмму для генерации негативных частот от N/1+1 до N-1. Таблица 5.1 показывает эту программу.

Таблица 5.1 – Программа

```
6000     'ГЕНЕРАЦИЯ НЕГАТИВНЫХ ЧАСТОТ
6010   'Эта подпрограмма создает комплексную частотную область из
реальной частотной области
6020   'На входе этой подпрограммы N% содержит число точек в сигнале, и
6030   'REX[ ] and IMX[ ] содержат реальную частотную область с отсчетами
от 0 до N%/2.
6040 'На выходе в REX[ ] и IMX[ ] комплексная частотная область с
отсчетами от 0 до N%-1.
6050 '
6060 FOR K% = (N%/2+1) TO (N%-1)
6070    REX[K%] = REX[N%-K%]
6080    IMX[K%] = -IMX[N%-K%]
6090 NEXT K%
6100 '
6110 RETURN
```

Что бы убедится, что симметрия правильная, после взятия инверсного БПФ посмотрите на мнимую часть временной области. Она должна состоять из одних нулей, если все правильно (за исключением миллионных долей шума за счет точности вычисления)

Как работает БПФ

БПФ сложный алгоритм, и его детали обычно оставляют для тех, кто специализируется в этих вещах. Этот раздел описывает основные операции БПФ, но обходит стороной ключевой вопрос, использование *комплексных чисел*. Если вы знакомы с основами математики комплексных чисел, вы можете между строк прочитать истинную природу алгоритма. Не переживайте, если детали ускользнут от вас, немногие ученые и инженеры, которые используют БПФ, могут написать безукоризненную программу.

В комплексной записи временная и частотная области содержат *один сигнал* из N *комплексных точек*. Каждая из этих точек состоит из двух чисел, реальной части и мнимой части. Например, когда мы говорим о комплексном числе X[42], это означает комбинацию из ReX[42] и ImX[42]. Другими словами, каждая комплексная переменная содержит два числа. Когда две комплексные переменные перемножаются, четыре индивидуальных компонента должны быть скомбинированы в две компоненты полученного числа. Этот раздел «*Как работает БПФ*» использует жаргон комплексных чисел. То есть, термины: *сигнал*, *точка*, *отсчет* и *величина* означают *комбинацию* из реальной части и мнимой части.

БПФ работает путем разложения N точек сигнала временной области на N сигналов временной области, каждый из которых состоит из одной точки. Следующий шаг заключается в вычислении N частотных спектров, соответствующим этим N сигналам. Наконец, N спектров синтезируются в один частотный спектр.

Рисунок 5.2 показывает пример разложения сигнала временной области, используемый в БПФ. N-точечный сигнал раскладывается на N сигналов, каждый из которых содержит одну точку. На каждом этапе проводится чередующееся разложение, разделяя четные и нечетные номера отсчетов.

В этом примере 16-ти точечный сигнал раскладывается за четыре отдельных шага. На первом шаге 16-ти точечный сигнал разделяется на два 8-ми точечных сигнала. На втором шаге – на четыре сигнала из 4 точек. Эта процедура продолжается пока сигнал не разложится на отдельные точки. Чередующееся разложение используется каждый раз, когда сигнал разделяется на два, то есть сигнал разделяется на четные и нечетные номера отсчетов. Наилучший способ понять эту процедуру - изучать рисунок 12-2 до тех пор, пока вы не усвоите этот шаблон. Требуется Log_2N шагов для разложения сигнала, то есть 16-точечный сигал (2^4) потребует четыре шага, 512-точечный сигнал (2^7) – 7 шагов, сигнал из 4096 точек (2^{12}) – 12 шагов и так далее. Запомните эту величину, Log_2N, она будет встречаться много раз в этой главе.

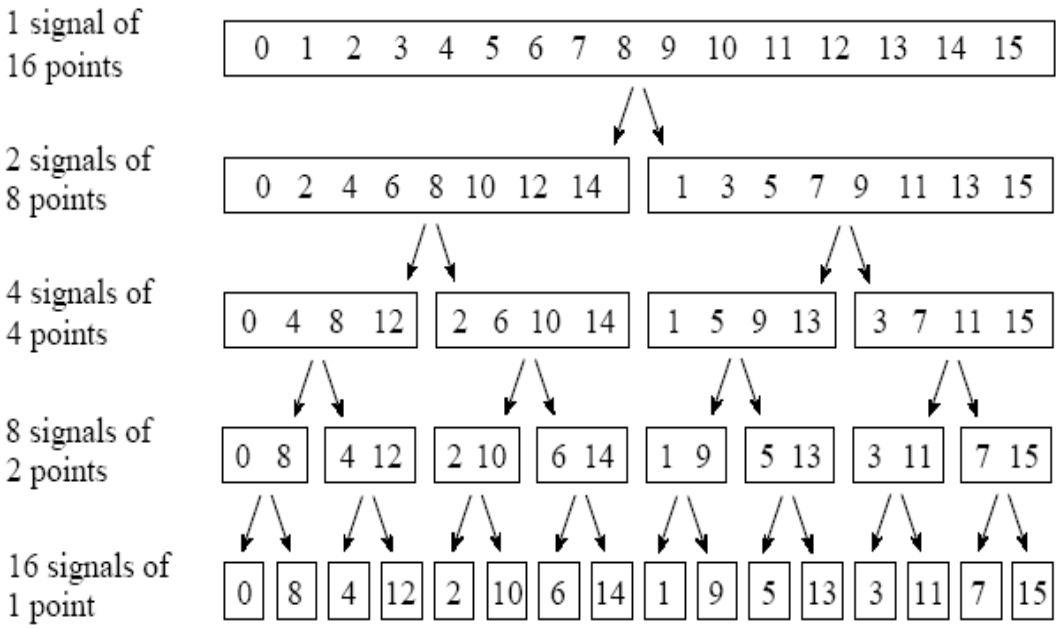

Рисунок 5.2 - БПФ разложение

Теперь, когда вы поняли структуру разложения, она может быть выполнена очень просто. Разложение есть ничего более, как *переорганизация* отсчетов. Рисунок 5.3 показывает шаблон переорганизации

Нормальный порядок отсчетов			Отсчеты после битового реверсирования	
Десятичный	Двоичный		Десятичный	Двоичный
0	0000		0	0000
1	0001		8	1000
2	0010		4	0100
3	0011		12	1100
4	0100		2	0010
5	0101		10	1010
6	0110	⇨	6	0110
7	0111		14	1110
8	1000		1	0001
9	1001		9	1001
10	1010		5	0101
11	1011		13	1101
12	1100		3	0011
13	1101		11	1011
14	1110		7	0111
15	1111		15	1111

Рисунок 5.3 - Сортировка побитового реверсирования при БПФ

Слева исходный порядок отсчетов. Разложение временной области может быть выполнено сортировкой отсчетов способом битового реверсирования.

Справа новый порядок отсчетов. Идея заключается в том, что бинарные числа *реверсируются* друг с другом. Например, отсчет 3 (0011) изменяется на отсчет 12 (1100). Подобным образом отсчет номер 14 (1110) меняется с отсчетом номер 7 (0111) и т.д. При БПФ разложение временной области обычно выполняется алгоритмом сортировки битового реверсирования (bit reversal sorting). Новый порядок N отсчетов временной области получается пересчетом двоичного номера путем переброски битов справа налево (как в правой колонке рисунка 5.3).

Следующий шаг алгоритма БПФ заключается в нахождении частотного спектра одной точки сигнала временной области. Нет ничего легче, частотный спектр 1-точечного сигнала равен *самому* сигналу. Это означает, что *ничего* не требуется делать на этом шаге. Хотя ничего и не требуется делать, однако не забывайте, что каждая точка теперь частотный спектр, а не сигнал временной области.

Последний шаг БПФ есть комбинация N частотных спектров в порядке обратном разложению временной области. В этом месте алгоритм сложен. К несчастью, тут побитовое реверсирование не годится, и мы должны идти назад по шагам. На первом шаге 16 частотных спектров (по 1 точке каждый) синтезируются в 8 частотных спектров (по 2 точки каждый). На втором шаге 8 частотных спектров (по 2 точки каждый) синтезируются в 4 частотных спектра (по 4 точки каждый) и т.д. На последнем шаге получается выходной спектр БПФ из 16 точек.

Рисунок 5.4 показывает, как два частотных спектра, состоящих из 4 точек, комбинируются в один частотный спектр из 8 точек. Когда сигнал временной области разбавляется нулями, частотная область удваивается. Если сигнал временной области был сдвинут при разбавлении, то спектр дополнительно умножается на синусоиду.

Этот синтез должен *уничтожить* чередующееся разложение, выполненное во временной области. Другими словами, операции над частотной областью должны соответствовать во временной области процедуре *комбинации* двух сигналов из 4 точек с помощью чередования. Рассмотрим два сигнала временной области, abcd и efgh. Сигнал временной области из 8 точек может быть сформирован за два шага: разбавить каждый 4-точечный сигнал нулями, сделав их 8-точечными, и затем сложить их. То есть abcd становится a0b0c0d0, а efgh становится 0e0f0g0h. Сложив эти два 8-точечных сигнала, мы получим aebfcgdh. Как показано на рисунке 5.4, разбавление сигнала временной области нулями соответствует *удвоению* частотного спектра. Поэтому частотный спектр БПФ получается путем удвоения спектров и, затем, сложения удвоенных спектров.

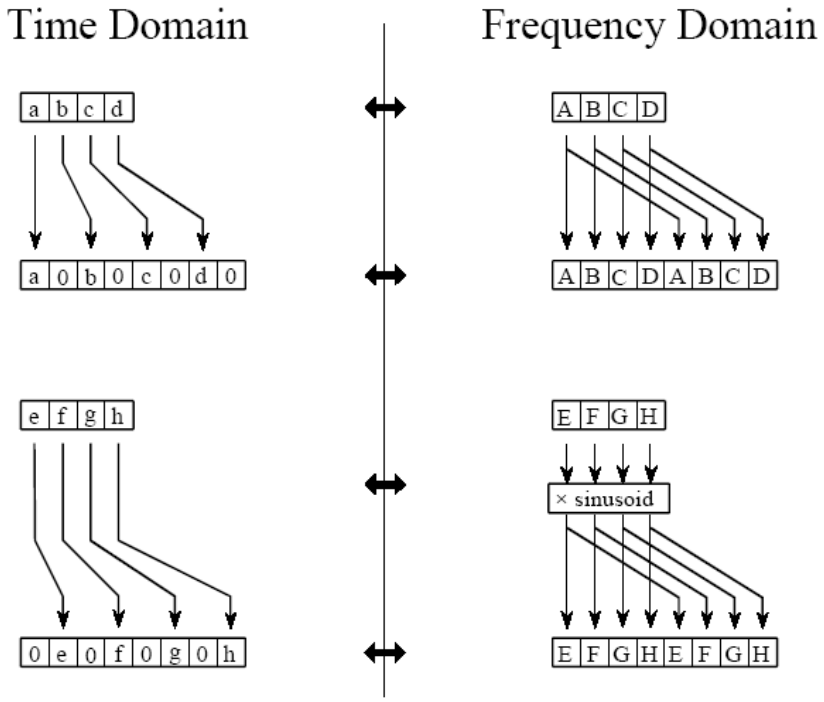

Рисунок 5.4 - Синтез БПФ

Что бы осуществить правильное сложение, необходимо учитывать, что сигналы временной области немного различно разбавляются нулями. В одном сигнале нечетные точки равны нулю, в другом – четные. Другими словами один из сигналов временной области (0e0f0g0h на рисунке 5.4) сдвинут на один отсчет вправо. Сдвиг во временной области соответствует умножению спектра на *синусоиду*. Видя это, вспомните, что сдвиг во временной области эквивалентен свертки сигнала со сдвинутой дельта функцией. А это есть перемножение спектра сигнала на спектр сдвинутой дельта функции. Спектр сдвинутой дельта функции есть синусоида (смотри рисунок 5.2).

Рисунок 5.5 показывает диаграмму комбинации двух 4-точечных спектров в один 8-точечный спектр. Оператор ×S означает, что сигнал умножается на синусоиду с соответствующей частотой.

Чтобы упростить ситуацию еще больше, рассмотрим рисунок 5.5 как повторение рисунка 5.6. Это базовый элемент вычисления в БПФ. Он преобразует две комплексные точки в две другие комплексные точки.

Эта простая диаграмма называется бабочкой, так как похожа на крылья. Бабочка основной элемент вычисления в БПФ, он преобразует две комплексные точки в две другие комплексные точки.

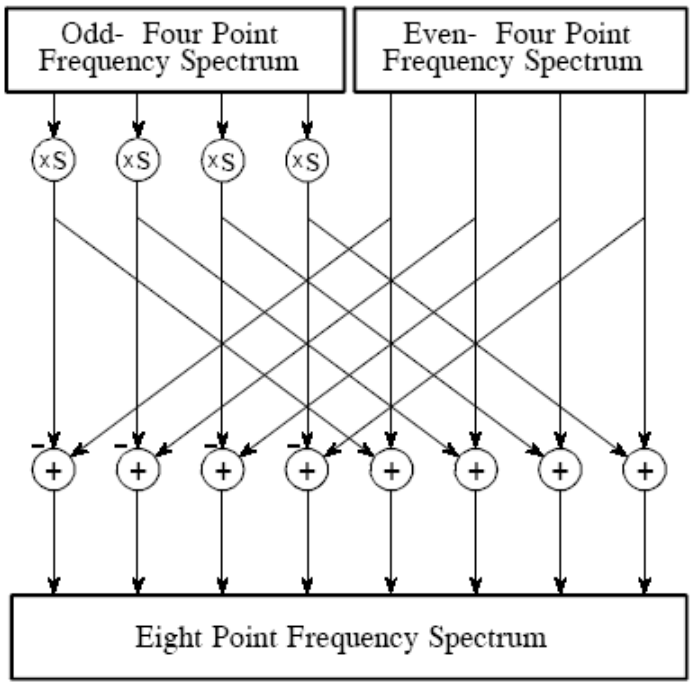

Рисунок 5.5 - Диаграмма синтеза БПФ

.

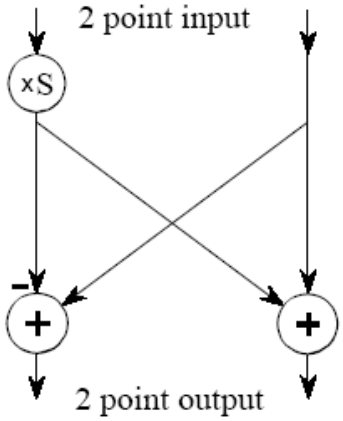

Рисунок 5.6 - Бабочка БПФ

Рисунок 5.7 показывает структуру БПФ целиком**.** Она базируется на трех шагах. (1) разложение N-точечного сигнала временной области на N сигналов из одной точки. (2) Нахождение каждого спектра для этих сигналов (ничего не требуется). (3) Синтез N спектров в один частотный спектр

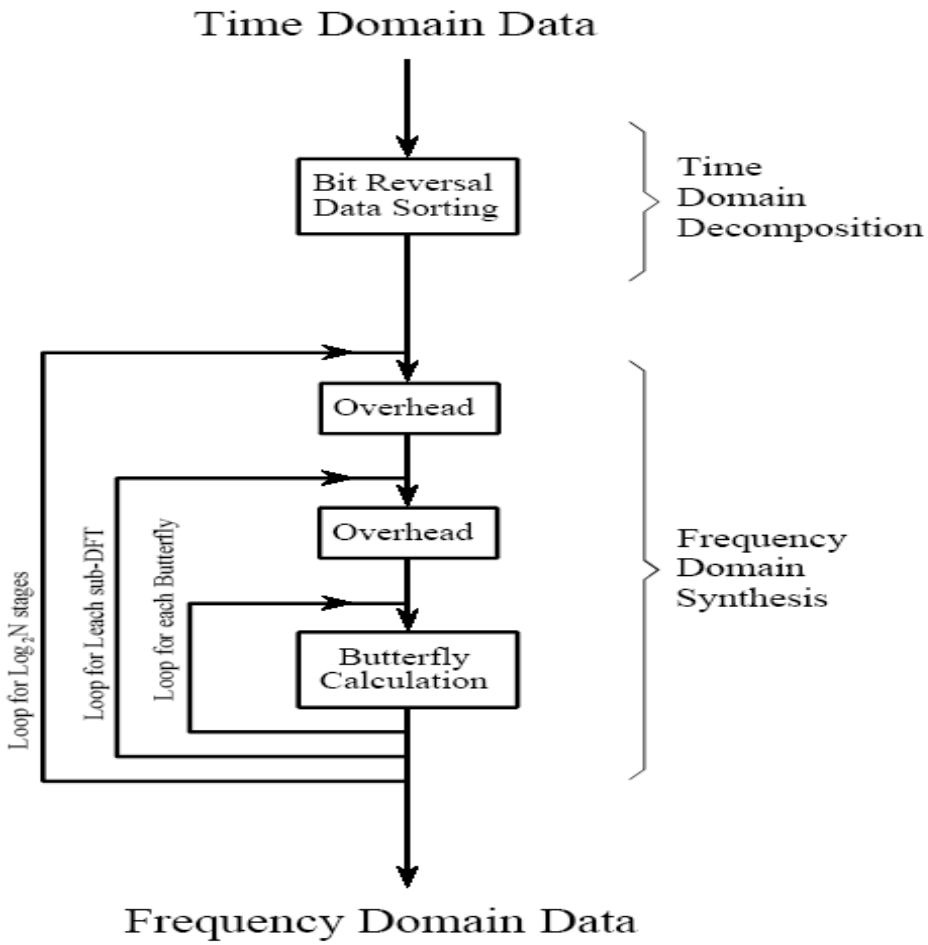

Рисунок 5.7 - Диаграмма БПФ

Разложение временной области выполняется сортировкой с помощью битового реверсирования. Преобразование разложенных данных в частотную область не требует *никаких* операций, и поэтому оно не показано.

Синтез частотной области требует три цикла. Внешний цикл включает Log_2N шагов (т.е. каждый уровень на рисунке 5.2, начиная снизу и двигаясь вверх). Средний цикл проходит через каждый отдельный частотный спектр в шаге (т.е. через каждый квадрат на любом уровне рисунка 5.2). Внутренний цикл использует бабочку для вычисления точек каждого частотного спектра (т.е. петляя через все отсчеты внутри любого квадрата на рисунке 5.2). Прямоугольники заголовков (overhead box) определяют начало и конец индексации для циклов, а также вычисляют синусоиды, необходимые для бабочки. Теперь мы подошли к сердцу этой главы – реальной программе БПФ.

Программа БПФ

Реальное ДПФ может быть вычислено с помощью корреляции сигнала временной области с синусными и косинусными волнами. Таблица 5.2 показывает программу вычисления *комплексного* ДПФ тем же самым методом. При подробном сравнении эта программ улучшается БПФ.

Таблица 5.2 - Программа вычисления комплексного ДПФ через корреляцию

```
5000 'КОМПЛЕКСНОЕ ДПФ ЧЕРЕЗ КОРРЕЛЯЦИЮ
5010 Вначале N% содержит число точек ДПФ и
5020 'XR[ ] и XI[ ] содержат реальную и мнимую части сигнала временной области,
5030 'REX[ ] и IMX[ ] содержат данные частотной области.
5040 'Все сигналы меняются от 0 до N%-1.
5050 '
5060 PI = 3.14159265                          'Ввод константы
5070 '
5080 FOR K% = 0 TO N%-1                       'Обнуление REX[ ] и IMX[ ], так как они
5090    REX[K%] = 0                           'используются для аккумуляции
5100    IMX[K%] = 0
5110 NEXT K%
5120 '
5130 FOR K% = 0 TO N%-1                       'Цикл для каждой величины частотной области
5140    FOR I% = 0 TO N%-1                    'Корреляция с комплексной синусоидой SR & SI
5150 '
5160       SR = COS(2*PI*K%*I%/N%)            'Вычисление комплексной синусоиды
5170       SI = -SIN(2*PI*K%*I%/N%)
5180       REX[K%] = REX[K%] + XR[I%]*SR - XI[I%]*SI
5190       IMX[K%] = IMX[K%] + XR[I%]*SI + XI[I%]*SR
5200 '
5210    NEXT I%
5220 NEXT K%
5230 '
5240 RETURN
```

Таблицы 5.3 и 5.4 показывают различные БПФ программы, одна на Фортране и одна на Бейсике.

Таблица 5.3 - Программа Быстрого Преобразования Фурье на ФОРТРАНЕ

```
      SUBROUTINE   FFT(X,M)
      COMPLEX   X(4096),U,S,T
      PI=3.14159265
      N=2**M
      DO 20 L=1,M
      LE=2**(M+1-L)
      LE2=LE/2
      U=(1.0,0.0)
      S=CMPLX(COS(PI/FLOAT(LE2)),-SIN(PI/FLOAT(LE2)))
      DO 20 J=1,LE2
      DO 10 I=J,N,LE
      IP=I+LE2
      T=X(I)+X(IP)
      X(IP)=(X(I)-X(IP))*U
10       X(I)=T
20       U=U*S
      ND2=N/2
      NM1=N-1
      J=1
      DO 50 I=1,NM1
      IF(I.GE.J) GO TO 30
      T=X(J)
      X(J)=X(I)
      X(I)=T
30       K=ND2
40       IF(K.GE.J) GO TO 50
      J=J-K
      K=K/2
      GO TO 40
50       J=J+K
      RETURN
      END
```

Таблица 5.4 - Программа быстрого преобразования Фурье на Бейсике

```
1000 'БЫСТРОЕ ПРЕОРАЗОВАНИЕ ФУРЬЕ
1010 'На входе N% содержит число точек ДПФ REX[ ] и
1020 'IMX[ ] содержат реальную и мнимую часть входа. На выходе
1030 'REX[ ] и IMX[ ] содержат выход ДПФ. Все сигналы меняются от 0 до
N%-1.
1040 '
1050 PI = 3.14159265                                    'Установка констант
1060 NM1% = N%-1
```

```
1070 ND2% = N%/2
1080 M% = CINT(LOG(N%)/LOG(2))
1090 J% = ND2%
1100 '
1110 FOR I% = 1 TO N%-2                    'Сортировка битового
реверсирования
1120    IF I% >= J% THEN GOTO 1190
1130    TR = REX[J%]
1140    TI = IMX[J%]
1150    REX[J%] = REX[I%]
1160    IMX[J%] = IMX[I%]
1170    REX[I%] = TR
1180    IMX[I%] = TI
1190    K% = ND2%
1200    IF K% > J% THEN GOTO 1240
1210    J% = J%-K%
1220    K% = K%/2
1230    GOTO 1200
1240    J% = J%+K%
1250 NEXT I%
1260 '
1270 FOR L% = 1 TO M%                      'Цикл для каждой ступени
1280    LE% = CINT(2^L%)
1290    LE2% = LE%/2
1300    UR = 1
1310    UI = 0
1320    SR = COS(PI/LE2%)                  'Вычисление sine & cosine
величин
1330    SI = -SIN(PI/LE2%)
1340    FOR J% = 1 TO LE2%                 'Цикл для каждой процедуры
ДПФ
1350      JM1% = J%-1
1360      FOR I% = JM1% TO NM1% STEP LE%   'Цикл для каждой бабочки
1370        IP% = I%+LE2%
1380        TR = REX[IP%]*UR - IMX[IP%]*UI  'Вычисление бабочки
1390        TI = REX[IP%]*UI + IMX[IP%]*UR
1400        REX[IP%] = REX[I%]-TR
1410        IMX[IP%] = IMX[I%]-TI
1420        REX[I%] = REX[I%]+TR
1430        IMX[I%] = IMX[I%]+TI
1440      NEXT I%
1450      TR = UR
1460      UR = TR*SR - UI*SI
1470      UI = TR*SI + UI*SR
1480    NEXT J%
1490 NEXT L%
1500 '
1510 RETURN
```

Сначала мы рассмотрим программу на Бейсике в таблице 5.4. Эта подпрограмма обеспечивает точно такой же выход, как корреляционный метод в таблице 5.2. за исключением того, что она делает это *много быстрее*. Блок диаграмма на рисунке 5.7 может быть использована для идентификации различных секций этой программы. Данные пропускаются через эту подпрограмму БПФ в виде массивов REX[ ] и IMX[ ], отсчеты в которых меняются от 0 до N-1. Возвращенные из подпрограммы, REX[ ] и IMX[ ] перезаписываются как данные частотной области. Это есть другой способ высокой оптимизации БПФ, тот же самый массив используется для хранения входных, промежуточных и выходных данных. Это эффективное использование памяти очень важно при создании оборудования для вычисления БПФ. Термин вычисление по месту (in-place computation) применяется для описания такого использования памяти.

В то время как все программы производят один и тот же результат, имеются тонкие вариации в программировании, с которыми вы должны быть настороже. Несколько из этих отличий иллюстрируются программой на фортране в таблице 5.3. Данные поступают на эту подпрограмму в виде переменных X( ) и M. Целое число M есть логарифм по основанию 2 от длины ДПФ, т.е. M=8 для 256-точечного ДПФ, M=12 для 4096-точечного и т.д. Комплексный массив X( ) содержит данные временной области на входе ДПФ. После возвращения его подпрограммой, в X( ) перезаписываются данные частотной области. Возьмите на заметку, что эта подпрограмма требует изменения входных и выходных сигналов от X(1) до X(N) вместо обычного от X(0) до X(N-1).

Эта программа использует алгоритм, называемый децимация по частоте, в то время как ранее описанный алгоритм называется децимацией по времени. В алгоритме с децимацией по частоте сортировка с помощью битового реверсирования выполняется *после* трех вложенных циклов. Имеются так же программы БПФ, которые полностью удаляют сортировку битового реверсирования. Ни одна из этих вариаций не улучшает значительно выполнение БПФ, и вы не должны заботиться о том, которая из них используется.

Важное различие между БПФ алгоритмами касается того, как данные поступают и покидают подпрограммы. В программе на БЕЙСИКЕ данные поступают и покидают подпрограмму в виде массивов REX[ ] и IMX[ ] с отсчетами меняющимися от 0 до N-1. В программе на ФОРТРАНЕ данные поступают в массив X( ) с отсчетами от 1 до N. Поскольку это массив комплексных переменных, то каждый отсчет в X( ) состоит из двух чисел, реальной и мнимой части. Длина ДПФ должна так же подходить для этой подпрограммы. В программе на БЕЙСИКЕ переменная N% используется для этой цели. Для сравнения, программа на ФОРТРАНЕ использует переменную M, которая выбирается равной Log2N. Например, M будет равна 8 для 256-точечного ДПФ, 12 для 4096-точечного ДПФ и т.д. Смысл в том, что программист, который пишет подпрограмму БПФ, имеет много опций для связи с основной программой. Массив, который меняется от 1 до N такой,

как в программе на ФОРТРАНЕ, особенно надоедлив. Большинство литературы по ЦОС объясняет алгоритмы, предполагая, что отсчеты в массиве меняются от 0 до N-1. Например, если массив меняется от 1 до N, симметрия в частотной области будет относительно точек 1 и N/2+1, вместо точек 0 и N/2.

Использование комплексного ДПФ для вычисления реального ДПФ имеет еще одно интересное преимущество. Комплексное ДПФ более симметрично между временной и частотной областями, чем реальное ДПФ. То есть двойственность сильнее. Среди всего прочего, это означает, что инверсное ДПФ ближе к прямому ДПФ. Действительно, самый легкий способ вычисления *инверсного ДПФ* есть вычисление *прямого БПФ,* и затем регулирование данных. Таблица 5.5 показывает подпрограмму для вычисления инверсного ДПФ таким способом.

Таблица 5.5 - Подпрограмма для вычисления инверсного ДПФ

```
2000 'ПОДПРОГРАММА ИНВЕРСНОГО БПФ
2010 'На входе N% содержит число точек инверсного ДПФ, REX[ ] и
2020 'IMX[ ] содержит реальную и мнимую часть комплексной частотной
области.
2030 'На выходе REX[ ] и IMX[ ] содержат комплексную временную область.
2040 'Все сигналы меняются от 0 до N%-1.
2050 '
2060 FOR K% = 0 TO N%-1                          'Изменение знака
IMX[ ]
2070    IMX[K%] = -IMX[K%]
2080 NEXT K%
2090 '
2100 GOSUB 1000                                  'Вычисление прямого
БПФ (Table 12-3)
2110 '
2120 FOR I% = 0 TO N%-1                          'Деление временной
области на N% и
2130    REX[I%] = REX[I%]/N%                     'изменение знака IMX[
]
2140    IMX[I%] = -IMX[I%]/N%
2150 NEXT I%
2160 '
2170 RETURN
```

Предположим, вы копируете один из этих алгоритмов БПФ в вашу компьютерную программу и запускаете на исполнение. Как вы узнаете, что она работает правильно? Два способа обычно используются для проверки. Первый, начинают с некоторого произвольного сигнала временной области,

такой как генератор случайных чисел, и пропускают его через БПФ. Затем, пропускают полученный частотный спектр через инверсное БПФ, и результат сравнивают с оригинальным сигналом. Они должны быть *идентичны*, за исключением шума округления (несколько миллионных долей сигнала).

Второй способ контроля правильной работы заключается в проверке сигнала на правильную *симметрию*. Когда мнимая часть сигнала временной области состоит из всех нулей (нормальный случай) частотная область комплексного ДПФ будет симметрична относительно отсчетов 0 и N/2, как описывалось ранее.

Точно также, когда правильная симметрия присутствует в частотной области, инверсное ДПФ будет давать во временной области сигнал, мнимая часть которого будет состоять из всех нулей (плюс шум округления). Эти отладочные приемы являются неотъемлемой частью использования БПФ, хорошо ознакомьтесь с ними.

Скорость и точность вычислений

Когда ДПФ (DFT) вычисляется с помощью корреляции (как в таблице 5.2), программа использует два вложенных цикла, каждый из N шагов. Это означает, что общее число операций пропорционально N *взятому* N раз. Время вычисления программы будет равно:

$$\text{ExecutionTime} = k_{DFT} N^2 \qquad (5.1)$$

Время, требуемое для вычисления ДПФ с помощью корреляции пропорционально длине ДПФ в квадрате

Здесь N число точек в ДПФ, и k_{DFT} постоянная величина. Если синусы и косинусы вычисляются *внутри* циклов, то k_{DFT} равна 25 миллисекундам для Pentium 100 МГц. Если вы *заранее* вычисляете синусы и косинусы и храните их в справочной таблице, то k_{DFT} уменьшается до 7 миллисекунд. Например, 1024-точечное ДПФ потребует 25 секунд, или 25 миллисекунд на точку. Вот как медленно!

Используя тот же подход, мы можем вычислить время вычисления для БПФ (FFT). Время, необходимое для битового реверсирования незначительно. В каждом из Log_2N шагов имеется N/2 вычислений бабочек. Это означает, что время вычисления программы будет равно:

$$\text{ExecutionTime} = k_{FFT} N \log_2 N \qquad (5.2)$$

Время, требуемое для вычисления БПФ пропорционально N умноженное на логарифм N.

Величина k_{FFT} около 10 микросекунд для 100 МГц Pentium. 1024-точечное БПФ потребует около 70 миллисекунд для вычисления, или 70

микросекунд на точку. Это более чем в 300 раз быстрее, чем для вычисления ДПФ с помощью корреляции!

Не только величина $NLog_2N$ меньше, чем N_2, но и увеличивается она медленнее с ростом N. Например, 32-точечное БПФ примерно в *десять* раз быстрее, чем корреляционный метод, а для 4096-точечнго БПФ – в *тысячу* раз быстрее. Для небольших величин N (скажем, 32 или 128) БПФ важно, а для больших величин N (1024 и выше) БПФ критично. Рисунок 5.8 показывает сравнение времени вычисления двух алгоритмов в графической форме. Алгоритм *корреляционного* метода представлен в таблице 5.2. Этот метод может быть ускорен предварительным вычислением синусов и косинусов и запоминанием их в справочной таблице (look-up-table – LUT). БПФ [FFT] (таблица 5.3) значительно более быстрый алгоритм, когда ДПФ длиннее 16 точек. Время показано для Pentium 100 МГц.

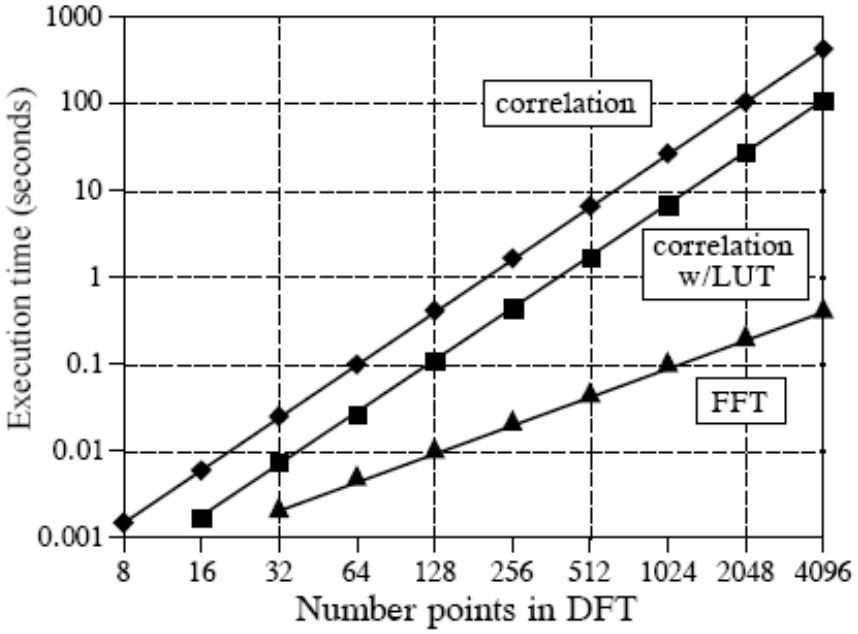

Рисунок 5.8 - Время вычисления ДПФ

БПФ имеет и другое преимущество кроме скорости. БПФ вычисляется более *точно*, так как числовой результат вычисляется с меньшей ошибкой округления. Это можно продемонстрировать следующим образом, взять БПФ от произвольного сигнала, и затем полученный частотный спектр пропустить через инверсное БПФ. Будет реконструирован исходный сигнал временной области, но с *отличием*, вызванным добавлением шума округления при вычислениях. Единственное число, характеризующее этот шум, может быть получено вычислением стандартного отклонения одного сигнала от другого. Ту же процедуру можно повторить при использовании корреляционного метода для вычисления ДПФ и соответствующего инверсного ДПФ. Каков будет шум округления для БПФ по сравнению с корреляционным методом вычисления ДПФ? Смотрите рисунок 5.9. БПФ не только *быстрее* вычисляет

ДПФ, чем корреляционный метод, но у него также меньше ошибка округления.

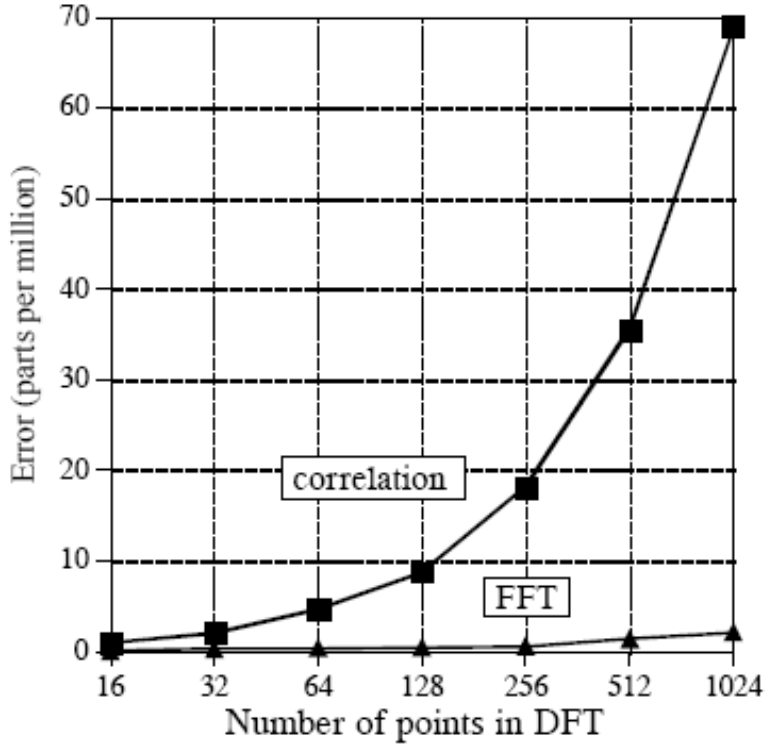

Рисунок 5.9 - Точность ДПФ

Дальнейшее увеличение скорости

Есть несколько способов сделать БПФ быстрее; однако улучшение составит только 20-40%. Один из этих способов заключается в прекращении разложения временной области на два шага раньше, когда каждый сигнал состоит только из четырех точек. Вместо вычисления последних двух шагов используется высоко оптимизированный код для прыжка прямо в частотную область, используя простоту четырех точек синусов и косинусов.

Другой популярный алгоритм, который экономит число вычислений, связан с мнимой частью временной области, состоящей из нулей, и симметрией частотного спектра. Другими словами, БПФ модифицируется для вычисления *реального* ДПФ вместо *комплексного* ДПФ. Эти алгоритмы называются реальным БПФ и реальным инверсным БПФ (или похожими названиями). Они на 30% быстрее, чем обычные программы БПФ. В таблицах 5.6 и 5.7 приведены программы с этими алгоритмами.

Имеется два маленьких неудобства при использовании *реального* БПФ. Первое, код программы примерно в два раза длиннее. Если это не беспокоит ваш компьютер, вы можете запустить еще одну программу на вашем компьютере. Второе, отладка этих программ немного сложнее, так как нельзя использовать для контроля симметрию сигнала. Эти алгоритмы

устанавливают мнимую часть временной области в нули, и частотная область имеет лево-правую симметрию. Для отладки сравнивайте выход этих программ с выходом обычных программ БПФ.

Таблица 5.6 – Инверсное БПФ для реального сигнала

```
4000 'ИНВЕРСНОЕ БПФ ДЛЯ РЕАЛЬНОГО СИГНАЛА
4010 'До возвращения, N% содержит число точек инверсного ДПФ, REX[ ] и
4020 'IMX[ ] содержат реальную и мнимую части частотной области с
индексами
4030 'от 0 до N%/2. Оставшиеся отсчеты в REX[ ] и IMX[ ] игнорируются.
4040 'После возвращения, REX[ ] содержит реальную часть временной
области, IMX[ ] - нули.
4050 '
4060 '
4070 FOR K% = (N%/2+1) TO (N%-1)                    'Делаем частотную
область симметричной
4080    REX[K%] = REX[N%-K%]                        '(как в таблице 5.1)
4090    IMX[K%] = -IMX[N%-K%]
4100 NEXT K%
4110 '
4120 FOR K% = 0 TO N%-1                             'Складываем реальную
и мнимую части
4130    REX[K%] = REX[K%]+IMX[K%]
4140 NEXT K%
4150 '
4160 GOSUB 3000                                     'Вычисляем прямое реальное
ДПФ (TABLE 12-6)
4170 '
4180 FOR I% = 0 TO N%-1                             'Складываем реальную
и мнимую части
4190    REX[I%] = (REX[I%]+IMX[I%])/N%              'и делим временную
область на N%
4200    IMX[I%] = 0
4210 NEXT I%
4220 '
4230 RETURN
```

Таблица 5.7 – БПФ для реальных сигналов

```
3000 'БПФ ДЛЯ РЕАЛЬНЫХ СИГНАЛОВ
3010 'До возвращения, N% содержит число точек в ДПФ, REX[ ] содержит
3020 'реальный входной сигнал, величины в IMX[ ] игнорируются. После
возвращения,
3030 'REX[ ] и IMX[ ] содержат выход ДПФ. Индексы всех сигналов
меняются от 0 до N%-1.
3040 '
3050 NH% = N%/2-1                          'Разделяем четные и
нечетные точки
3060 FOR I% = 0 TO NH%
3070    REX(I%) = REX(2*I%)
3080    IMX(I%) = REX(2*I%+1)
3090 NEXT I%
3100 '
3110 N% = N%/2                             'Вычисляем N%/2 точек БПФ
3120 GOSUB 1000                            '(GOSUB 1000 is the FFT in
Table 12-3)
3130 N% = N%*2
3140 '
3150 NM1% = N%-1                           'Четно/нечетное разложение
частотной области
3160 ND2% = N%/2
3170 N4% = N%/4-1
3180 FOR I% = 1 TO N4%
3190    IM% = ND2%-I%
3200    IP2% = I%+ND2%
3210    IPM% = IM%+ND2%
3220    REX(IP2%) = (IMX(I%) + IMX(IM%))/2
3230    REX(IPM%) = REX(IP2%)
3240    IMX(IP2%) = -(REX(I%) - REX(IM%))/2
3250    IMX(IPM%) = -IMX(IP2%)
3260    REX(I%) = (REX(I%) + REX(IM%))/2
3270    REX(IM%) = REX(I%)
3280    IMX(I%) = (IMX(I%) - IMX(IM%))/2
3290    IMX(IM%) = -IMX(I%)
3300 NEXT I%
3310 REX(N%*3/4) = IMX(N%/4)
3320 REX(ND2%) = IMX(0)
3330 IMX(N%*3/4) = 0
3340 IMX(ND2%) = 0
3350 IMX(N%/4) = 0
3360 IMX(0) = 0
3370 '
```

Окончание таблицы 5. 7

```
3380 PI = 3.14159265                    'Завершаем последнюю
ступень БПФ
3390 L% = CINT(LOG(N%)/LOG(2))
3400 LE% = CINT(2^L%)
3410 LE2% = LE%/2
3420 UR = 1
3430 UI = 0
3440 SR = COS(PI/LE2%)
3450 SI = -SIN(PI/LE2%)
3460 FOR J% = 1 TO LE2%
3470    JM1% = J%-1
3480    FOR I% = JM1% TO NM1% STEP LE%
3490       IP% = I%+LE2%
3500       TR = REX[IP%]*UR - IMX[IP%]*UI
3510       TI = REX[IP%]*UI + IMX[IP%]*UR
3520       REX[IP%] = REX[I%]-TR
3530       IMX[IP%] = IMX[I%]-TI
3540       REX[I%] = REX[I%]+TR
3550       IMX[I%] = IMX[I%]+TI
3560    NEXT I%
3570    TR = UR
3580    UR = TR*SR - UI*SI
3590    UI = TR*SI + UI*SR
3600 NEXT J%
3610 RETURN
```

Рисунки 5.10 и 5.11 иллюстрируют, как работает реальное БПФ. На рисунке 5.10 (a) и (b) показан сигнал временной области, который состоит из импульса в реальной части и нулей в мнимой части. Рисунки (c) и (d) показывают соответствующий частотный спектр. Как описывалось ранее, реальная часть частотной области имеет *четную* симметрию относительно отсчетов 0 и N/2, в то время как мнимая часть имеет *нечетную* симметрию относительно тех же точек.

Теперь рассмотрим рисунок 5.11, где импульс в мнимой части временной области, а реальная часть вся состоит из нулей. Симметрия в частотной области *обратилась*, реальная часть нечетная (odd), а мнимая часть четная (even).

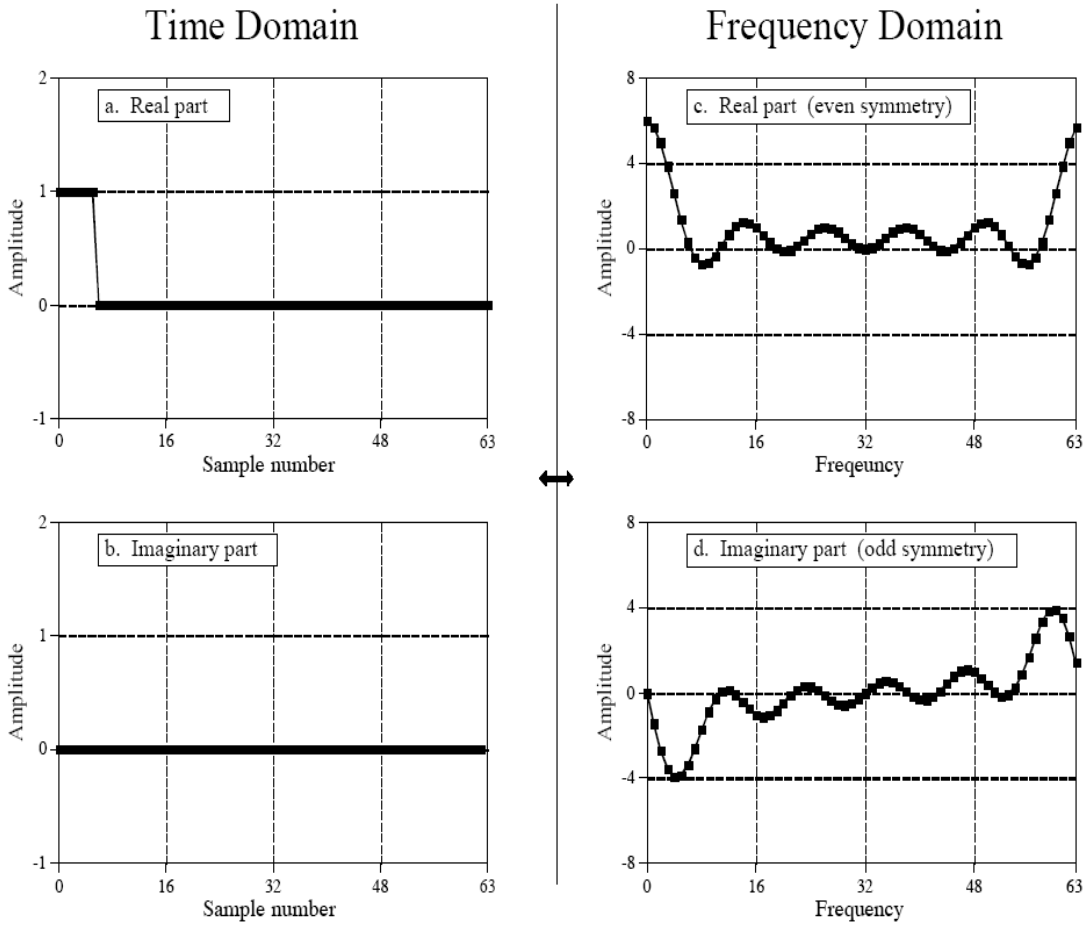

Рисунок 5.10 - Симметрична реальная часть ДПФ

Что будет, если сигнал присутствует в *обеих частях* временной области? Благодаря свойству аддитивности, частотная область будет *суммой* двух частотных спектров. Теперь ключевой элемент: частотный спектр, сформированный этими двумя типами симметрии, может быть полностью разделен на два составляющих сигнала. Это достигается *четным/нечетным разложением*.. Другими словами, два реальных ДПФ могут быть вычислены ценой одного БПФ. Один из сигналов располагается в реальной части временной области, другой – в мнимой части. После вычисления комплексного ДПФ (через БПФ, конечно) спектры разделяются, используя четное/нечетное разложение. Когда два и более сигналов необходимо пропустить через БПФ, этот прием уменьшает время вычисления, примерно, на 40%. Улучшение не полностью получается в два раза, так как требуется время на четное/нечетное разложение. Это относительно простой способ с несколькими рытвинами, ничего похожего на черновики записанных программ БПФ.

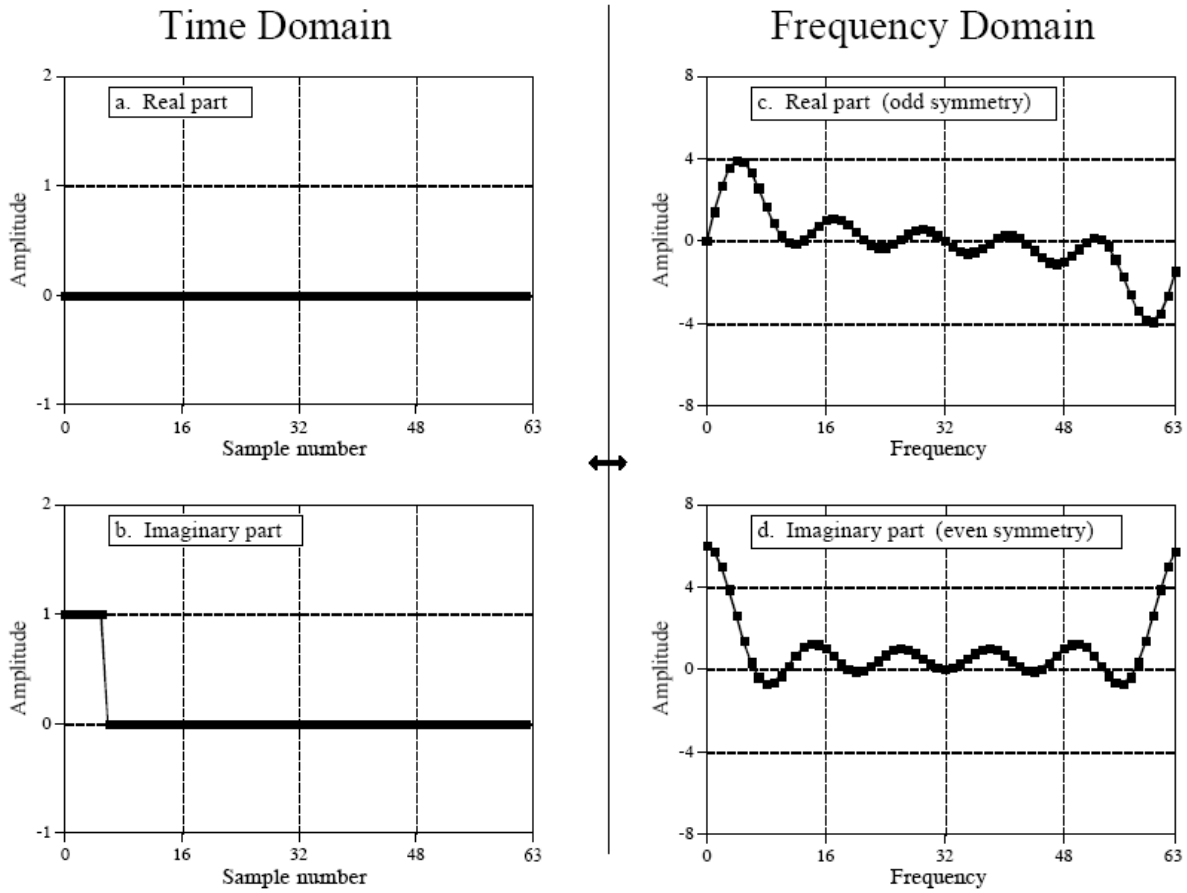

Рисунок 5.11 - Симметрична мнимая часть ДПФ

Следующий шаг заключается в модификации алгоритма для более быстрого вычисления *одного* ДПФ. Это неприятно, но можно сделать. Входной сигнал разбивается на половинки, используя чередующееся разложение. N/2 четных точек помещаются в реальную часть сигнала временной области, а N/2 нечетных точек – в мнимую часть. Затем вычисляется N/2-точечное БПФ, требующее в два раза меньшее время, чем N-точечное БПФ. Полученный частотный спектр затем разделяется с помощью четного/нечетного разложения, превращаясь в частотные спектры двух чередующихся сигналов временной области. Два частотных спектра потом комбинируются в один спектр, так же как на последнем шаге БПФ.

БПФ в *Цифровой Обработке Сигналов* есть то же самое, что и *транзистор* в *электронике*. Это есть основа технологии; каждый специалист в этой области знает его характеристики и как его использовать. Однако только небольшое число специалистов понимают детали его внутренней работы.

3.4 Содержание отчета

Отчет должен состоять из следующих частей:

- введение;
- постановка задачи;

- основная часть;
- заключение;
- приложение.

Список литературы

1. Канаев Ф.Ю., В.П. Лукин. Адаптивная оптика. Численные и экспериментальные исследования. – Томск: Изд-во Института оптики атмосферы СО РАН, 2005. – 249 с.

2. Магдич Л.Н., Молчанов В.Я. Акустооптические устройства и их применение.- М.: Сов. радио, 1978, 112 с.

3. Максфилд К. Проектирование на ПЛИС - Курс молодого бойца. – М.: Издательский дом «Додэка XXI», 2007. -408 с.

4. Василенко Г.И., Цибулькин Л.М. Голографические распознающие устройства. –М.: Радио и связь, 1985.-312 с.

Учебное пособие

Слядников Е.Е.

Быстрый алгоритм вычисления дискретного преобразования Фурье

Методические указания к лабораторной работе
по дисциплине «Основы оптоинформатики»

Усл. печ. л. _____ Препринт
Томский государственный университет
систем управления и радиоэлектроники
634050, г.Томск, пр.Ленина, 40