|  |  |
| --- | --- |
| Изображение выглядит как текст, эмблема, герб, нашивка  Автоматически созданное описание | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет имени Н.Э. Баумана**  **(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)

КАФЕДРА «Информационная безопасность» (ИУ8)

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

**«Реализация устройства кодирования данных с помощью кода Хаффмана»**

|  |  |
| --- | --- |
| Студент: группа ИУ8-62 л.д. 20У582 Лагов Сергей Павлович | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |
| Руководитель курсовой работы: Ковынёв Николай Витальевич | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой ИУ-8

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.А.Басараб

(И.О.Фамилия)

«17» февраля 2023 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине Электроника и схемотехника

Студент группы ИУ8-62

Лагов С.П.

(Фамилия, имя, отчество)

Тема курсовой работы Реализация устройства кодирования данных с помощью кода Хаффмана.

Направленность КР (учебная, исследовательская, практическая, производственная, др.)

учебная

Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения КР: 25% к 4 нед., 50% к 7 нед., 75% к 10 нед., 100% к 14 нед.

***Техническое задание реализовать устройство кодирования данных с помощью кода Хаффмана. На вход подаются данные в формате ASCII. Длина входной последовательности данных не превышает 16 байт. Входной алфавит ограничен 10-ю символами. Индикация результата осуществляется с помощью светодиодов.***

***Оформление курсовой работы:***

Расчетно-пояснительная записка на 20 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

1.Схема электрическая функциональная 2.Схема электрическая принципиальная

Дата выдачи задания «17» февраля 2023 г.

**Руководитель курсовой работы**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Ковынёв

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** С.П. Лагов

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

АННОТАЦИЯ

В курсовой работе выполняется проектирование устройства, кодирующего данные с помощью кода Хаффмана.

Цель курсовой работы – рассчитать элементы микросхемы устройства кодирования данных кодом Хаффмана, получить опыт в разработке микросхем подобного типа.

Результатом работы является принципиальная схема микросхемы, содержащая номинальные значения всех пассивных и активных элементов, которые обеспечивают правильную работу схемы устройства в соответствии с заданными требованиями.

Работа включает в себя схемы, рисунки, технические данные по используемым элементам.

СОДЕРЖАНИЕ

АННОТАЦИЯ 3

СОДЕРЖАНИЕ 4

ВВЕДЕНИЕ 5

ОСНОВНАЯ ЧАСТЬ 7

1 Описание структурной схемы устройства 7

2 Составление и расчёт схемы устройства 10

2.1 Составление функциональной схемы блока управления ОЗУ 10

2.2 Составление функциональной схемы блока ввода 13

2.3 Составление функциональной схемы блока сортировки 15

2.4 Составление функциональной схемы блока построения дерева 20

2.5 Составление функциональной схемы блока получения кода 25

2.6 Составление функциональной схемы блока вывода 26

ВВЕДЕНИЕ

Кодирование информации — процесс преобразования сигнала из формы, удобной для непосредственного использования информации, в форму, удобную для передачи, хранения или автоматической переработки.

Реализация устройства кодирования данных с помощью кода Хаффмана — это актуальная задача, поскольку кодирование Хаффмана является одним из наиболее популярных методов сжатия данных. К другим преимуществам этого метода относятся простота реализации и высокая скорость работы. Разработка такого устройства может применяться во многих областях, включая телекоммуникации, обработку и хранение данных, медицинскую технику и т. д.

Алгоритм Хаффмана — жадный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью. Был разработан в 1952 году аспирантом Массачусетского технологического института Дэвидом Хаффманом при написании им курсовой работы. В настоящее время используется во многих программах сжатия данных.

Классический алгоритм Хаффмана на входе получает таблицу частотностей символов в сообщении. Далее на основании этой таблицы строится дерево кодирования Хаффмана (Н-дерево).

1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.
2. Выбираются два свободных узла дерева с наименьшими весами.
3. Создается их родитель с весом, равным их суммарному весу.
4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.
5. Одной дуге, выходящей из родителя, ставится в соответствие бит 1, другой — бит 0. Битовые значения ветвей, исходящих от корня, не зависят от весов потомков.
6. Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

Результатом работы будет являться спроектированное устройство, аппаратно реализующее приведенный алгоритм и кодирующее последовательность 8-битных ASCII-символов размером не более 16 и мощностью алфавита не более 10 с последовательным выводом кода. Индикация кода будет производиться на двухцветных диодах.

ОСНОВНАЯ ЧАСТЬ

1. Описание структурной схемы устройства

Структурная схема предназначена для описания принципа работы устройства и его состава в общем виде. На схеме изображаются все основные функциональные части изделия, а также основные взаимосвязи между ними. Построение такой схемы даёт наглядное представление о совместном взаимодействии функциональных узлов.

На схеме, изображенной на рисунке 1, следующие функциональные части:

1. Блок ввода данных — часть микросхемы, производящая запись входной последователь.
2. Блок сортировки данных — часть микросхемы, преобразующая данные в отсортированный по частотности по возрастанию массив пар символ-частотность.
3. Блок построения кодового дерева — часть микросхемы, структурно связывающая данные между собой в кодовое дерево Хаффмана.
4. Блок построения кода — часть микросхемы, производящая сопоставление символ-код на основе построенного дерева.
5. Блок вывода данных — часть микросхемы, реализующая последовательный вывод итогового кода в соответствии с определенным правилом.
6. ОЗУ — память устройства.
7. Блок управления ОЗУ — часть микросхемы, отвечающая за выбор источника адреса, вводимых данных, а также команд управления ОЗУ среди функциональных частей 1 — 6 на основе флагов состояния работы частей.

Изображение выглядит как текст, диаграмма, линия, План

Автоматически созданное описание

Рисунок 1 – Структурная схема кодирующего устройства

Устройство имеет управляющие сигналы: C, AE, R, S, D0…D7.

Таблица 1 — Таблица соответствий входных сигналов устройства

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| C | AE | R | S | Действие |
| 0 | X | X | X | — |
| 1 | 0 | 0 | 0 | — |
| 1 | 0 | 0 | 1 | Начать работу схемы |
| 1 | 0 | 1 | 0 | Сброс |
| 1 | 0 | 1 | 1 | Запрещенная комбинация |
| 1 | 1 | 0 | 0 | Добавить символ D0…D7 |
| 1 | 1 | 0 | 1 | Запрещенная комбинация |
| 1 | 1 | 1 | 0 | Запрещенная комбинация |
| 1 | 1 | 1 | 1 | Запрещенная комбинация |

Блок устройства начинает работу в тот момент, когда предыдущий блок выставил бит, сигнализирующий о конце работы. Все блоки взаимодействуют с ОЗУ устройства. Все блоки могут читать данные на выходах ОЗУ. Доступ к записи в память блоки имеют только в тот момент, когда они находятся в работе.

Блок управления (БУ) ОЗУ от каждого блока получает адрес ОЗУ, данные для записи, а также сигналы управления запоминающим устройством. БУ подаёт на ОЗУ сигналы, основываясь на текущих флагах состояния работы остальных блоков.

В работе участвуют 5 логических блоков: блок ввода (INPUT), блок сортировки (этот блок для удобства разделен на 2 подблока — SORT-1, SORT-2), блок построения дерева (TREE), блок получения кода (CODE), блок вывода (OUTPUT). В таблице приведена таблица истинности, в соответствии с которой реализуется выбор подаваемых блоками в БУ сигналов для управления ОЗУ.

Таблица 1 — Таблица истинности управляющих сигналов для блока управления ОЗУ

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| OUTPUT | CODE | TREE | SORT-2 | SORT-1 | INPUT | Y3 | Y2 | Y1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |

Других комбинаций флагов работы появиться не может, так как логические единицы выставляются последовательно на каждом из них в вышеуказанном порядке. INPUT всегда обозначен как 1, так как при сбросе на нем единственном не меняется бит состояния. Для того, чтобы давать возможность писать в ОЗУ 5-ю блокам, на БУ должен подаваться 3-битный адрес этого блока.

Заметим, что максимальная длина кода символа при размере последовательности до 16 и мощности алфавита до 10 символов не может быть больше 4. Таким образом можно дать оценку сверху числа светодиодов для индикации вывода — 16 × 4 = 64. Значит, индикация будет производиться на сетке светодиодов 8×8.

1. Составление и расчёт схемы устройства

Задачу получения принципиальной схемы устройства можно разделить на несколько частей:

1. Составление функциональной схемы блока управления ОЗУ
2. Составление функциональной схемы блока ввода
3. Составление функциональной схемы блока сортировки
4. Составление функциональной схемы блока построения дерева
5. Составление функциональной схемы блока получения кода
6. Составление функциональной схемы блока вывода
7. Объединение составленных элементов
8. Расчет элементов микросхемы
9. Выбор реальных элементов для схемы

#### **Составление функциональной схемы блока управления ОЗУ**

Изображение выглядит как текст, снимок экрана, диаграмма, График

Автоматически созданное описание

Рисунок 2

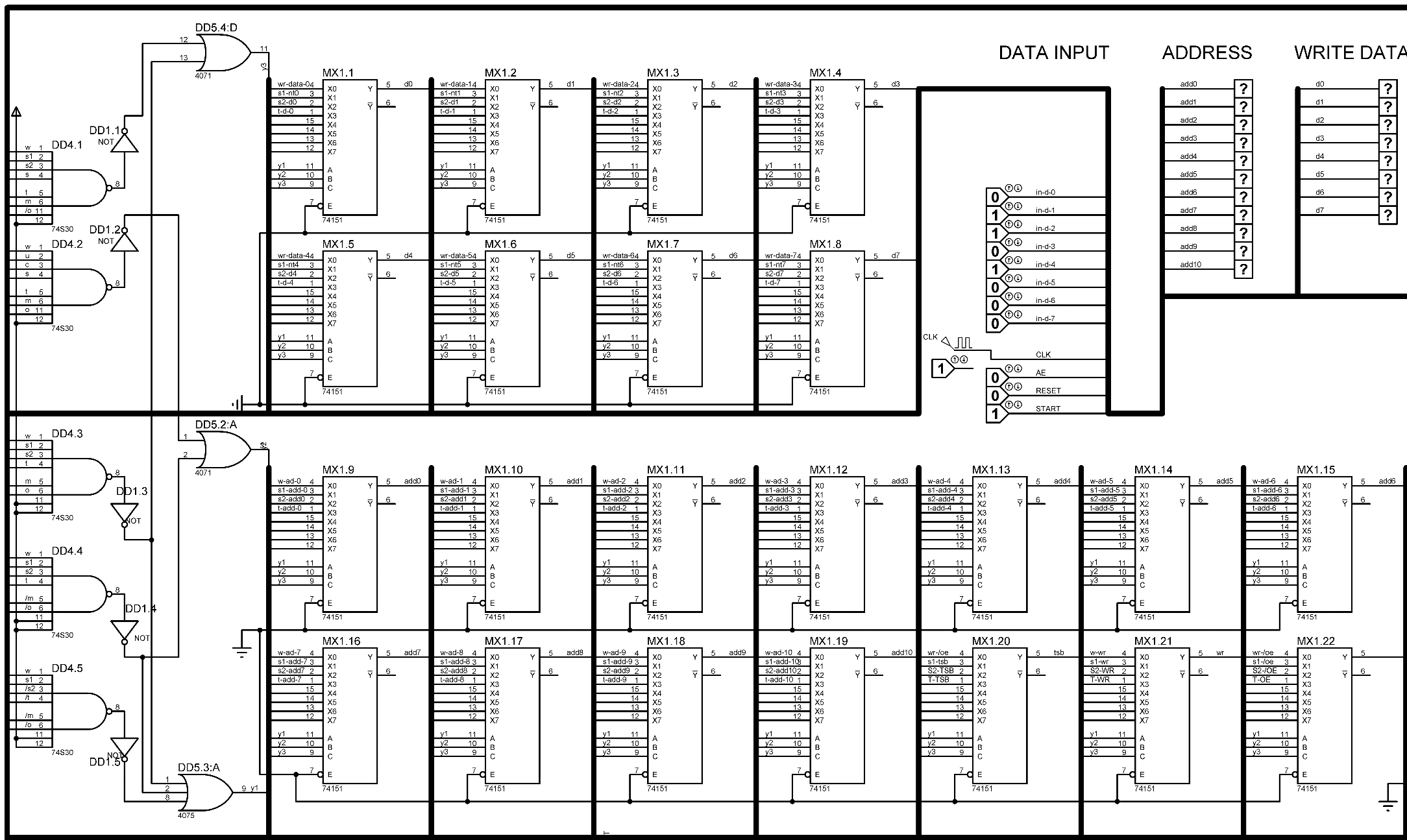


Рисунок 3 — Схема блока управления с ОЗУ

В процессе работы право на запись и чтение в оперативно запоминающее устройство должен переходить последовательно от блока к блоку. В работе помимо блока управления задействовано 6 блоков.

Выбор адреса ОЗУ и записываемых по этому адресу данных осуществляется на двух секциях из 10-и и 8-и 7-канальных одноразрядных мультиплексоров. Управляющие мультиплексорами сигналы — Y1, Y2, Y3.

Изображение выглядит как диаграмма, текст, линия, Параллельный

Автоматически созданное описание

Рисунок 4

#### **Составление функциональной схемы блока ввода**

Для удобства хранения и использования данных была выбрана статическая ОЗУ 1Кx8 с 10-разрядными адресами ячеек.

ОЗУ логически разделена на 16 секций. Адрес ячеек памяти логически разделён на 3 секции. Он имеет вид в формате Big-endian (сначала старшие разряды) LLLLYYYYXX, X, Y, L ∈ {0, 1}. Разряды LLLL показывают, к какому слою принадлежит ячейка (0000 — 0-му, 0001 — 1-му, 1110 — 14-му, участки памяти интерпретируются как слои в связи с нуждой в дальнейшем строить кодовое дерево Хаффмана). YYYY аналогично указывают на определенный участок памяти в слое, дополнительно имеющий 4 ячейки (поля) по 8 бит (2 бита XX).

При высоком уровне AE и фронте на C данные на информационных входах устройства D0…D7 записываются во временный регистр, а на спаде такта — в ОЗУ в 0-й слой по адресу 0000IIII00, где IIII — количество уже записанных символов в двоичном виде, поступающее со счетчика. Количеством записанных в память символов считается число на информационных выходах счетчика. По достижении максимального их количества в 16 единиц оно более не увеличивается, не вызывая переполнение и обнуление младших разрядов. В таком случае все символы, чьи порядковые номера превышают 16, будут циклично перезаписывать уже хранящиеся в памяти символы, начиная с первого. Счетчик записанных слов обнуляется только в случае фронта на тактирующем входе при высоком уровне R.

Начало процесса обработки данных должно быть сопровождено фронтом на тактирующем входе устройства с поднятым уровнем на входе S. После этого S следует перевести в уровень логического нуля.

Изображение выглядит как диаграмма, План, Технический чертеж, схематичный

Автоматически созданное описание

Рисунок 5

#### **Составление функциональной схемы блока сортировки**

В целях оптимизации построения кодового дерева заранее отсортируем записанные символы по возрастанию, взяв в качестве ключа сортировки их абсолютную частотность появления во входной последовательности.

В данной реализации в целях упрощения схемы устройства сортировка данных будет выполняться только один раз перед построением кодового дерева. В процессе его построения полная сортировка как таковая не потребуется.

Для упорядочивания данных будет использован алгоритм сортировки подсчётом. Он будет логически разделен на 2 этапа:

1. Накапливание абсолютной частотности символа в ячейке, адресом которой является этот символ
2. Последовательное выписывание в упорядоченный массив всех элементов с учётом их абсолютной частотности

ASCII-символы кодируются 8-ю битами. Big-endian адрес ячейки, в которой в концу 1-го этапа сортировки будет храниться число её вхождений с исходную последовательность, будет иметь вид 11-SSSSSSSS, S ∈ {0, 1}. 8 битов S — 8 битов символа, для которого ищется его абсолютная частотность. Таким образом 4 логических слоя (с 12 по 15) будут использованы для хранения информации о не более чем 10 возможных разных символах, которые могли быть записаны пользователем.

Внешний цикл итерируется по 0-му слою, для каждого символа выполняя следующие операции:

1. Чтение ASCII-символа
2. Чтение из ОЗУ по 8-битному адресу (код ASCII-символа) уже записанного количества его вхождений в последовательность (изначально равно 0)
3. Запись в ОЗУ по тому же адресу инкрементированного числа вхождений символа

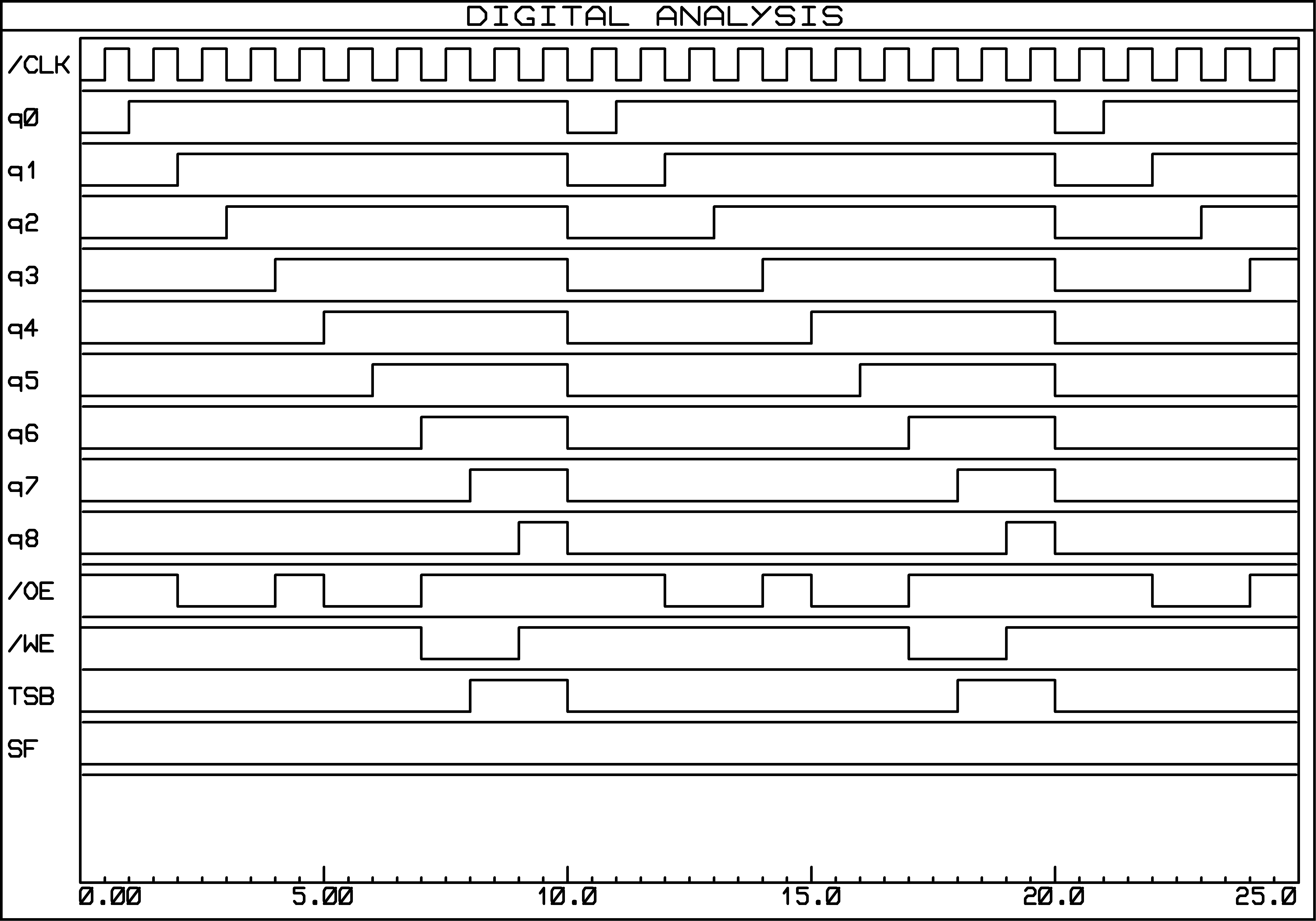


Рисунок 6

Циклическая выполнение такой последовательности команд выполняется при помощи регистра сдвига. По спаду тактового сигнала регистр совершает сдвиг влево, записывая в младший разряд 1. Таким образом постепенно все разряды регистра заполняются значениями 1. Назовем эти значения q\_i. От совокупности значений q\_i зависят сигналы /OE (выводит значение в ОЗУ по указанному адресу по спаду сигнала), /WE (записывает в ОЗУ значение на двунаправленных I/O выводах по фронту сигнала) и TSB (стробирующий сигнал ОЗУ, управляющий шинными тристабильными буферами (ключами) на выводах данных запоминающего устройства).

Исходный символ записывается в RG2.5. Количество его вхождений, прочитанное из ОЗУ, записывается в R2.3. Из сумматора DD8.1, прибавляющего 1 к прочитанному числу, инкрементированное значение выводится на мультиплексоры, контролируемые битами состояния выполнения текущей итерации цикла. В нужный момент они передают на контакты данных, исходящих из данного блока на ОЗУ, увеличенное на 1 значение, которое затем записывается в запоминающее устройство с поднятым уровнем TSB и фронтом на /WE.

После окончания 1-го этапа сортировки начинается 2-й этап. Он заключается в последовательном выписывании в массив на 1-м слое сначала символом, встречающихся 1 раз, затем тех, чьё количество вхождений составляет 2 раза и т.д. Формально алгоритм можно записать следующим образом:

1. В переменную H загрузить 0
2. В переменную N загрузить 1
3. Установить указатель I на нулевой символ записанной последовательности
4. Прочитать символ из ячейки по адресу I, записать его в регистр W
5. Прочитать из ОЗУ по адресу 11W количество вхождений символа C
6. Если C равно N, записать 0 в 11W, записать W по адресу 0001HHHH00 и C по адресу 0001HHHH01, увеличив затем H на 1
7. Если I не равно WW (words written — значение из регистра из этапа ввода) то увеличить I на 1, иначе перейти на пункт 8
8. Если N равно 10, то закончить алгоритм, иначе увеличить N на 1 и перейти на пункт 3

Изображение выглядит как диаграмма, План, Технический чертеж, схематичный

Автоматически созданное описание

Рисунок 7 — Схема блока 1-го этапа сортировки (подсчет количеств вхождений)

Изображение выглядит как диаграмма, План, Технический чертеж, зарисовка

Автоматически созданное описание

Рисунок 8 — Схема блока 2-го этапа сортировки (выпивание по возрастанию)

#### **Составление функциональной схемы блока построения дерева**

Построение кодового дерева — неотъемлемая часть построения кода Хаффмана. В открытых источниках какие-либо статьи или информация касательно проблемы аппаратной реализации построения дерева (не только кодового), а также реализации алгоритма Хаффмана найдено не было. Придумаем алгоритм с чистого листа.

Сортировка — достаточно тяжелая в аппаратной реализации алгоритм. Совершим его единожды в начале, постараемся избежать его повторного использования впредь.

Имея отсортированную по возрастанию последовательность уникальных символов, мы всегда можем гарантировать правомерность образования узла из первых 2-х (0-го и 1-го) элементов. Образовав узел, их абсолютные частотности складываются. Их значение может быть таким, что при следующем образовании узла данный узел участвовать не будет. Предлагается записать на следующий за текущим слой сперва все символы из текущего слоя с индексами от 2 до 9, чья частотность меньше суммы первых двух узлов. Затем после них дописать на следующий слой в ячейку суммарную частотность первых 2-х узлов, после чего снова пройтись по текущему слою с 2 по 9 символы, выписывая те, чья частотность больше или равна суммарной частотности первых двух узлов.

Таким образом мы формируем узел и динамически поддерживаем массив уникальных символов отсортированным по частотности.

Следует сказать, что для более простой реализации получения кодировки для каждого из символов исходного уникального массива, при записи символа/узла с его частотностью на следующий слой, необходимо в текущем слое по адресу LLLLYYYY10 (3-е из 4-х полей для каждого символа) записать адрес ячейки следующего слоя. Так мы отбрасываем необходимость реализовывать рекурсию, а будем просто переходить со слоя на слой, накапливая код Хаффмана для символа с конца.

Опишем формально этот алгоритм:

1. Загрузить 1 в переменную адреса слоя L
2. Загрузить 0 в переменную NLH (next layer head)
3. Прочитать частотность 0-го символа на слое L в переменную T0
4. Прочитать частотность 1-го символа на слое L в переменную T1
5. Если хотя бы одно из значений T0 и T1 равно 0, то закончить алгоритм (либо пользователь ничего не ввёл, либо мы находимся на последнем слое с одним узлом/ячейкой и закончили построение дерева). Записать сумму T0 и T1 в переменную S
6. Загрузить 2 в ячейку итератора по слою I
7. Если частотность символа в I ячейке слоя L равна 0, то перейти на пункт 9
8. Если частотность символа в I ячейке слоя L меньше S, то в I ячейку слоя L записать NLH (адрес ячейки следующего слоя, в которую переходит текущая ячейка), в частотность ячейки NLH слоя L+1 записать частотность ячейки I слоя L. Увеличить NLH на 1
9. Если I равна 16, то перейти на пункт 10, иначе увеличить I на 1 и перейти на пункт 7
10. В 0-ю и 1-ю ячейки слоя L в поля адреса записать NLH. В частотность ячейки NLH слоя L+1 записать S. Увеличить NLH на 1
11. Загрузить 2 в ячейку итератора по слою I
12. Если частотность символа в I ячейке слоя L равна 0, то перейти на пункт 14
13. Если частотность символа в I ячейке слоя L больше или равна S, то в I ячейку слоя L записать NLH (адрес ячейки следующего слоя, в которую переходит текущая ячейка), в частотность ячейки NLH слоя L+1 записать частотность ячейки I слоя L. Увеличить NLH на 1
14. Если I равна 16, то конец, иначе увеличить I на 1 и перейти на пункт 12

Следует упомянуть, что в силу ограниченности мощности входного алфавита сверху 10 уникальными символами, высота дерева не может превышать 9, то есть может занимать максимум 10 слоев в ОЗУ с 1-го по 10-й. Поэтому остается еще 5 неиспользованных слоев (с 11 по 15), которые можно использовать (в конце сортировки все частотности были очищены и перезаписаны нулями).

Изображение выглядит как диаграмма, План, Технический чертеж, схематичный

Автоматически созданное описание

Рисунок 9

Изображение выглядит как диаграмма, План, Технический чертеж, схематичный

Автоматически созданное описание

Рисунок 10

Изображение выглядит как диаграмма, План, Технический чертеж, схематичный

Автоматически созданное описание

Рисунок 11

#### **Составление функциональной схемы блока получения кода**

Завершение работы блока построения кодового дерева разрешает управляющему блоку передать контроль над ОЗУ блоку получения кода для каждого символа.

Устройству будет необходимо каждый такт выдавать следующий бит выходной последовательности, поэтому надо заранее подготовиться к этому, установив соответствие символ-код.

Получение кода будет начинаться с 1-го слоя дерева. Если ячейка имеет адрес 0, то запишем 0 как последний бит кода этого символа, увеличив счетчик длины кодировки на 1. Если ячейка имеет адрес 1, то проделаем аналогичные операции, записав 1 как последний бит кода символа. В случае, если адрес ячейки не равен 0 или 1, совершать дополнительные операции нет необходимости.

После этого перейдем по на следующий за текущим слой в ячейку по адресу, хранящемуся в поле адреса текущей ячейки. Аналогично проанализировав адрес ячейки (равен ли 0 или 1), продолжим выполнение алгоритма, пока не дойдем до последнего слоя дерева.

Получившийся код размером не более 4-х битов. Его длина кодируется 2-я битами. Как было описано в блоке сортировки подсчетом, запишем в ОЗУ по адресу символа 6 битов — код Хаффмана и его длину.

Проделаем эту процедуру для каждого символа 1-го слоя с ненулевой частотностью.

В итоге для каждого символа из входной последовательности будем иметь его код и длину кода, записанные в ОЗУ по адресу этого символа.

#### **Составление функциональной схемы блока вывода**

Поднимем T в уровень логической единицы.