МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Вятский государственный университет»**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Допущено к защите

Руководитель проекта

\_\_\_\_\_\_\_\_/Караваева О.В./

(подпись) (Ф.И.О)

«\_\_»\_\_\_\_\_\_\_\_\_\_2023г.

**Проектирование процессорного элемента систолической системы**

Пояснительная записка курсового проекта по дисциплине

«Комплекс знаний бакалавра в области программного и аппаратного обеспечения вычислительной техники»

ТПЖА.09.03.01.285 ПЗ

Разработал студент группы ИВТ-33 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Пестов М.А./

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Караваева О.В./

Консультант \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Крутиков А.К./

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (дата)

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ \_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

Киров 2023

РЕФЕРАТ

Пестов М.А.. Проектирование процессорного элемента систолической системы. ТПЖА.090301.285 ПЗ: Курс. проект / ВятГУ, каф. ЭВМ; рук. Караваева О.В. - Киров, 2022. – ПЗ 32с, 18 рис., 3прил.

Объект исследования – специализированные вычислительные системы с систолической архитектурой.

Предмет курсового проекта – **архитектура** процессорного элементаспециализированной систолической вычислительной системы.

Цель курсового проекта – повышение качества обучения по дисциплине «Организация ЭВМ» за счёт разработки, в дальнейшем, программного эмулятора процессорного модуля систолической вычислительной системы.

Результатом выполнения курсового проекта является архитектура и организация функционирования специализированного процессорного элемента.

Содержание

[Введение 5](#_Toc126579976)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6](#_Toc126579977)

[1.1 Общие сведения о систолических системах 6](#_Toc126579978)

[1.2 Обзор аналогов 9](#_Toc126579979)

[1.2.1 Структура процессора Intel Itanium 2 9](#_Toc126579980)

[1.2.2 Структура процессора AMD Athlon 10](#_Toc126579981)

[1.2.4 Универсальный лабораторный стенд 12](#_Toc126579982)

[1.3 Расширенное техническое задание 13](#_Toc126579983)

[1.3.1 Полное наименование системы и её условное обозначение 13](#_Toc126579984)

[1.3.2 Наименование разработчика системы и реквизиты заказчика 13](#_Toc126579985)

[1.3.3 Основания для создания системы 13](#_Toc126579986)

[1.3.4 Плановые сроки начала и окончания работы по созданию системы 13](#_Toc126579987)

[1.3.5 Источник финансирования работ по созданию системы. 13](#_Toc126579988)

[1.3.6 Порядок оформления и предъявления заказчику результатов работ по созданию системы: 13](#_Toc126579989)

[1.3.7 Назначение и цели создания системы 14](#_Toc126579990)

[1.3.9 Требования к системе 14](#_Toc126579991)

[1.3.9.1 Требования к системе в целом 14](#_Toc126579992)

[1.3.9.2 Требование к видам обеспечения. 15](#_Toc126579993)

[1.4 Выводы по первому разделу 16](#_Toc126579994)

[2. Организация потоков данных в процессоре 17](#_Toc126579995)

[2.1 Диаграммы моделей взаимодействия 17](#_Toc126579996)

[2.2 Функциональные диаграммы 18](#_Toc126579997)

[2.3 Диаграммы потоков данных 19](#_Toc126579998)

[2.5 Выводы по разделу 2 21](#_Toc126579999)

[3. Реализация функциональной схемы 22](#_Toc126580000)

[3.1. Определение взаимодействия узлов и блоков процессора 22](#_Toc126580001)

[3.2 Описание системы команд 25](#_Toc126580002)

[3.3. Методика работы с моделью процессора 27](#_Toc126580003)

[3.2 Выводы по разделу 3 28](#_Toc126580004)

[Заключение 29](#_Toc126580005)

[Приложение А 31](#_Toc126580006)

[Приложение Б 32](#_Toc126580007)

# Введение

На сегодняшний день существует множество архитектур и моделей вычислений. Каждые обладают рядом плюсов и минусов, поэтому сложно выбрать универсальный подход для решения поставленной задачи. Существуют системы, которые решают все задачи, но из-за этого скорость вычислений уменьшается. Но так же есть системы, которые являются ускоспециализированными, решают только класс поставленных задач. Преимущество такого подхода заключается в том, что каждая такая задача будет решаться быстрее. Недостатком же таких систем является то, что такая система будет решать только те задачи, под которые она настроена, а настраивать так систему это значить сделать для нее отдельную конфигурацию, что требует дополнительного времени и средств. Так же система сможет выполнять только узкий тип задач.

Систолическая система настраивается для каждой задачи отдельно, и выполняет только конкретно поставленную задачу. Само соединение процессорных элементов настроено для выполнения конкретных алгоритмов, что дает большой выигрыш в скорости. Так же в такой системе используется локальная память, чтобы при считывании из оперативной памяти операнд можно было бы обрабатывать и передавать другому процессорному элементу, чтобы вновь не записывать элемент в оперативную память. Это дает прирост скорости, потому что операция записи и чтения из памяти является достаточно долгой. Поэтому такую систему удобно использовать для конкретных алгоритмов.

# 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данном разделе рассматривается развитие различных архитектур и систем, а так же техническое задания для лабораторной установки.

## 1.1 Общие сведения о систолических системах

Электронные вычислительные машины изначально создавались как помощь и ускорение вычислений. Главной целью все было сокращение времени вычисления, путем увеличения производительности системы. Задачи становились более сложными, количество данных становилось все больше. За этим требовалось повышать производительность систем, а так же надежность, так как ошибки в ряде вычислений могли привести к тяжелым последствиям (в зависимости от поставленной задачи). Основные архитектуры вычислительных систем были предложены еще в 80-х годах 20-го века, вносились только некоторые правки и усовершенствования. Далее производительность увеличивалась за счет увеличения элементной базы.

На сегодняшний день есть 2 пути ускорения вычислений с точки зрения элементной базы: увеличение количества элементов и усовершенствование элементов. Есть несколько прототипов новой элементной базы, но они работают не надежно и не полностью исследованы. Поэтому увеличением количества элементов заинтересованы большинство компаний, выпускающие элементы вычислительной системы.

Так же для ускорения вычислительных систем создаются узкоспециализированные системы, которые решают конкретные задачи, но для этого приходится разрабатывать для каждого такого класса задач специальную систему, что требует дополнительных вложений.

Систолическая система относится к классу последовательно групповой организации, в которых конструкции, образующие какой-либо алгоритм, объединяются в группы. То что выполняется внутри конструкции - может выполняться параллельно, а то что между – выполняются последовательно (в зависимости от системы и алгоритма).

В этой же группе состоит векторно-конвейерная обработка. Введение векторных команд позволяет эффективнее использовать систему, так как с помощью них задается одна и та же операция над элементами одного или нескольких векторов. Вычислительные конструкции внутри групп исполняются конвейерным процессором. Плюсы такой системы заключаются в том, что для современного уровня технологии векторно-конвейерные системы являются более гибкими и эффективными с точки зрения стоимости, такая обработка достигает значительного уровня ускорения по сравнению со скалярными машинами. Самая сильная сторона того типа одновременно является и его самой слабой. С одной стороны, поскольку учитывается только форма представления программ и данных, имеется теоретическая возможность реализовать процесс векторизации с помощью специальных программных средств - так называемых автоматических векторизаторов. Но с другой стороны, что же делать, если эти программные векторизаторы не найдут формального преобразования, позволяющего превратить некоторую программно-информационную структуру в векторную. Структура векторной вычислительной системы представлен на рисунке 1.

Другой тип вычислительных машин, ориентированных на алгоритмы, содержащие главным образом векторные операции - это матричные системы. Общая идея построения таких систем чрезвычайно проста - имеется большое количество одинаковых обрабатывающих устройств, в каждый из которых поступают свои данные, но которое одновременно выполняет одну и ту же операцию. Множество процессорных элементов организованы таким образом, что они исполняют векторные команды, которые были заданы устройством управления. Каждый процессорный элемент работает отдельно и все они соединены через коммутационное устройство.

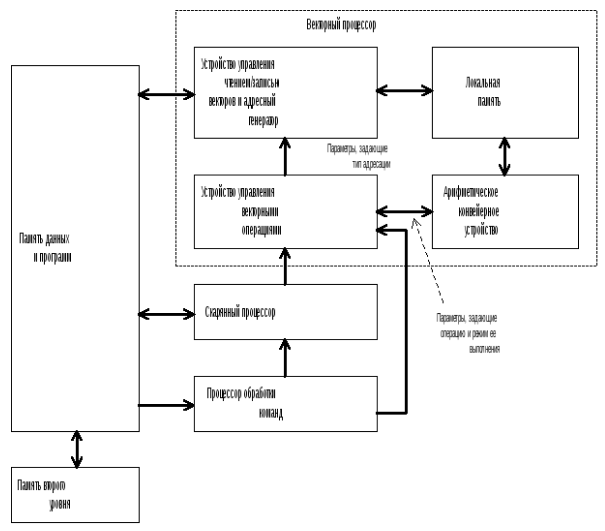


Рисунок 1 – Структура векторной вычислительной системы

Структура матричной системы показана на Рисунке 2. Доступ к данным происходит через коммутационную сеть. К недостатком можно отнести:

* в случае, когда число элементов обрабатываемых векторов не кратно числу процессоров, появляются незанятые процессоры и общая пропускная способность уменьшается;
* алгоритмы, как правило, содержат скалярные операции, выполнение которых не может перекрываться с выполнением векторных команд. Таким образом, весь массив процессоров какое-то время бездействует;
* коммутационная сеть имеет ограниченные возможности. Данные должны передаваться из памяти в соответствующие процессоры одновременно. Чтобы обеспечить возможность любому процессору работать с любым модулем памяти, необходимо использовать коммутатор перекрестных связей.

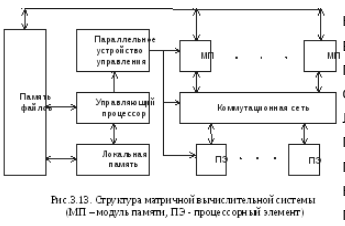


Рисунок 2– Структура матричной вычислительной системы

## 1.2 Обзор аналогов

В данном разделе можно сравнить проектируемый процессорный элемент как официально выпущенные процессорные элементы и с точки зрения учебной установки. Очевидно, что выпущенные модели будут гораздо производительнее, но их следует рассмотреть для изучения архитектур.

### 1.2.1 Структура процессора Intel Itanium 2

Первым примером послужит процессорный элемент Intel Itanium 2. Принципиальная схема представлена на Рисунке 6. Процессорный элемент работает на частоте 1.6 ГГц, размер кэша 6 MB. В нем есть 128 регистров, разрядность 32 бита. Производительность такого процессорного элемента будет больше, чем у проектируемового, но такая система имеет большую стоимость.

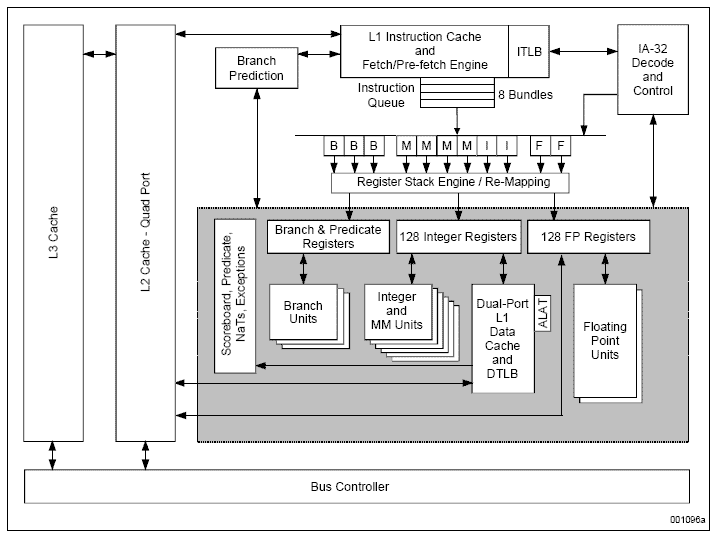


Рисунок 6 – Принципиальная схема Intel Itanium 2

### 1.2.2 Структура процессора AMD Athlon

Следующий пример будет архитектура процессорного элемента AMD. Ядро процессоров с архитектурой AMD использует другие принципы своего функционирования, по сравнению с intel. По сути Athlon является RISC процессором – в режиме реального времени выполняется преобразование потока CISC команд в унифицированные RISC команды.

1.2.3 Лабораторный стенд на основе ПКП К1810ВН59А

Далее следует рассмотреть лабораторные установки, сделанные для помощи студентам. Одной из таких установок является установка, которая обрабатывает приходящие вектора прерываний. Принцип работы ПКП К1810ВН59А зависит от типа используемого микропроцессора.

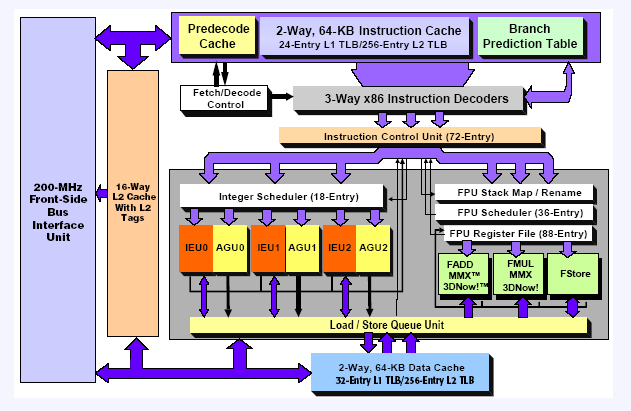


Рисунок 7 – Принципиальная схема AMD

Режим работы микросхемы устанавливается путём её программирования как устройства ввода/вывода с помощью программного обеспечения микропроцессорной системы. В процессе работы ПКП можно изменять алгоритмы обслуживания прерываний. Это осуществляется с помощью управляющих слов. Разница в проектируемом будет заключаться в том, что не нужно будет самим формировать управляющее слово, так как будут уже фиксированные команды. Структура установки показана на Рисунке 8.

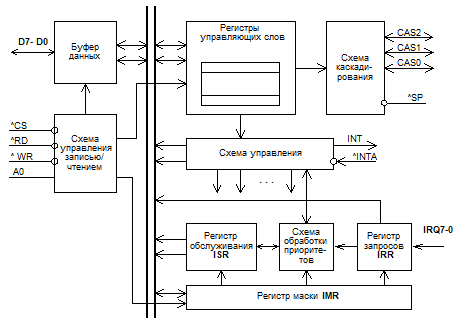


Рисунок 8 – Структура БИС

### 1.2.4 Универсальный лабораторный стенд

В следующей лабораторной установке программное обеспечение выполняет подачу управляющих сигналов в модули исследуемых устройств, принимает осведомительные сигналы, копирование состояния внутренних регистров (ячеек памяти) МИУ после каждого такта (цикла) работы; организация работы МИУ по тактам и в автоматическом режиме; обеспечение ввода пользователем управляющих воздействий (сигналов, микрокоманд) и их последовательностей (микропрограмм, программ); вывод на экран текущего и нового состояния МИУ после каждого такта (цикла) работы; запись и считывание с диска введенных микропрограмм (программ) и данных; в данной лабораторной установке набором команд можно выполнять операции, отличие в том, что выполнение происходит последовательно на одном процессоре.

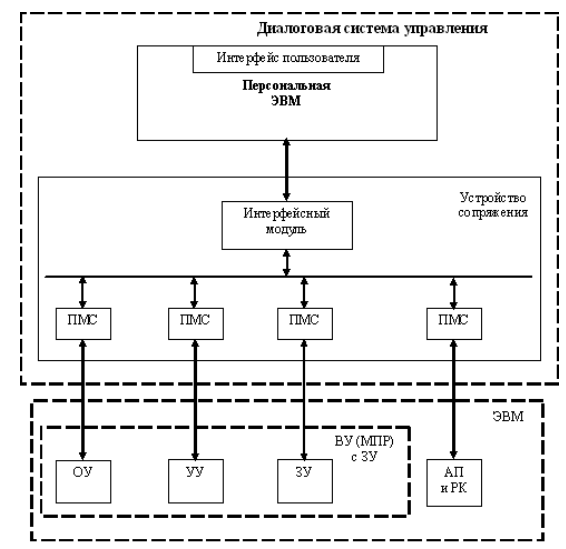


Рисунок 9 – Структура лабораторной установки

## 1.3 Расширенное техническое задание

### 1.3.1 Полное наименование системы и её условное обозначение

Архитектура процессорного элемента систолической системы

### 1.3.2 Наименование разработчика системы и реквизиты заказчика

ФГБОУ ВО «ВятГУ», кафедра электронных вычислительных машин.

Разработчик – студент группы ИВТб 3301-01-00 Пестов Максим Александрович

### 1.3.3 Основания для создания системы

Учебный план по дисциплине комплекс знаний бакалавра в области программного и аппаратного обеспечения вычислительной техники.

### 1.3.4 Плановые сроки начала и окончания работы по созданию системы

* Начало работ по созданию системы – весна 2022
* Окончание работ по созданию системы – начало лета 2022

### 1.3.5 Источник финансирования работ по созданию системы.

Собственные средства разработчика.

### 1.3.6 Порядок оформления и предъявления заказчику результатов работ по созданию системы:

К результату труда разработчика относится:

* Оригинальное программное обеспечение.
* Типовые проектные решения и особенности дизайна системы.
* Проектная и рабочая документация.

Результаты передаются заказчику частями по завершении каждой стадии работы по созданию системы

* Программное обеспечение.
* Документация – в электронном виде в формате PDF, на бумажных носителях.

Проектная документация должна быть разработана в соответствии с ГОСТ 34.201-89 и ГОСТ ЕСПД. Процедуры приемки – передачи результатов работ оформляются актами приемки-передачи.

### 1.3.7 Назначение и цели создания системы

*Назначение системы.*

Проектирование учебного эмулятора систолической системы для студентов.

*Цели создания системы.*

Целью создания системы является помощь в изучении систолической системы.

1.3.8 Характеристики объекта автоматизации

*Краткие сведения об объекте автоматизации.*

Требуется настроить операционное устройство и устройство управления, для того, чтобы правильно решать поставленную задачу в соответствии с алгоритмом.

*Сведения об условиях эксплуатации объекта автоматизации и характеристика окружающей среды:*

Пошаговое выполнения алгоритма, распределение памяти и выполнения дополнительных заданий.

### 1.3.9 Требования к системе

1.3.9.1 Требования к системе в целом

*Требования к структуре и функционированию системы*

Процессорный элемент систолической системы должен представлять собой систему, включающую в себя подсистемы:

* п/с операционного устройства
* п/с устройства управления
* п/с памяти

1. П/с операционного устройства:

* Операции, которые будут производится над данными.
* Операции обработки массивов

1. П/с устройства управления:

* Выбор, того какие операции должны будут быть выполнены.

1. П/с организация памяти:

* Прямая адресация, косвенная адрессация
* Получение данных на входе и работа только с имеющимися данными.

*Требования к средствам и способам связи для информационного обмена между компонентами системы.*

Для информационного обмена между компонентами системы требуется передача элементов из локальной памяти процессорного элементу к процессорному элементу, который следует по алгоритму.

*Требования к характеристикам взаимосвязи создаваемой системы со смежными системами, требования к её совместимости.*

Разработанное приложение должно быть совместима с операционными системи Windows 7 и выше, Linux и MacOs.

*Требования по диагностированию системы.*

При нахождении ошибок (логических или программных), разработчик должен модернизировать приложение.

*Перспективы развития, модернизация системы*

При модернизации программного обеспечения могут добавляться новые процессорные элементы, исправляться ошибки и добавляться новые операции.

*Требования к надёжности комплекса.*

Необходимо чтобы система обладала устойчивостью к отказам оборудования программных систем, а также электропитания. Требования надежности должны быть регламентированы для следующих аварийных ситуаций: Предлагать сохранение на каждом этапе работы.

1.3.9.2 Требование к видам обеспечения.

*Математическое обеспечение.*

Каждый процессорный элемент выполняет одну примитивную операцию. При соединении нескольких процессорных элементов возможно реализовать алгоритм умножения матриц.

*Требования к информационному обеспечению.*

* В качестве входной информации на процессорный элемент поступают два операнда
* На процессорном элементе выполняется одна (несколько) примитивная операция.
* Выходными данными являются результаты выполнения примитивных операций.

*Требования к методическому обеспечению.*

Состав программной документации должен включать в себя:

* Техническое задание.
* Программное обеспечение.
* Руководство пользователя.
* Пояснительная записка.

## 1.4 Выводы по первому разделу

В данном разделе рассмотрены аналоги лабораторных установок и реальные архитектуры, рассмотрены их достоинства и недостатки. Так же в разделе выдвинуты основные требования для создаваемой системы.

# 2. Организация потоков данных в процессоре

В данном разделе показывается, где находятся данные и откуда они посутают. Так же показана передача данных между процессорным элементом и оперативной памятью.

## 2.1 Диаграммы моделей взаимодействия

Диаграмма кооперацииуказывает обмен сообщениями между объектами при реализации любого варианта использования. При помощи данной диаграммы можно легко отследить, что происходит при выполнении каких-либо действий на сайте. Отслеживается взаимосвязь выходными и выходными операндами (рис. 10).

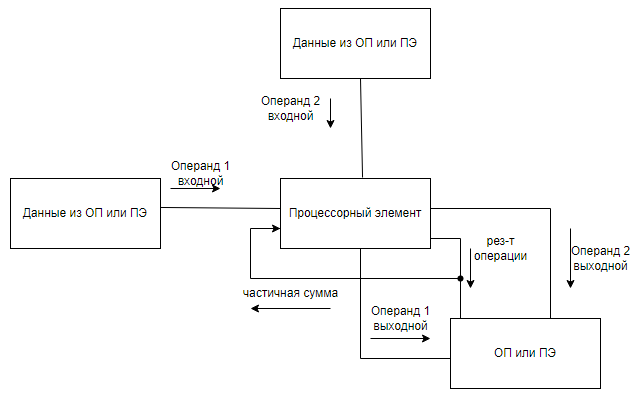


Рисунок 10 - Диаграмма кооперации

## 2.2 Функциональные диаграммы

Функциональная диаграмма верхнего уровня позволяет сформировать наиболее точное и описание разрабатываемого программного обеспечения. В данном случае в качестве отображения взаимосвязей была выбрана нотация IDEF0. В качестве входных данных выступают операнды, которые будут участвовать в команде, так же сюда передается частичная сумма от предыдущих вычислений, если они были. В выходных параметрах будет те же операнды, что и на входе, а так же частичная сумма (результат выполнения команды). Эти параметры следует передавать, потому что только первый процессорный элемент получает данные из оперативной памяти, а остальные передают их между собой, чтобы обращение к оперативной памяти было гораздо меньше (Рис. 12).

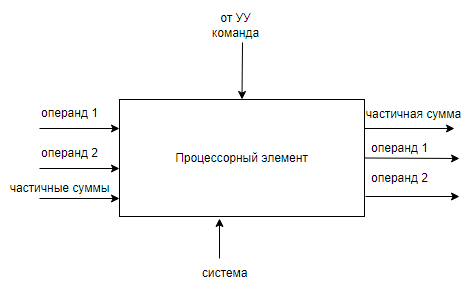


Рисунок 12 - Функциональная диаграмма верхнего уровня

Детализирующая функциональная диаграмма более подробно раскрывает функциональную диаграмму верхнего уровня: описывает взаимодействия и связи процессов, происходящих внутри системы. На ней можно увидеть, какие процессы взаимосвязаны и что между ними общего. Сихронизация обеспечивает правильную работу без ошибок. Операнды заносятся в локальную память процессорного элемента. Команда из устройства управления сначала выбирается из памяти, потом осуществляется декодирование команды ее полей, которые содержат в себе местоположение операндов и кода операции (для логических и арифметических команд). По окончании декодирования начинается выполнение команды в арифметико логическом устройстве. Полученный результат записывается в локальную память, после чего его можно передать на другой процессорный элемент для дальнейших вычислений (рис. 13).

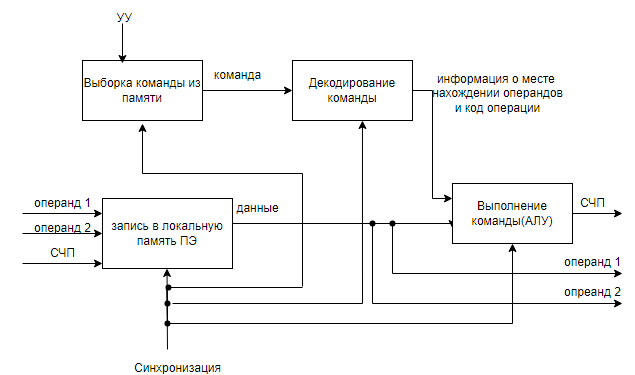


Рисунок 13 - Функциональная диаграмма детализированного уровня

## 

## 2.3 Диаграммы потоков данных

Детализированная диаграмма потоков данных демонстрирует, как каждый процесс преобразует свои входные данные в выходные, а также выявляет отношения между этими процессами. На данной диаграмме показано движение данных в процессорном элементе, показывают какие данные берутся в первую очередь, а именно сама команда, а дальше из нее считываются адреса операндов и действие над нами.

Благодаря диаграмме потоков данных можно проследить взаимосвязь между входными и выходными данными, которые будут проходить так же во всем остальных процессорных элементах (рис. 14).

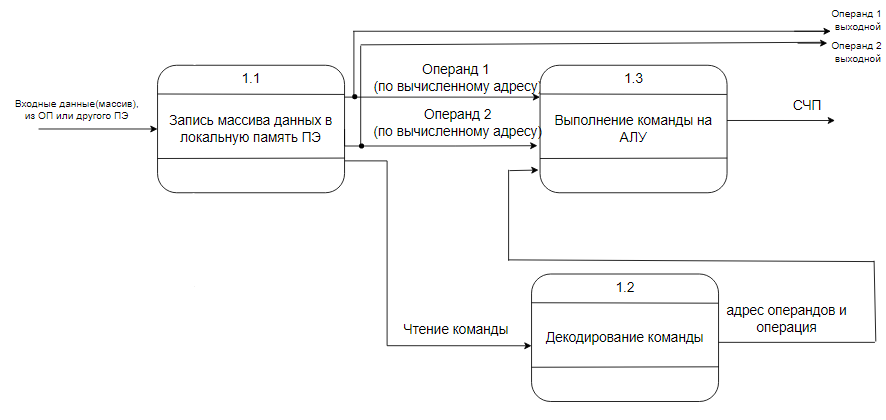


Рисунок 14 - Детализированная диаграмма потоков данных

**2.4 Схема включения с другими ПЭ.**

ПЭ могут соединяться между собой различными способами. Есть основные виды:

Линейная структура – ПЭ расположены последовательно. Линейная структура изображена на Рисунке 14.

Прямоугольная структура - ПЭ расположены в виде матрицы Рисунок 15.

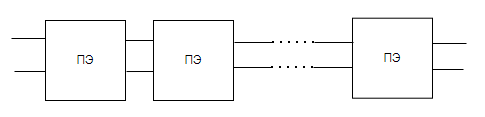


Рисунок 14 – линейная стуктура систолической системы

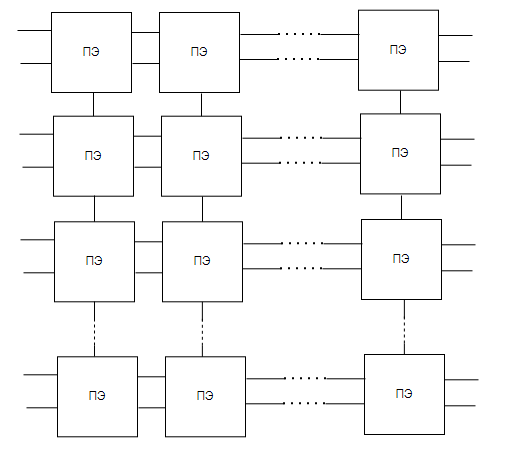


Рисунок 15 – прямоугольная структура систолической системы

## 2.5 Выводы по разделу 2

В данном разделе рассмотрено, как потоки данных переходят между процессорными элементами и оперативной памятью. Из оперативной памяти данные приходят только в первый процессорный элемент, а записываются в ОП только из последнего.

# 3. Реализация функциональной схемы

В данном разделе описывается функциональная схема и компоненты из которых она состоит. Так же показана методика работы с процессором.

## 3.1. Определение взаимодействия узлов и блоков процессора

Систолические архитектуры возникли из желания совместить преимущества конвейерной и матричной обработки. Первоначально систолические архитектуры разрабатывались для узкоспециализированных ВС, однако в дальнейшем оказалось возможным найти подходящие алгоритмы для достаточно широкого класса задач, позволяющие реализовать принципы систолической обработки.

Основная идея систолической обработки заключается в том, чтобы выполнить все стадии обработки каждого элемента данных, извлеченного из памяти, прежде чем вновь поместить в память результат этой обработки. Этот принцип реализуется систолической матрицей ПЭ, в которой отдельные ПЭ объединены между собой прямыми и регулярными связями, образующими конвейеры. По этим конвейерам как бы прокачиваются операнды, т.е. каждый элемент данных извлекается из памяти и ритмически продвигается по матрице ПЭ прежде, чем опять попадет в память. Таким образом, может формироваться несколько потоков данных, каждый из которых образован исходными операндами (элементами структуры данных, хранящейся в памяти), промежуточными результатами, получаемыми при выполнении элементарных операций в каждом ПЭ, и элементами результирующей структуры. Потоки данных синхронизированы единой для всех ПЭ системой тактовых сигналов. Во время тактового интервала все ПЭ выполняют короткую неизменную последовательность команд. Есть существенные плюсы:

1. Минимальное обращение в память, что повышает производительность;
2. Меньше операций ввода-вывода, значит меньше конфликтов при обращении в оперативную память;
3. Минимизируются связи между процессорными элементами за счет регулярности потоков данных и управляющих сигналов.

Выше рассматривалось как процессорные элементы работают между собой и по каким алгоритмам. А отдельно взятый процессорный элемент необходимо должным образом разработать так, что бы он выполнял элементарные действия, имел локальную память и правильно передавать данные в зависимости от строения системы.

При проектировании процессорного элемента требуется выделить отдельно задачу устройства управления и операционного устройства. Будет общая шина данных, которая соединяет локальную память и ядро вычислений. Так же для синхронной работы ОУ и УУ требуется механизм синхронизации.

От устройства управления будет требоваться, чтобы оно считывала команду из памяти команд, для этого понадобится регистр, для хранения команды – назовем его RgC(register of command). Далее команду нужно декодировать, для этого требуется отдельный блок. Так как нам нужно в целом соединить устройство управления и операционное устройство декодирование команд будем отображать как отдельный блок и назовем его – Дешифратор команд. После декодирования становится известно какие откуда брать операнды, куда записывать результат и какая будет выполнена операция.

В операционное устройство будут поступать операнды, для этого потребуются регистры RgO1 и RgO2. Далее в блоке АЛУ будет совершаться операция из декодированной команды. Результат операции записывается через шину данных в локальную память процессорного элемента. Так же по результату формируется регистр флагов (flag register). Так же следует добавить счетчик команд для счета команд. В регистре адреса результата будет формироваться адрес, где будет храниться результат, это значение будет передано на шину адреса. Функциональная схема представлена на рисунке 17. Структура представлена на Рисунке 18.

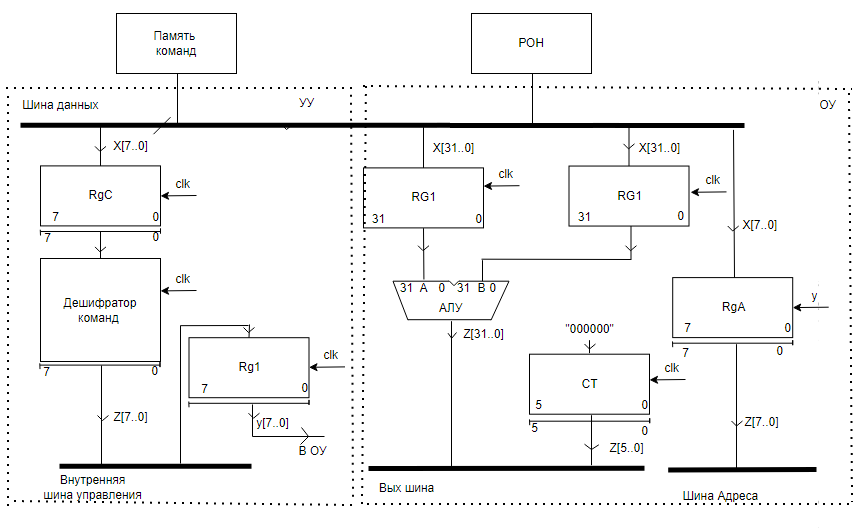


Рисунок 17 - Функциональная схема процессорного элемента

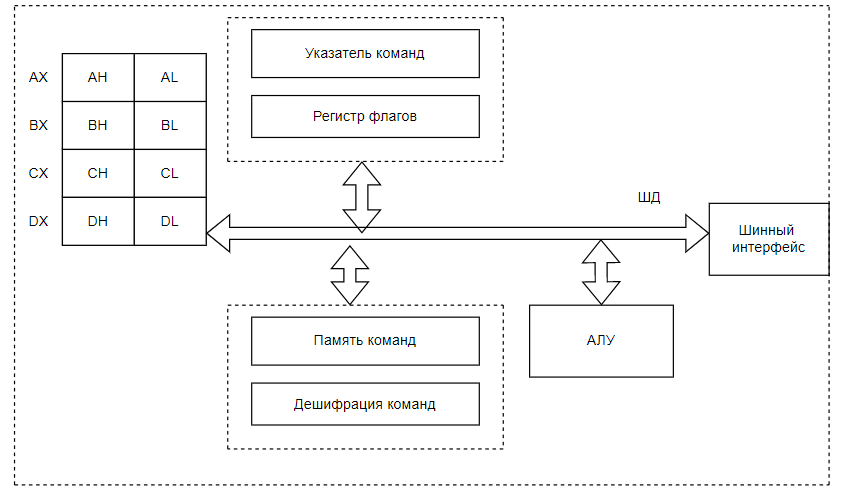


Рисунок 18 – Структура процессорного элемента

## 3.2 Описание системы команд

Требуется составить систему команд для использования процессорного элемента. Потребуется такая система команд:

1. Команды передачи данных;
2. Арифметические операции;
3. Логические операции;
4. Сдвиги;
5. Команды ввода вывода;
6. Команды прерывания;
7. Команды условного перехода;
8. Команды передачи управления;
9. Безусловный переход.

Описание групп системы команд и их команды:

1. Команды передачи данных – перепись информации из одного блока в другой. Данной группе будет одна команда: MOV приемник, источник. Данные должны быть одного размера, команда присваивает значение операнда источника операнду приемника.
2. Арифметические операции. Основные операции:

- ADD оп1,оп2 - сложение целых чисел – сложение двух операндов и запись результата в первый операнд;

- SUB оп1,оп2 - вычитание целых чисел – вычитание двух операндов и запись результата в первый операнд;

- INC/DEC оп1 – инкремент и декремент значения операнда;

- MUL оп1 – умножение операнда и содержимого регистра AX и запись в регистр AX.

- DIV оп1 – деление операнда и содержимого регистра AX и запись в регистр AX.

- CMP оп1, оп2 – команда сравнения, по его результату устанавливаются флаги, по которым можно устанавливать условия.

3. Логические команды. Основные операции:

- AND оп1, оп2 – логическое умножение;

- OR оп1, оп2 – логическое сложение;

- XOR оп1, оп2 – исключающее или, сложение по модулю 2;

- NOT оп1 – инверсия;

4. Сдвиговые команды.

- SHR оп1 – логический сдивг вправо;

- SHR оп1,cl – логический сдивг вправо на cl разрядов;

- SHL оп1 – логический сдивг влево;

- SHL оп1,cl – логический сдивг влево на cl разрядов;

- ROR оп1 – циклический сдивг вправо;

- ROL оп1, – циклический сдивг влево;

5. Команды ввода-вывода

- IN оп1, оп2 – ввод из порта (пример, IN AL, c8 – AL = port byte);

- OUT оп1, оп2 – вывод из порта (пример, IN c8,AL – AL = port byte);

6. Команды прерываний.

INT c – программное прерывание. С – номер прерывания.

IRET – возврат из обработчика прерываний.

7. Команды условного перехода.

- JB метка – переход на метку, если при CMP оп1, оп2 выполнится условие: оп1 < оп2.

- JBE метка - переход на метку, если при CMP оп1, оп2 выполнится условие: оп1 <= оп2.

- JA метка – переход на метку, если при CMP оп1, оп2 выполнится условие: оп1 > оп2.

- JAE метка – переход на метку, если при CMP оп1, оп2 выполнится условие: оп1 >= оп2.

8. Команды передачи управления.

CALL функция – вызов процедуры.

RET – возврат из процедуры.

9. Команда безусловного перехода.

- JMP метка – переводит на указанную метку.

## 3.3. Методика работы с моделью процессора

Взаимодействие с пользователем происходит посредствам пользовательского интерфейса, который будет реализован в виде оконного приложения, написанным на языке программирования C#.

При открытии приложении должно быть меню, в котором можно сохранить промежуточный результат, создать новый файл, открыть результат, и выход из приложения. Данные функции доступны при нажатии на файл.

Для редактирования команд потребуются функции копирования, вставки, а так же удалить. Данные функции будет находиться в пункте Правка.

В пункте задание пользователь получает свое задание, которое ему нужно будет сделать.

В пункте помощь требуются методические указания к выполнению лабораторной работы, схемы устройств, описание установки.

Текущее задание находится на в блоке задание, в котором показано, что нужно сделать пользователю. В блоке флаги будут показаны флаги данных. Так же на странице будут показаны Регистры общего назначения (РОН). Полученный макет представлен в Приложении В.

## 3.2 Выводы по разделу 3

В данном разделе рассмотрены блоки функциональной схемы, и как они функционируют между собой. Так же получена функциональная схема.

# Заключение

В процессе выполнения курсового проекта был разработан процессорный элемент систолической системы.

В ходе выполнения проекта выполнены основные задачи, которые необходимо было реализовать, а именно понять как и откуда будут поступать данные в процессорный элемент, где данные будут храниться и использоваться, а так же как эти данные будут передаваться другим процессорным элементам, либо записываться в операционную память. Определены команды и операции, которые сможет выполнять процессорный элемент.

В дальнейшем планируется объединить несколько процессорных элементов для составления систолической системы которая будет решать поставленную задачу. При этом организуя различные структуры системы, такие как линейная, прямоугольная, гексагональная. Расширением может так же служить добавление новых операций над массивами чисел или символов.

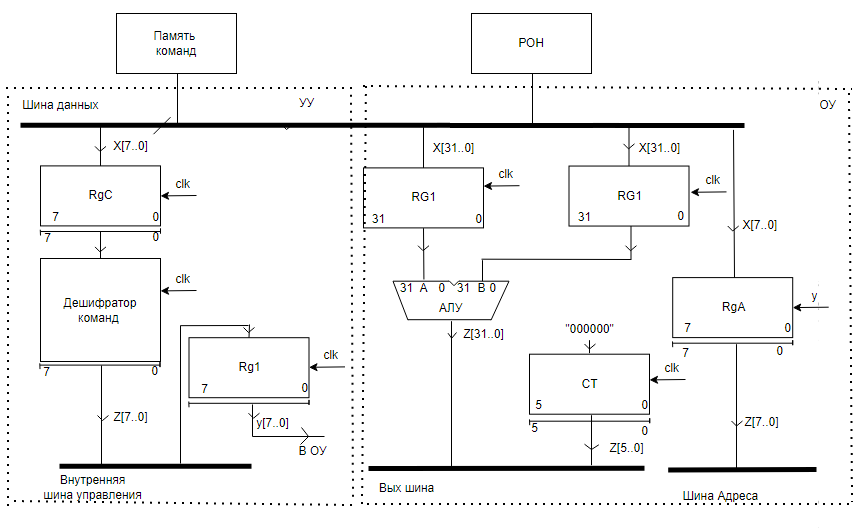
Список используемой литературы

1. Studfile.net[Электронный ресурс]. – Режим доступа: <https://studfile.net/preview/6266279/page:7/>
2. Fpga-e.ru[Электронный ресурс]. – Режим доступа: <https://fpga-e.ru/design/sistolicheskie-kih/>
3. scatspb.ru[Электронный ресурс]. – Режим доступа: <https://scatspb.ru/protsessor-intel-xeon-5430/>

Приложение А

(обязательное)

Функциональная схема



Приложение Б

(обязательное)

Макет интерфейса пользователя

