

*Лабораторные работы кафедры ВТ ИКТИБ ЮФУ для курсов «Проблемно-ориентированные методы и средства цифровой обработки сигналов», «Специализированные методы и средства цифровой фильтрации»*

## **Лабораторная работа №5.**

### *Конвейерная реализация КИХ-фильтра на ПЛИС*

#### **Цель работы.**

Лабораторная работа №5 нацелена на изучение принципов аппаратной конвейерной реализации блока КИХ-фильтрации цифрового сигнала. Также лабораторная работа направлена на: изучение студентами принципов использования ПЛИС в области цифровой обработки сигналов; изучение принципов построения параллельных схем; изучение принципов аппаратной реализации алгоритмов с «плавающим окном».

#### **Ход работы:**

1. Выполнить построение дискретного сигнала с использованием языка Python, как это было сделано в п.1 лабораторной работы №4:

$$x_n = \sum_{k=1}^7 A_k \cdot \sin(2\pi \cdot f_k \cdot t_n)$$

**Изменить частоту дискретизации:**  $fd = 2.2 * \max(f)$  КГц. Количество семплов  $N = 2048$ .

2. По графику АЧХ со шкалой абсолютных частот (построенному в п.5 ЛР4) переопределить новые диапазоны частот, относящихся к полосам пропускания и полосам заграждения КИХ-фильтра. Для этого необходимо определить частоты среза фильтра (на рисунке красная линия): частоту, при которой наблюдается спад амплитуды сигнала в  $1/\sqrt{2}$  раза (на 3 дБ), а именно, найти точки пересечения графика АЧХ с прямой, отражающей значение амплитуды  $A = 1/\sqrt{2}$  (на рисунке синяя линия). В зависимости

от типа фильтра, может присутствовать как единственная частота среза (для ФНЧ и ФВЧ), так и несколько (для полосовых и режекторных фильтров).



3. **Пересчитать** эталонный модельный сигнал, состоящий только из тех гармоник, которые согласно варианту, попадают в диапазон **полос пропускания** КИХ-фильтра. Для этого по графику АЧХ фильтра в абсолютных частотах (построенный в п.4 лабораторной работы №5) необходимо определить, какие частоты были отфильтрованы, а какие нет. Эталонный модельный сигнал рассчитывается аналогично исходному сигналу по формуле:

$$x_n = \sum_{m=1}^M A_m \cdot \sin(2\pi \cdot f_m \cdot t_n)$$

где  $A_m$  и  $f_m$  — амплитуды и частоты гармоник, попадающих в **полосы пропускания** фильтра;  $M$  — число таких гармоник.

4. Исходный сигнал  $x_n$ , полученный в п.1 данной работы, конвертировать в целые числа, отбросив все знаки после запятой с помощью функции `int()`, по примеру, как это было сделано в п.1 лабораторной работы №2.
5. Сохранить полученный дискретный сигнал в виде и формате \*.mif файла, для инициализации памяти в САПР Quartus 9.2. Подробнее о формате \*.mif файлов можно узнать здесь:

[https://manpages.ubuntu.com/manpages/bionic/man5/srec\\_mif.5.html](https://manpages.ubuntu.com/manpages/bionic/man5/srec_mif.5.html)

Для сохранения дискретного сигнала в виде и формате \*.mif файла можно использовать следующий код:

```

discret_signal_file = "signal.mif"
with open(discret_signal_file, 'w') as f:
    f.write(f"ADDRESS_RADIX=DEC;\n")
    f.write(f"DATA_RADIX=DEC;\n")
    f.write(f"CONTENT BEGIN\n")
    for idx in range(len(discret_signal)):
        f.write(f"\t{idx} : {str(discret_signal[idx])};\n")
    f.write(f"END;")

```

6. Масштабировать значения  $H$  коэффициентов импульсной характеристики КИХ-фильтра, отобразив их на разрядную сетку, заданную вариантом лабораторного задания, по примеру, как это было реализовано в п.2 лабораторной работы №2. Для этого рассчитать диапазон разрядной сетки по формуле:  $D = 2^q$ , где  $q$  – требуемая разрядность данных, а шаг разрядной сетки равен  $1/D$ . Так как работа ведется над числами со знаком (signed), старший бит от общей разрядности отведен под знак. В связи с этим, масштабировать коэффициенты необходимо согласно формуле:  $H_{scale} = H * D/2$ .

После масштабирования необходимо конвертировать полученные коэффициенты в целые числа, отбросив все знаки после запятой с помощью функции `int()`, по примеру, как это было сделано в п.4 данной работы.

7. Относительно варианта лабораторного задания, выбрать одну из предложенных схем: с каскадным или параллельным суммированием. В случае, если импульсная характеристика КИХ-фильтра является симметричной (часть коэффициентов в точности равна второй части, записанной в обратном порядке), помимо выбора типа суммирования, следует выбрать соответствующую схему.
8. В САПР Quartus 9.2 реализовать схему фильтра, выполняющую операции над **целыми знаковыми** числами с разрядностью, выбранной согласно варианту лабораторного задания.

Реализацию схемы КИХ-фильтра необходимо выполнить с помощью мегафункций Quartus. Латентность операций целочисленного

сложения/вычитания со знаком (signed):  $latency_{add/sub} = 1$ . **Латентность (latency, clock cycles) операций целочисленного умножения со знаком (signed) определяется вариантом лабораторного задания.** При реализации схемы не забыть после операции умножения обрезать младшие  $q - 1$  разрядов и один старший биты для сохранения разрядности вычислений, как это было реализовано в лабораторной работе №3.

9. Реализовать схему тестового окружения:

- а. память ROM: инициализировать \*.mif-файлом, полученным в п.5 данной работы и содержащим входную последовательность отсчетов сигнала  $X_n$ );
- б. память RAM для записи результата работы схемы фильтрации.

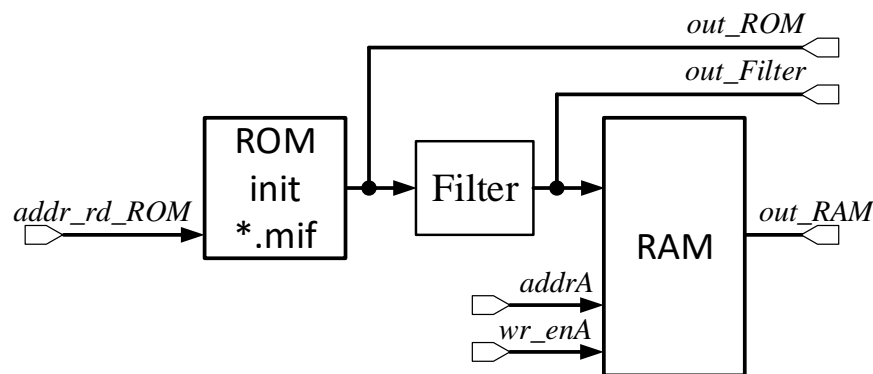


Рисунок 1 – схема тестового окружения реализации КИХ-фильтра

10. Выполнить функциональное моделирование схемы. Для чтения данных из памяти ROM подать на входной порт  $addr\_rd\_ROM$  счетчик с инкрементом (кнопка на левой панели окна моделирования Quartus 9.2). **Запись результата осуществить в память RAM начиная с того момента, как данные появятся на выходе блока КИХ-фильтрации Filter.**
11. Сохранить содержимое памяти RAM по результатам Simulation Report в виде текстового файла.
12. Загрузить данные, полученные в предыдущем пункте работы, в код Python лабораторной работы №4. Так как после загрузки файла данные будут

представлены в строковом формате, их необходимо поэлементно преобразовать в формат `int` и записать в массив. Для этого можно **частично** использовать следующий код:

```
quartus_result_file_data = []
with open("quartus_result_file.txt", 'r') as f:
    for line in f:
        quartus_result_file_data.append(int(line))
```

13. Вывести в семплах на одном графике:

- a. Исходный сигнал
- b. Эталонный модельный сигнал
- c. Сигнал, полученный после аппаратной фильтрации во временной области.

14. В отчете сделать вывод о точности вычислений в ПЛИС.

15. В отчете привести RTL-схему, полученную по результатам компиляции проекта.

16. В отчете привести ресурс ПЛИС, требуемый на реализацию всей схемы, полученную по результатам компиляции проекта.

17. В отчете привести данные о максимально возможной тактовой частоте схемы, полученной по результатам компиляции проекта. Для этого найти максимальное значение в разделе Timing Analyzer -> Summary и взять обратную величину от него. Например, если максимальное значение периода тактовой частоты равно 12.685ns, значение максимальной тактовой частоты равно  $1 \cdot 10^9 / 12.685$  Гц.

## **Варианты заданий к лабораторной работе №5**

Варианты лабораторного задания выбираются согласно номеру студента в общем списке группы. Нечетный номер реализует схему КИХ-фильтра с **каскадным суммированием**, четный — с **последовательным суммированием**.

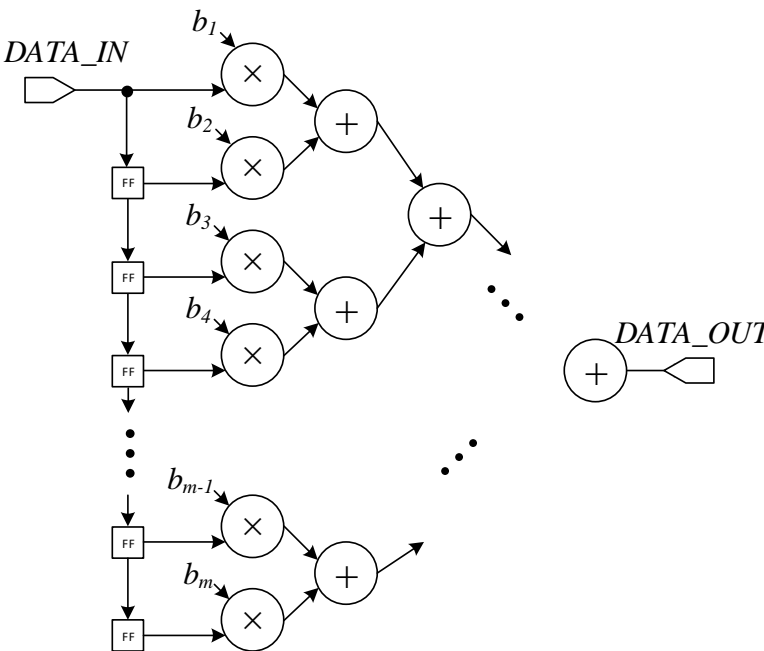
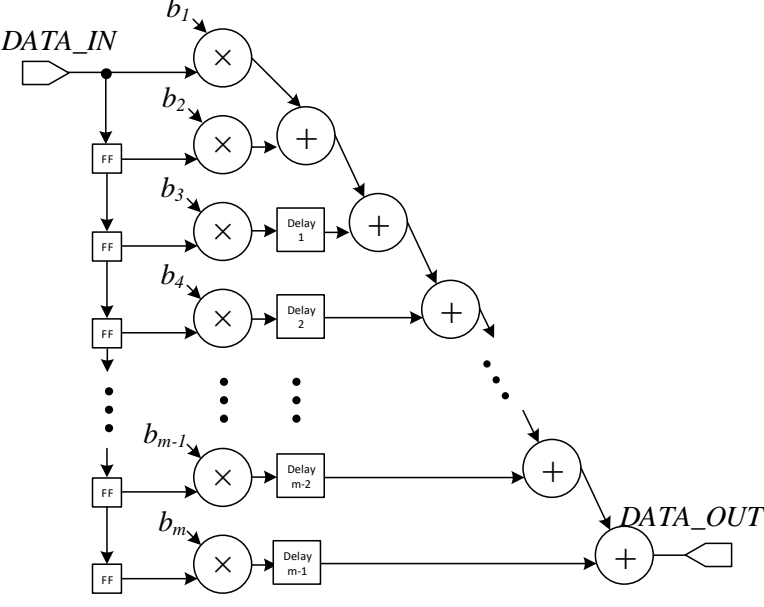
В таблице приведены значения частот и амплитуд для генерации модельного сигнала, латентность операции умножения и разрядность выполняемых операций.

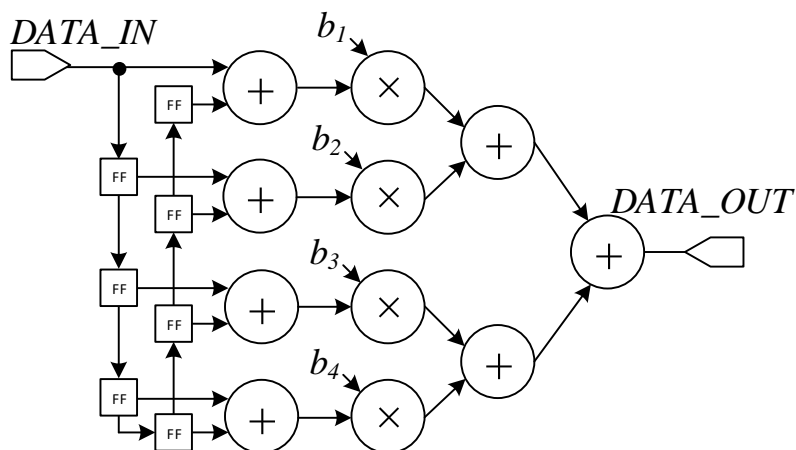
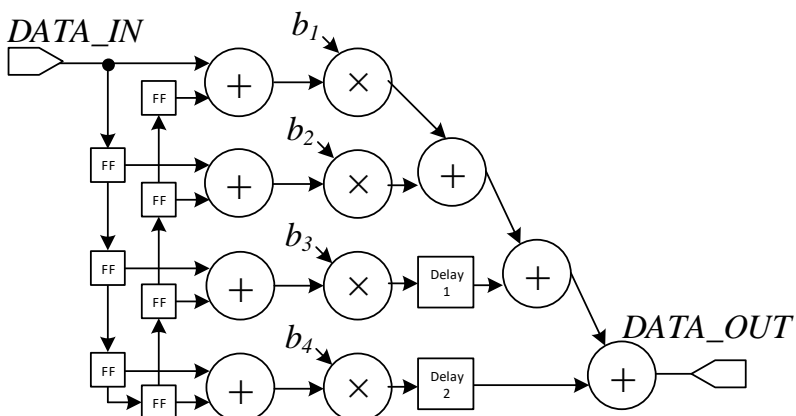
Таблица 1 – варианты лабораторных заданий

№		1	2	3	4	5	6	7	Латентность операции умножения:	Разрядность:
1	F, кГц	0,5	1	2	5	7	9	12	3 такта	8 бит
	A	1	5	3	7	3	2	1		
2	F, кГц	3	5	6	9	12	15	18	4 такта	9 бит
	A	7	9	8	12	7	4	6		
3	F, кГц	0,9	1	2	9	12	15	18	5 тактов	10 бит
	A	2	5	3	12	7	4	6		
4	F, кГц	3	5	6	8	9	10	12	6 тактов	11 бит
	A	7	9	8	7	3	2	5		
5	F, кГц	0,5	1	2	9	12	14	17	3 такта	12 бит
	A	1	5	3	12	15	4	6		
6	F, кГц	2	4	5	7	9	12	15	4 такта	13 бит
	A	11	15	12	13	21	10	9		
7	F, кГц	9	11	15	17	21	29	35	5 тактов	14 бит
	A	17	19	36	34	28	25	40		
8	F, кГц	0,3	0,56	1,2	3,5	5	7	9	6 тактов	15 бита
	A	21	18	23	19	13	20	11		
9	F, кГц	10	13	14	17	21	22	25	3 такта	16 бита
	A	9	14	15	19	21	26	30		
10	F, кГц	1	5	7	13	18	19	29	4 такта	17 бита
	A	9	11	12	17	21	29	35		
11	F, кГц	10	11	12	13	15	20	25	5 тактов	18 бита
	A	9	11	15	17	21	29	35		
12	F, кГц	21	25	32	33	41	45	50	6 тактов	17 бита
	A	9	11	15	17	21	29	35		
13	F, кГц	20	21	22	23	27	30	40	3 такта	16 бита
	A	9	11	15	17	21	29	35		
14	F, кГц	11	15	12	13	21	23	25	4 такта	15 бит
	A	19	12	15	27	23	29	25		
15	F, кГц	5	11	12	13	21	23	30	5 тактов	14 бит
	A	9	11	15	17	21	29	35		
16	F, кГц	11	13	15	17	21	23	25	6 тактов	13 бит
	A	22	23	25	17	21	29	35		
17	F, кГц	1	2	3	7	9	11	13	3 такта	12 бит
	A	2	12	11	15	3	8	13		
18	F, кГц	3	9	14	16	21	26	32	4 такта	11 бит
	A	12	2	17	5	9	15	6		
19	F, кГц	5	7	12	17	24	26	43	5 тактов	10 бит
	A	5	14	3	23	6	20	7		
20	F, кГц	1	4	14	19	20	23	29	6 тактов	8 бит
	A	8	5	12	7	3	15	8		

Различные виды схем КИХ-фильтров приведены в таблице 2.

Таблица 2 – виды схем КИХ-фильтров

<p>Схема КИХ-фильтра каскадным суммированием</p>	
<p>Схема КИХ-фильтра последовательным суммированием</p> <p>Элементы <b>Delay</b> – задержка на регистрах <math>lpm\_dff</math>, на <math>latency_{add/sub}</math> тактов.</p>	

<p>Схема фильтра каскадным суммированием симметричными коэффициентами</p>	<p>КИХ- с</p>	 <p>Схему необходимо масштабировать под заданное число коэффициентов!</p>
<p>Схема фильтра последовательным суммированием симметричными коэффициентами Элементы Delay – задержка на регистрах lpm_dff, на <math>latency_{add/sub}</math> тактов.</p>	<p>КИХ- с</p>	 <p>Схему необходимо масштабировать под заданное число коэффициентов!</p>



## **Требования к отчету и защита**

Процесс выполнения лабораторной работы документируется с помощью текстового редактора MS Word, полученные сведения служат основой для формирования отчета о выполнении лабораторной работы. Отчет в общем случае должен включать:

- титульный лист;
- описание задач в выбранном варианте лабораторной работы;
- схемы из Quartus 9.2
- отчет о компиляции проекта
- временная диаграмма моделирования проекта
- все графики, полученные в ходе работы в Python
- выводы
- листинг программы

Защита отчета о выполнении лабораторной работы сопровождается демонстрацией работоспособности кода программ, теоретических знаний и ответов на дополнительные вопросы преподавателя по теме занятия.

## **ПРИЛОЖЕНИЕ 1.**

### ***1. Инструкция по установке IDE PyCharm.***

- a. Зайти на страницу загрузки IDE PyCharm с официального сайта JetBrains, опуститься вниз страницы и нажать Download PyCharm Community Edition. Данная версия программного продукта не нуждается в лицензировании.

<https://www.jetbrains.com/pycharm/download/?section=windows>

- b. Установить IDE PyCharm. При установке выбрать все чекбоксы.
- c. С официального сайта Python скачать и установить последнюю версию интерпретатора языка. При установке отметить все чекбоксы (обязательно выбрать «Add python.exe to PATH»).

<https://www.python.org/downloads/>

### ***2. Инструкция по установке VSCode и начальная настройка среды.***

- a. Зайти на главную страницу официального сайта VSCode и нажать кнопку Download. VSCode не нуждается в лицензировании.
- b. Установить VSCode. При установке выбрать все чекбоксы.
- c. Открыть среду VSCode и зайти в раздел расширения (Extensions).



- d. Найти и установить расширение «Russian Language Pack for Visual Studio Code» (опционально).
- e. Найти и установить расширение «Python».
- f. С официального сайта Python скачать и установить последнюю версию интерпретатора языка. При установке отметить все чекбоксы (обязательно выбрать «Add python.exe to PATH»).


<https://www.python.org/downloads/>

### ***3. Подготовка работы с IDE.***

- a. Проверить, установлена ли на компьютере IDE PyCharm или VSCode. Если нет, выполнить установку согласно описанной выше инструкции.
- b. Создать рабочую директорию с вашей фамилией. Желательно, чтобы папка находилась не на рабочем столе Windows.
- c. Запустить IDE и выполнить команду File => Open Folder, где выбрать путь к созданной вами директории.
- d. Создать новый файл Python File: команда File => New => Python File и задать имя файла с расширением \*.py (например, lab1.py).
- e. Файл открылся в правой области окна проекта. Здесь можно набирать код программы.

#### **4. Пример написания и запуска программы в IDE.**

Хорошей традицией при изучении первого языка программирования является написание программы, выводящей на экран компьютера приветствие «Hello World!». Для создания такой программы необходимо предварительно создать и открыть новый python файл, как это было описано выше.

- a. Наберите команду: `print('Hello Python!')`
- b. Выполните запуск программы нажав на клавишу RUN , расположенную в правом верхнем углу IDE. Результат работы программы появится мгновенно в окне вывода, расположенном в нижней части IDE.
- c. Для запуска программы в первый раз нужно щелкнуть правой кнопкой мыши в окне с текстом программы и выбрать пункт Run.
- d. Если запуск в IDE PyCharm произошел с ошибкой, проверьте, указан ли интерпретатор языка в окне Run => Edit Configuration.

#### **5. Инструкция по установке пакетов matplotlib, scipy, numpy.**

- a. В IDE VSCode зайти в раздел расширения (Extensions); в IDE PyCharm открыть вкладку Python Packages.
- b. Найти и установить библиотеки matplotlib, scipy, numpy.