**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)

Москва, 2023

**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.

# З А Д А Н И Е
## 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 выполнена на 37 листах и включает в себя схемы, рисунки, технические данные по используемым элементам.

# СОДЕРЖАНИЕ

# 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.

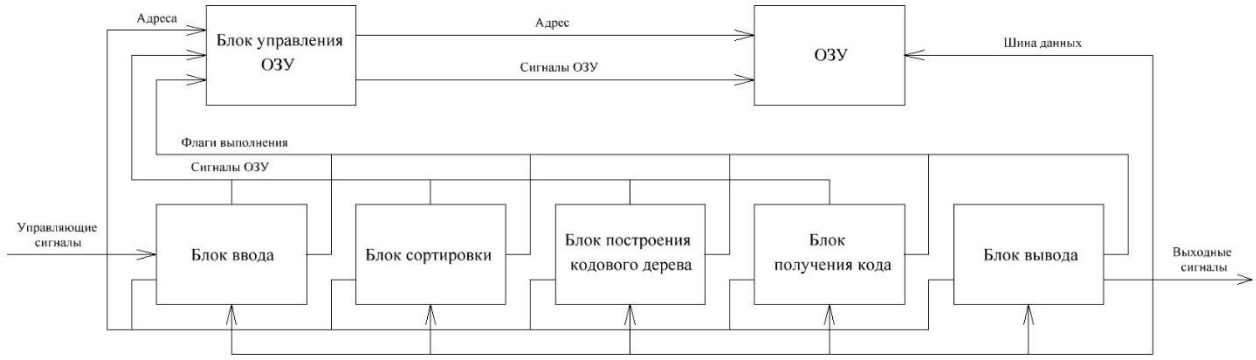


Рисунок 1 – Структурная схема кодирующего устройства

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 2 — 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 — $16 \times 4 = 64$. Значит, индикация будет производиться на сетке светодиодов 8×8.

## 2 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. Составление функциональной схемы блока индикации
8. Расчет элементов микросхемы
9. Выбор реальных элементов для схемы
10. Расчёт мощности устройства

### 2.1 Составление функциональной схемы блока управления ОЗУ

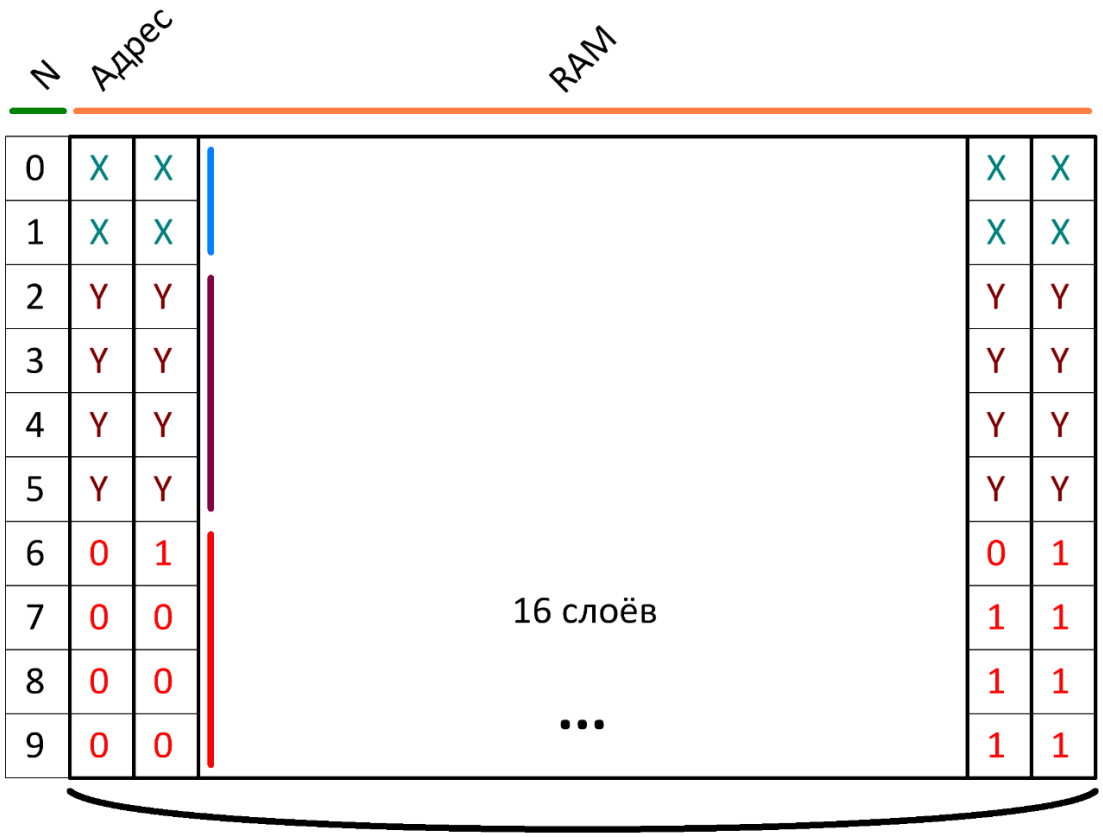


Рисунок 2 — Схема логического разбиение адресного пространства ОЗУ на слои

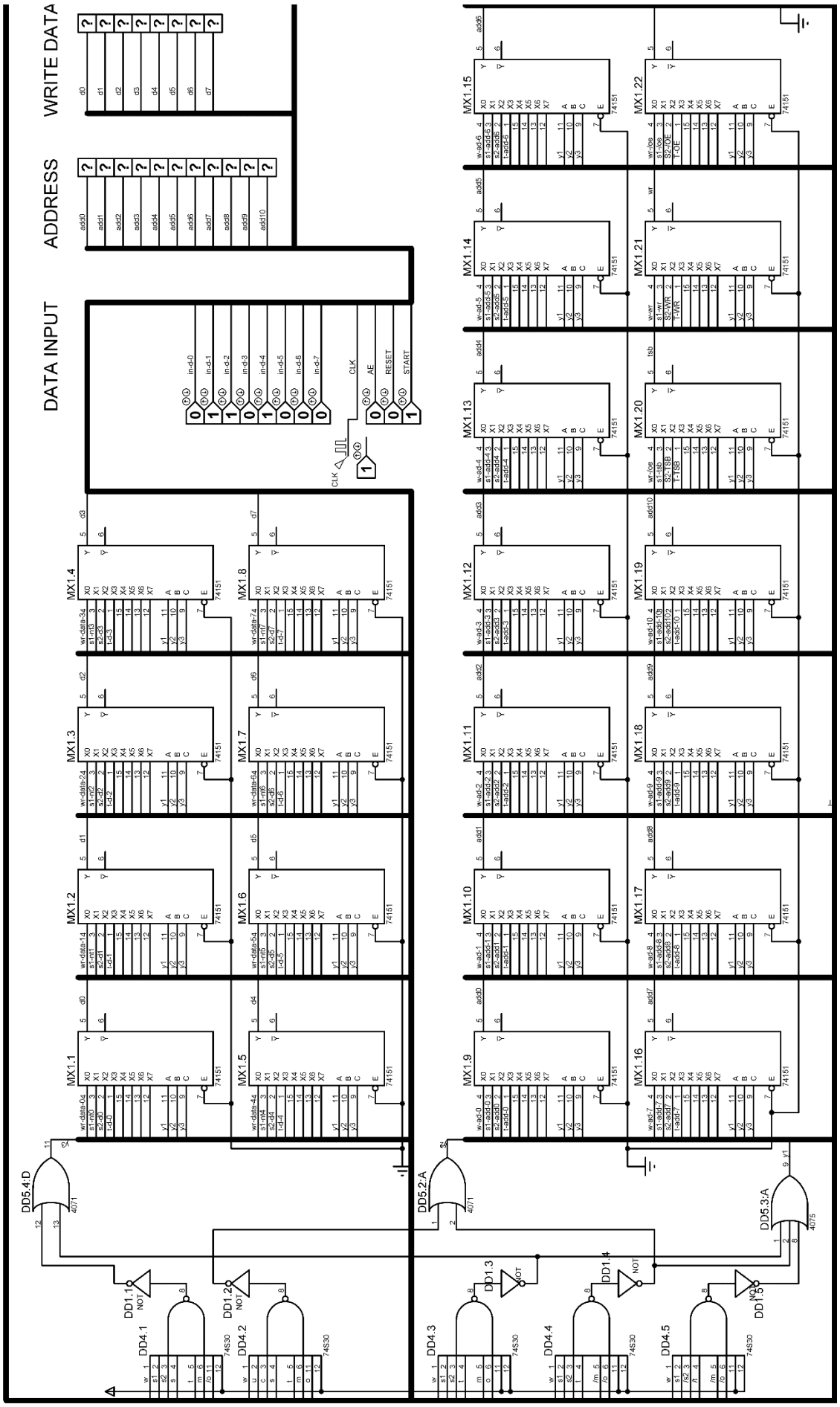


Рисунок 3 — Схема блока управления ОЗУ

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.

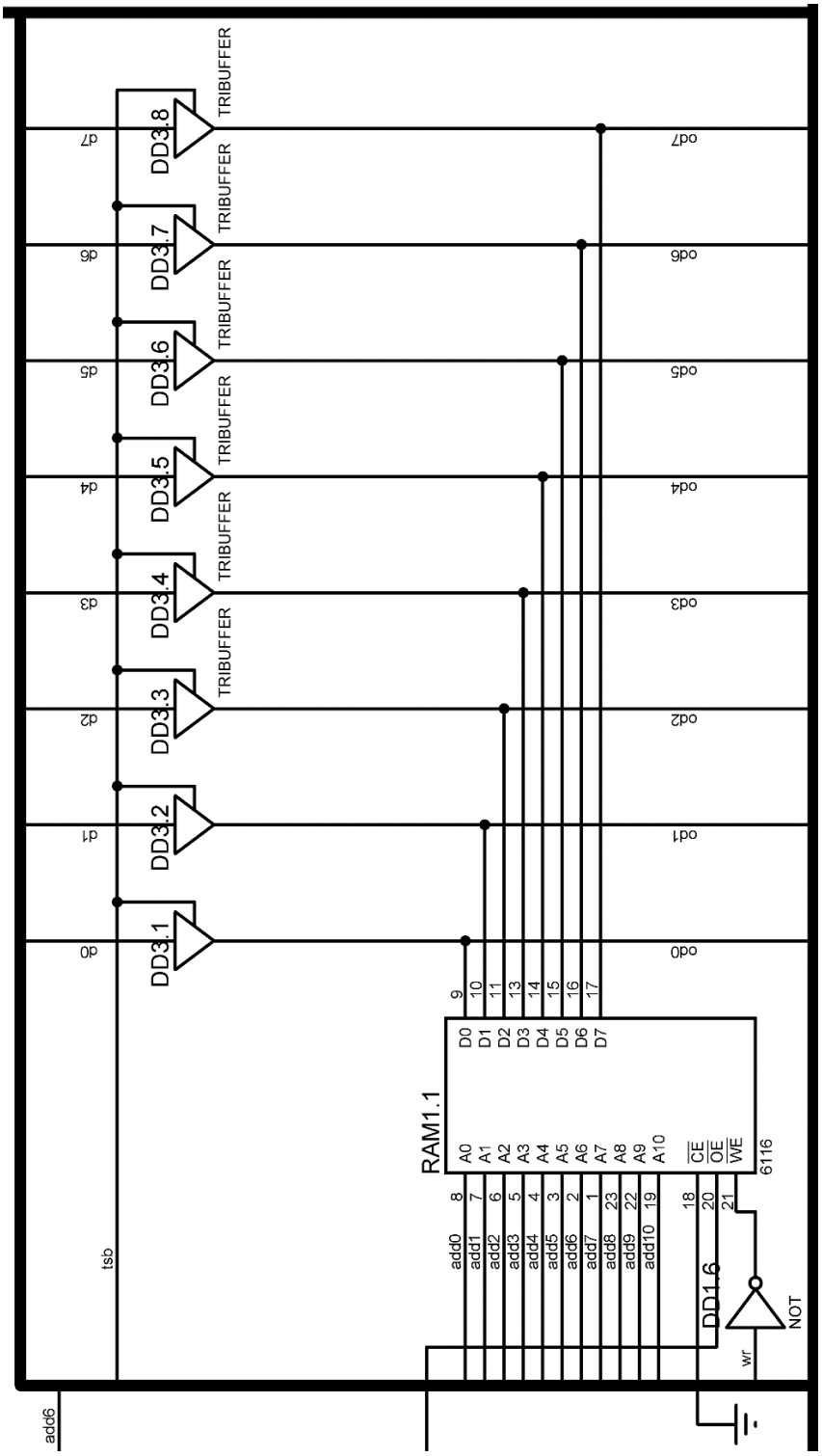


Рисунок 4 — Схема включения ОЗУ через тристабильные буферы

## 2.2 Составление функциональной схемы блока ввода

Для удобства хранения и использования данных была выбрана статическая ОЗУ 1Кх8 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.

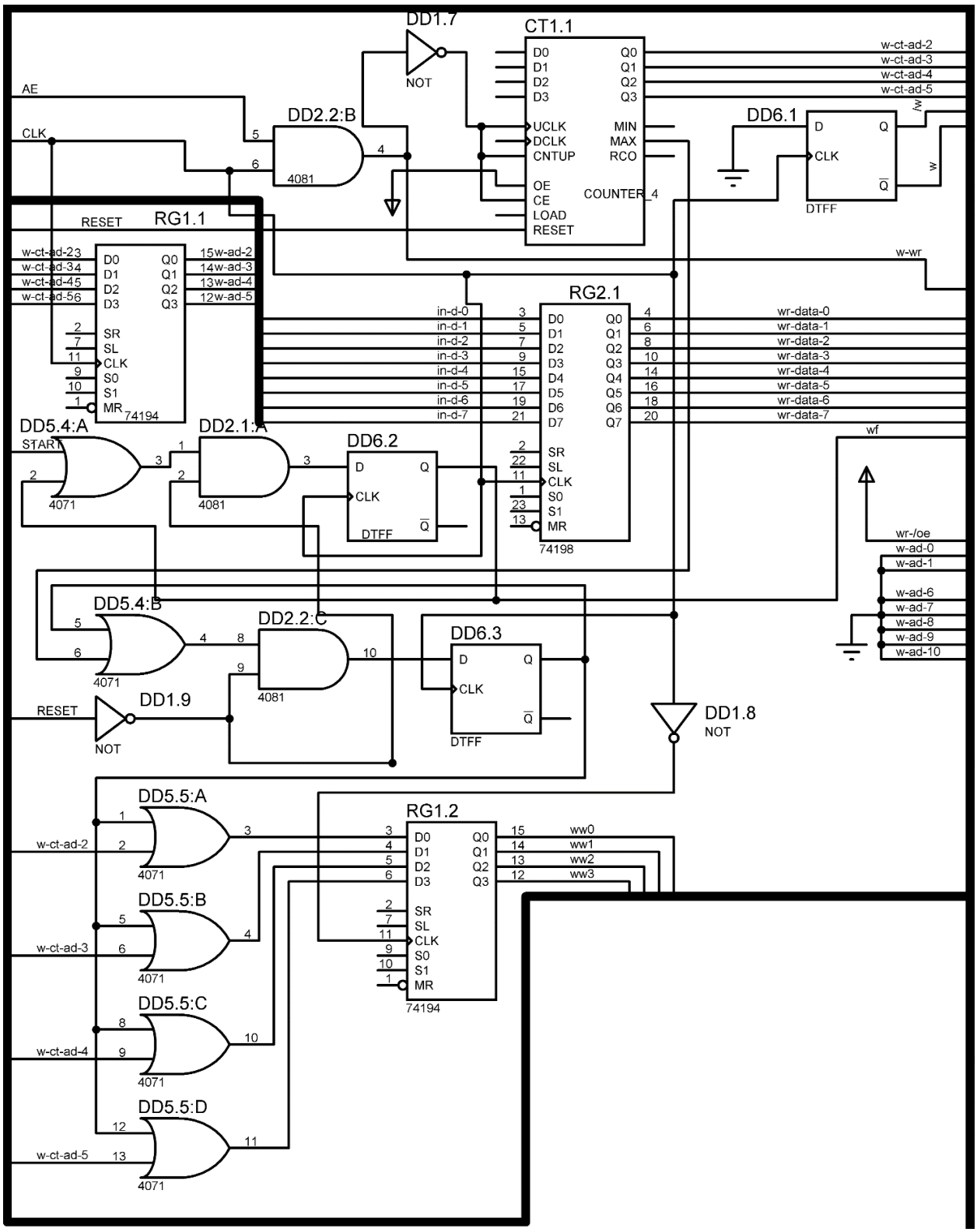


Рисунок 5 — Схема блока ввода данных

## 2.3 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, $S \in \{0, 1\}$. 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

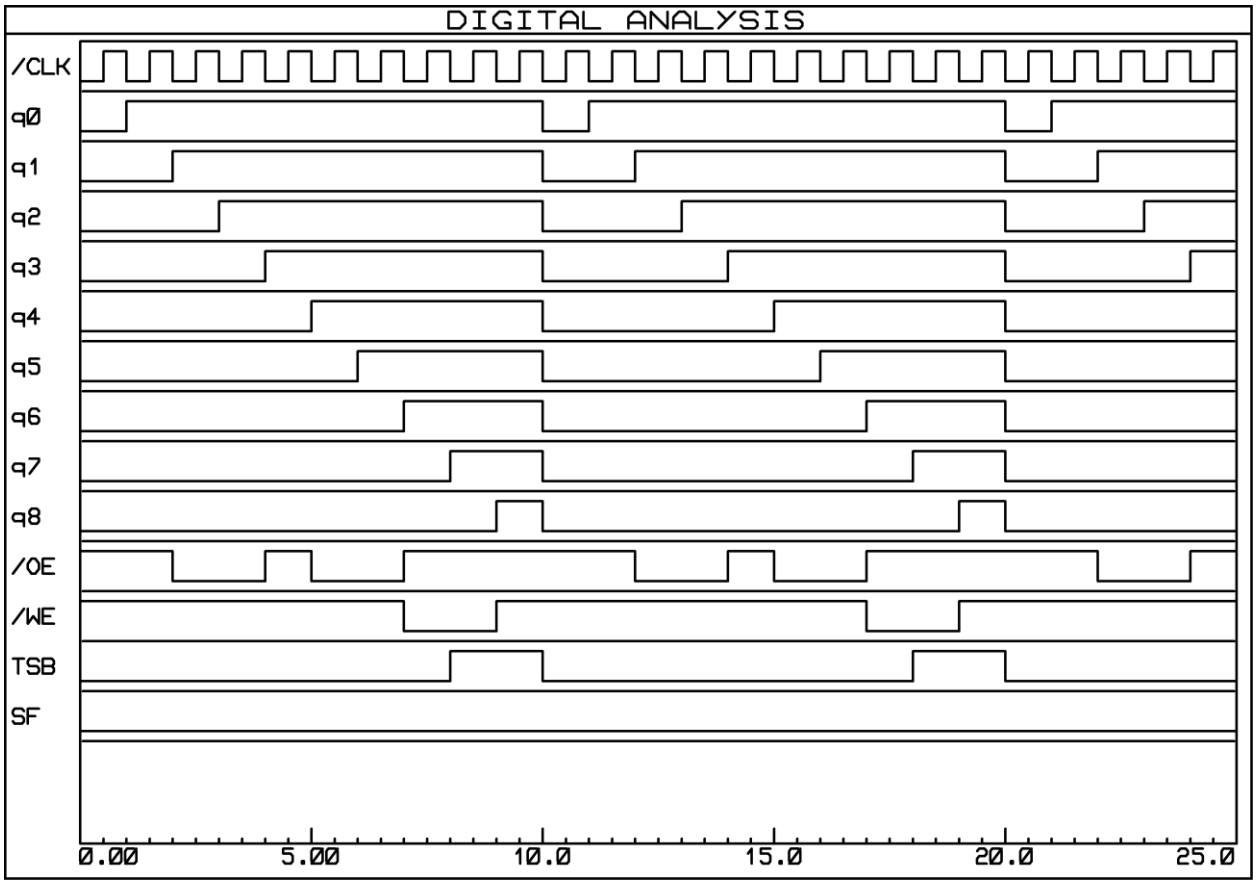


Рисунок 6 — График логических состояний и сигналов управления ОЗУ 1 этапа сортировки

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

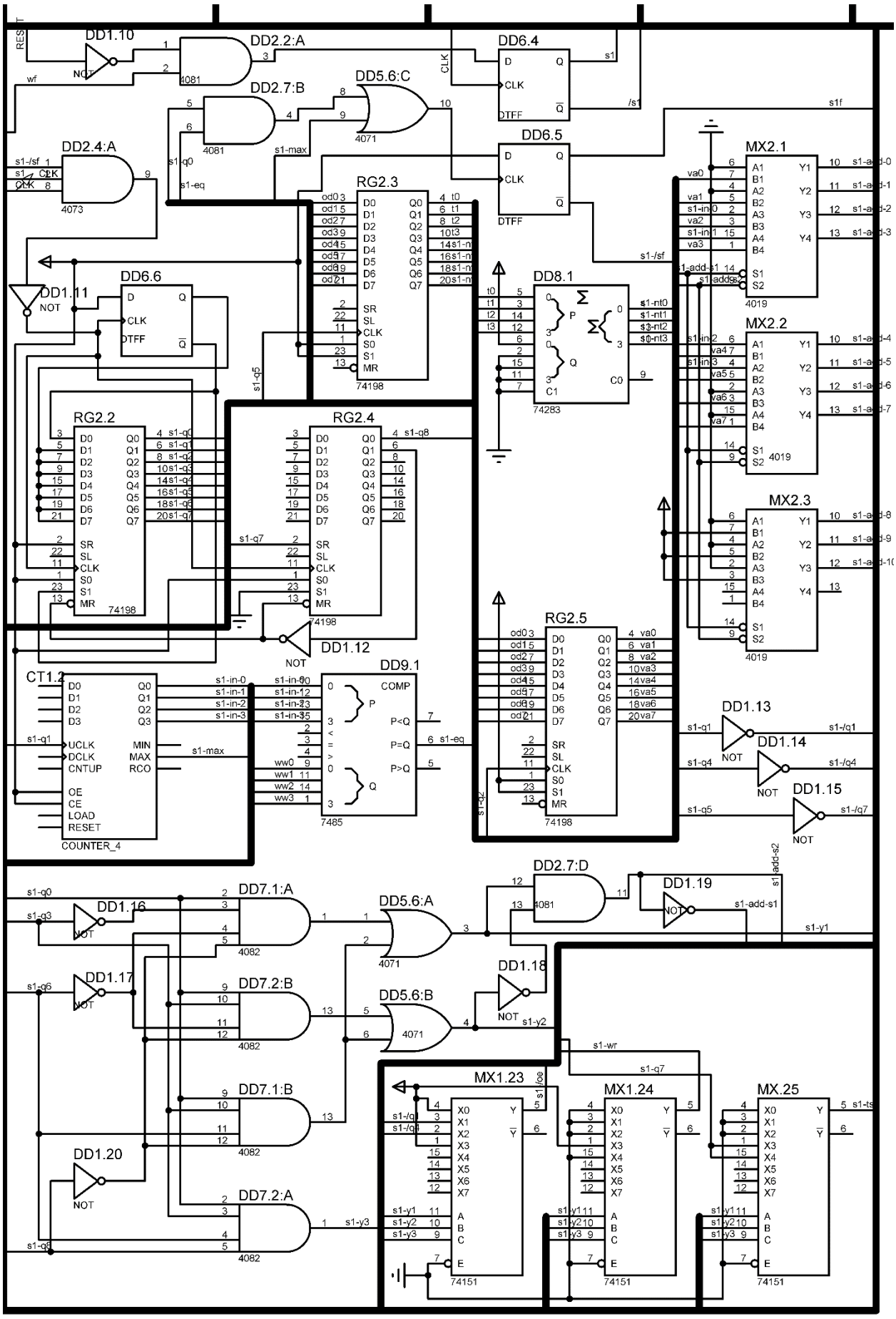


Рисунок 7 — Схема блока 1-го этапа сортировки (подсчет количеств вхождений)

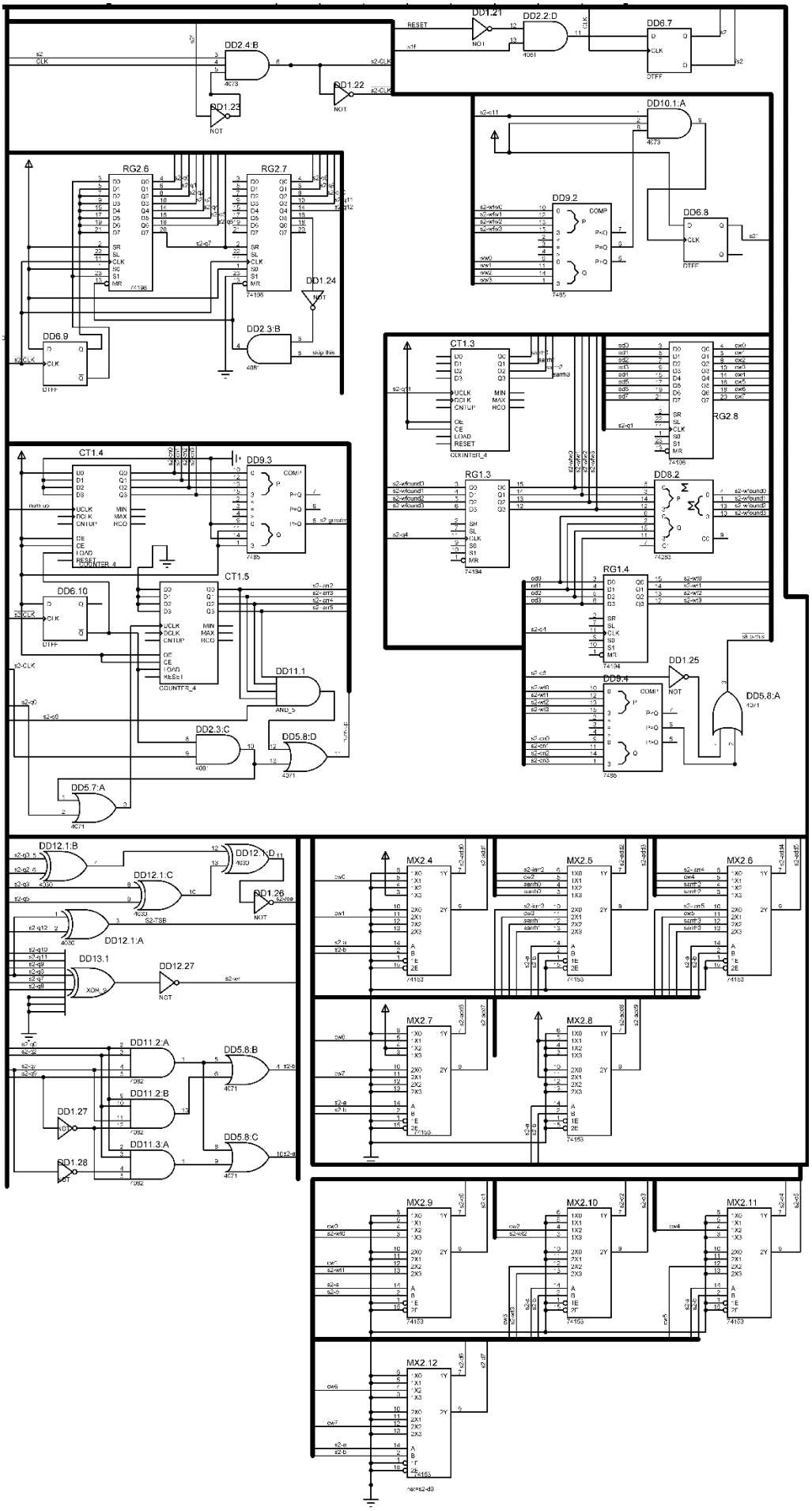


Рисунок 8 — Схема блока 2-го этапа сортировки (выписывание по возрастанию)

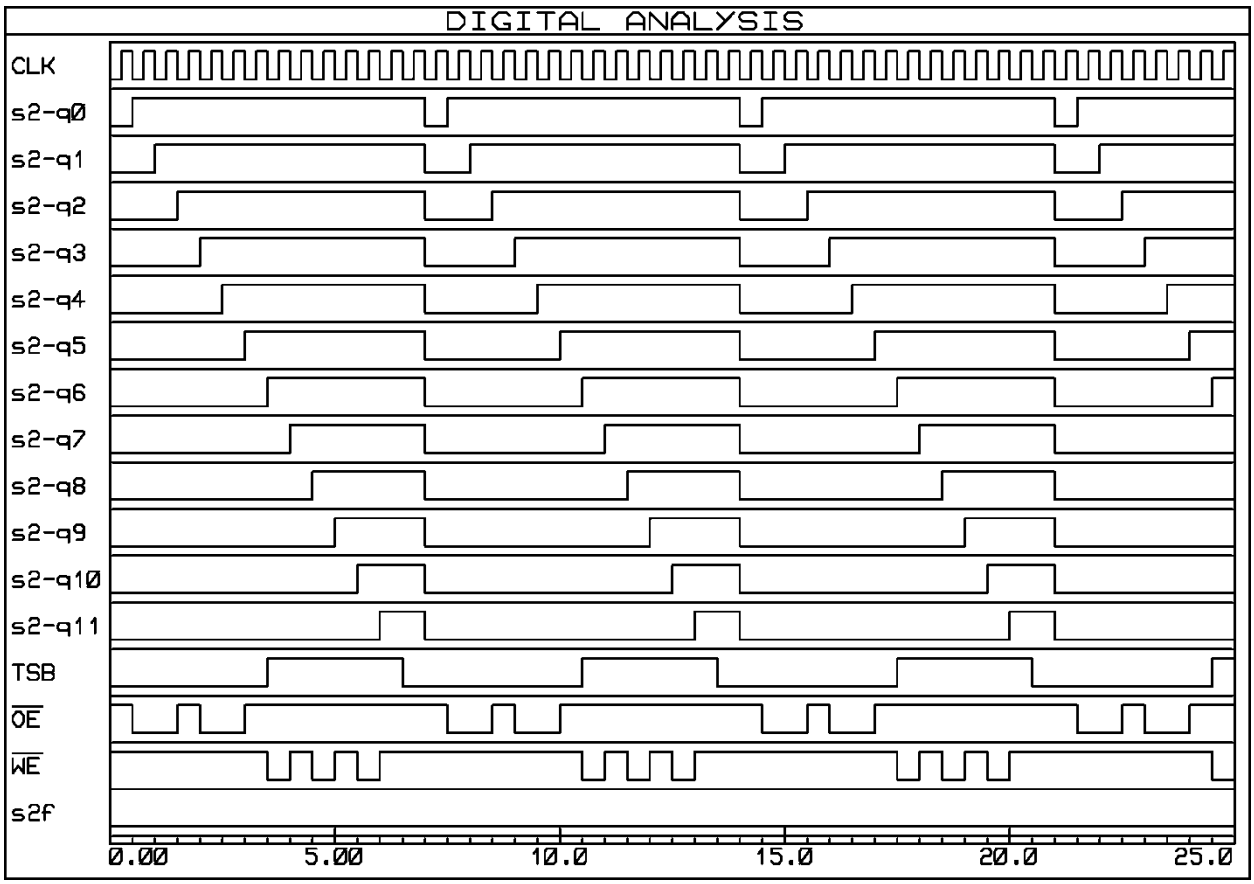


Рисунок 9 — График логических состояний и сигналов управления ОЗУ 2 этапа сортировки

## 2.4 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).

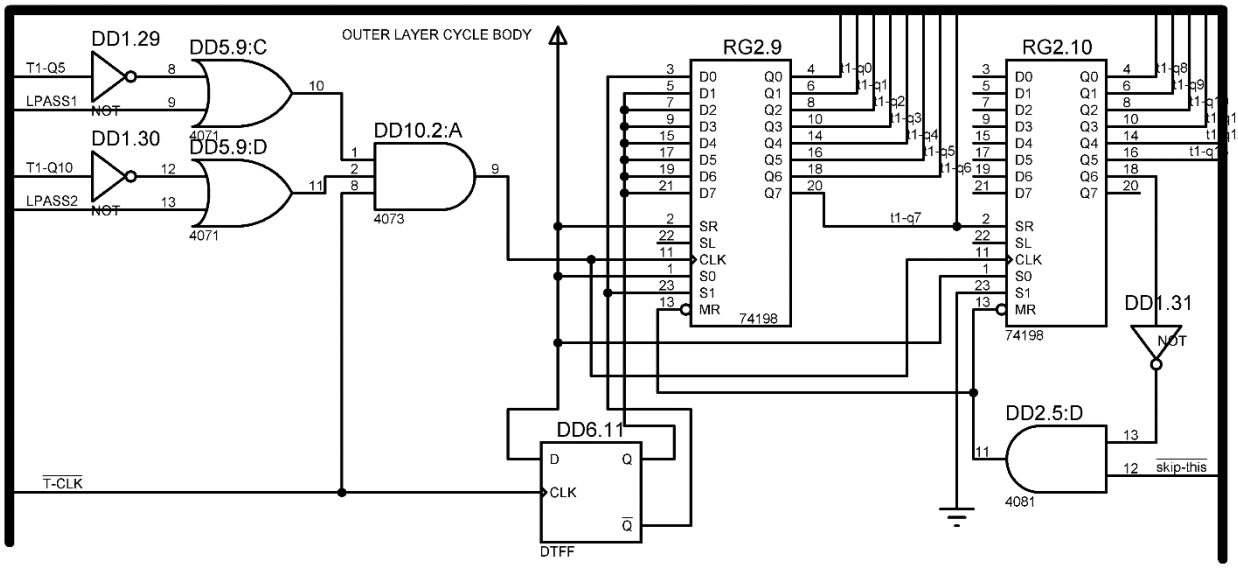


Рисунок 10 — Цикл для прохода по слою

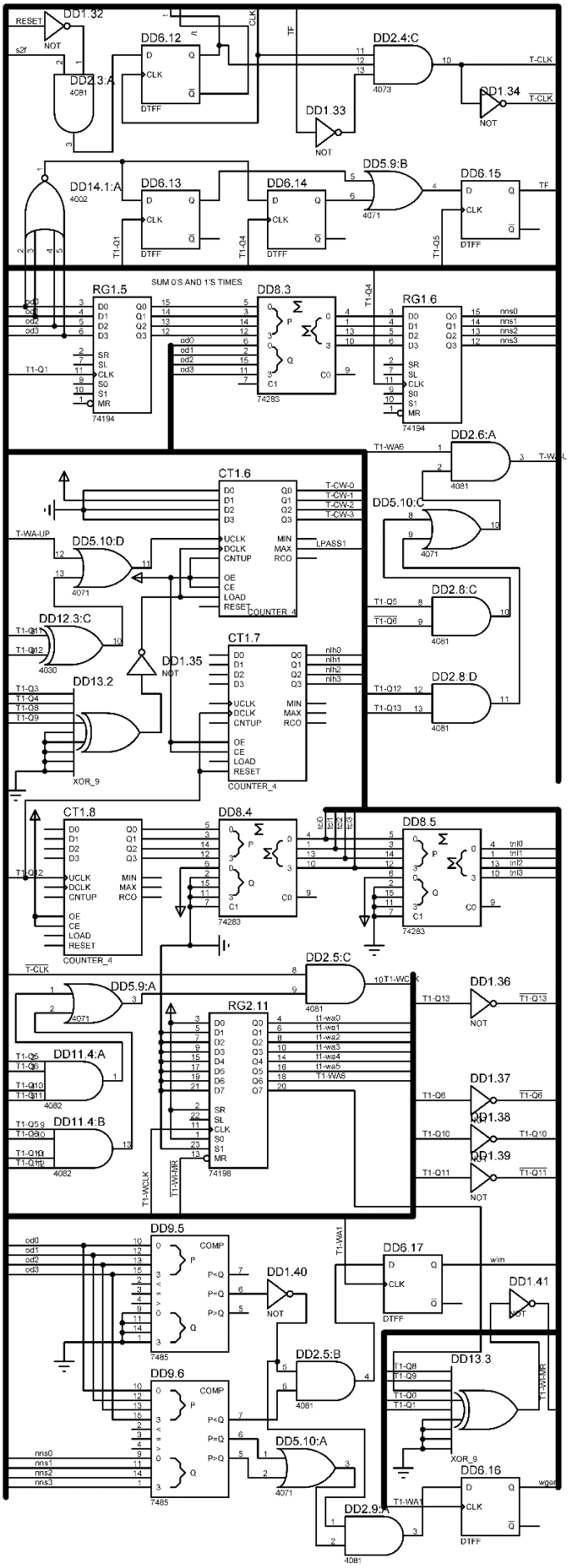


Рисунок 11 — Схема блока построения кодового дерева Хаффмана

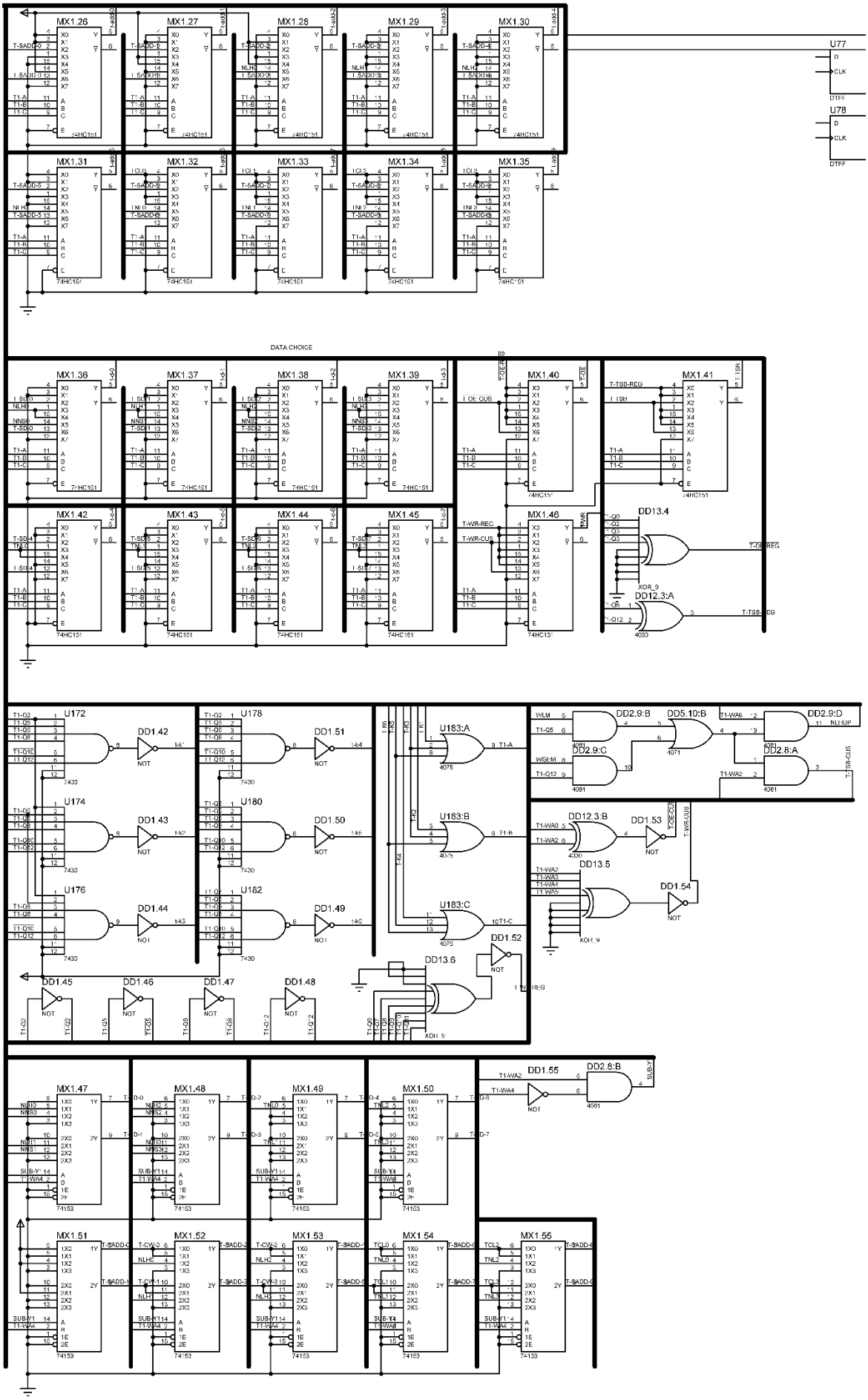


Рисунок 12 — Секция мультиплексоров блока построения и реализация функции управления

## 2.5 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.

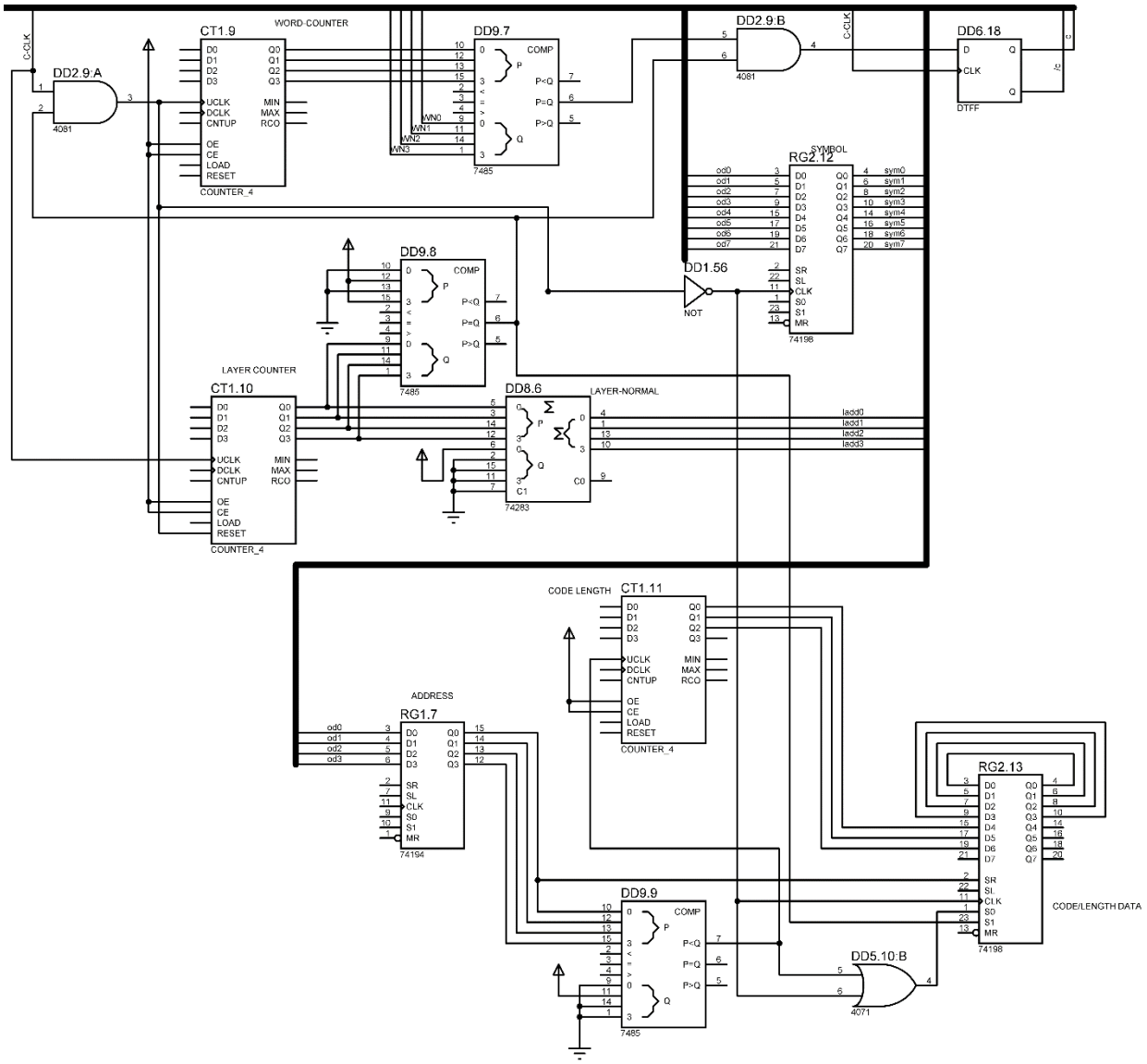


Рисунок 13 — Схема получения кода Хаффмана для каждого символа

## 2.6 Составление функциональной схемы блока вывода

Непрерывный вывод кода осложнен тем, что чтение из ОЗУ занимает 3 такта — выставление нужного адреса, понижение уровня /OE и считывание информации с информационных выводов ОЗУ. В связи с этим вывод следующего бита выходного кода трудоёмок в реализации.

Как альтернативный подход будем использовать 2 логических вывода устройства T и I. I будет поднят в уровень логической единицы на протяжении всего времени вывода кода. Сигнал T будет поднят только в те такты, когда с выводов должен быть считан следующий бит выходной последовательности.

В начале работы блока вывода I поднимается в уровень логической единицы. Устройство за 4 такта читает первый символ введённой последовательности и его код из ОЗУ. Затем сигнал T становится равным 1, одновременно с чем на выводе B появляется первый бит кода. Счётчик количества выведенных бит увеличивается на 1 и проверяется на равенство с заявленной длиной кода символа, полученной на этапе получения кода символа и лежащей рядом с самим кодом в ОЗУ. Как только код символа выведен в полном размере, T опускается в 0. Если последовательности символов заканчивается, то W переводится в 0 и блок заканчивается работу. Иначе алгоритм повторяется для следующего символа входной последовательности, пока она не закончится.

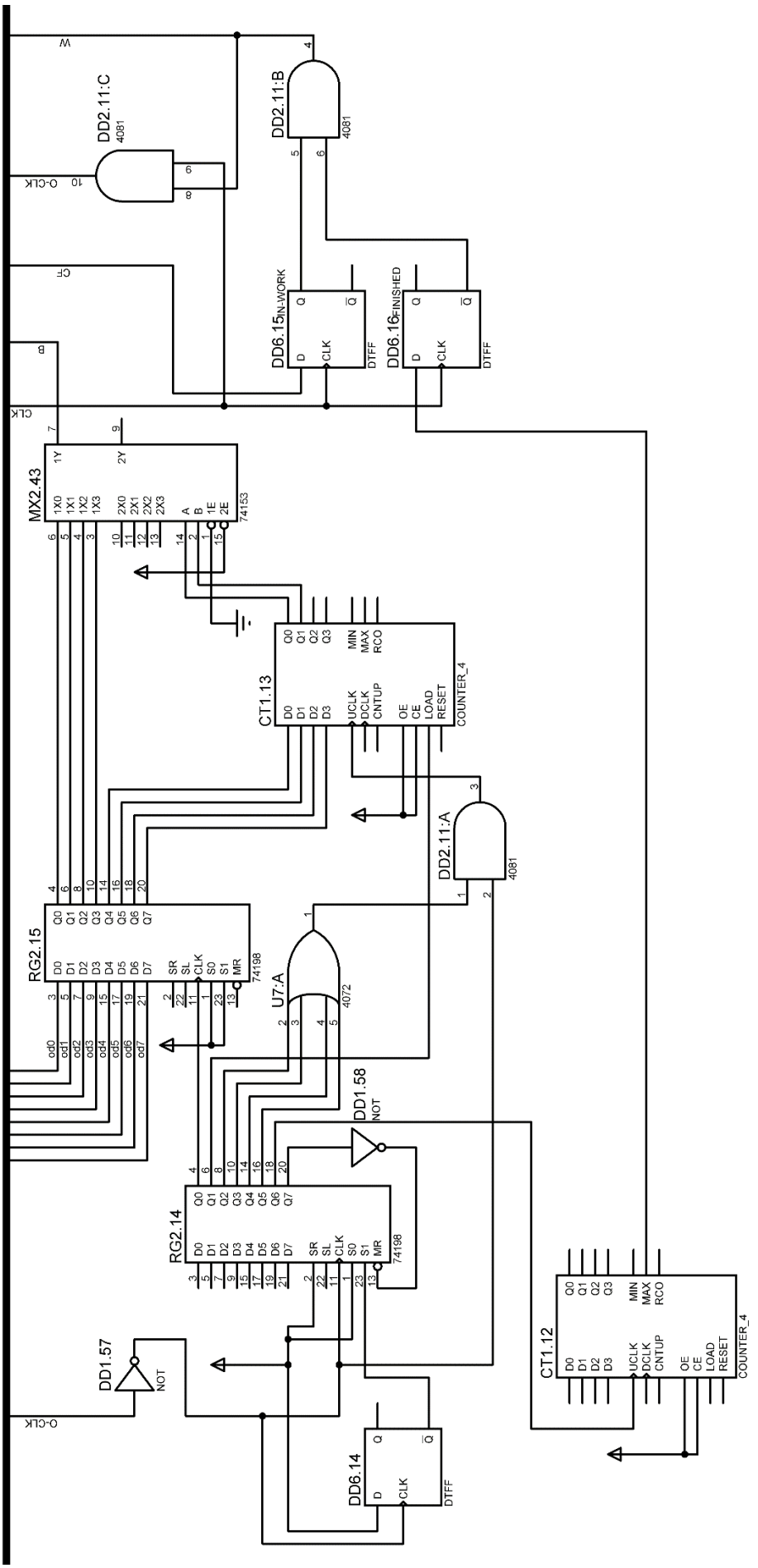


Рисунок 14 — Схема блока вывода кода

## 2.7 Составление функциональной схемы блока индикации

Индикация вывода производится на сетку двуцветных диодов (красный — ‘0’, зеленый — ‘1’, не горит — информации больше нет).

Длина сообщения не превышается 16 символов. Длина кода каждого символа оценивается сверху в 4 бита. Реализуем схему диодной сетки 8x8 с последовательным построчным её заполнением.

Для хранения информации о состоянии 1 диода необходимо 2 бита информации — горит/не горит, красный/зеленый (0/1). Для этого у каждого диода поместим блок хранения информации о нём, содержащий в себе 2 D-триггера, где триггер активности диода стробирует прямой и инверсный выводы диода цвета.

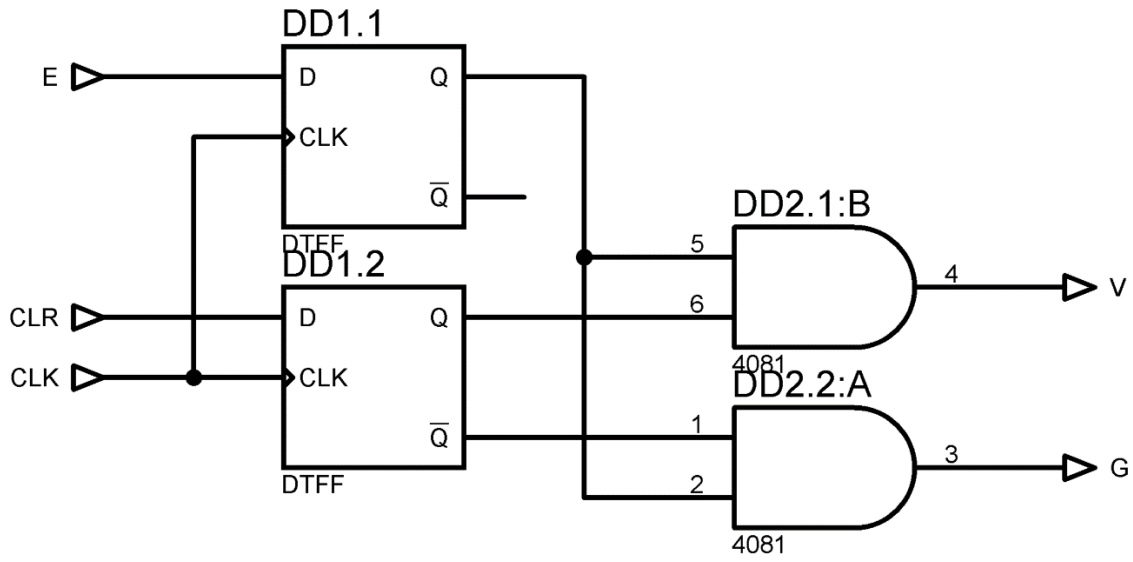


Рисунок 15 — Схема блока хранения информации о состоянии диода

В самой схеме старшие 3 бита 6-разрядного адреса диода демультиплексируют (производят дешифрование) строку, в которой необходимо зажечь/потушить диод. Младшие 3 бита адреса подаются на каждый из 8 дешифраторов, выбирающие 1 из 8 диодов в строке, к которой они относятся.

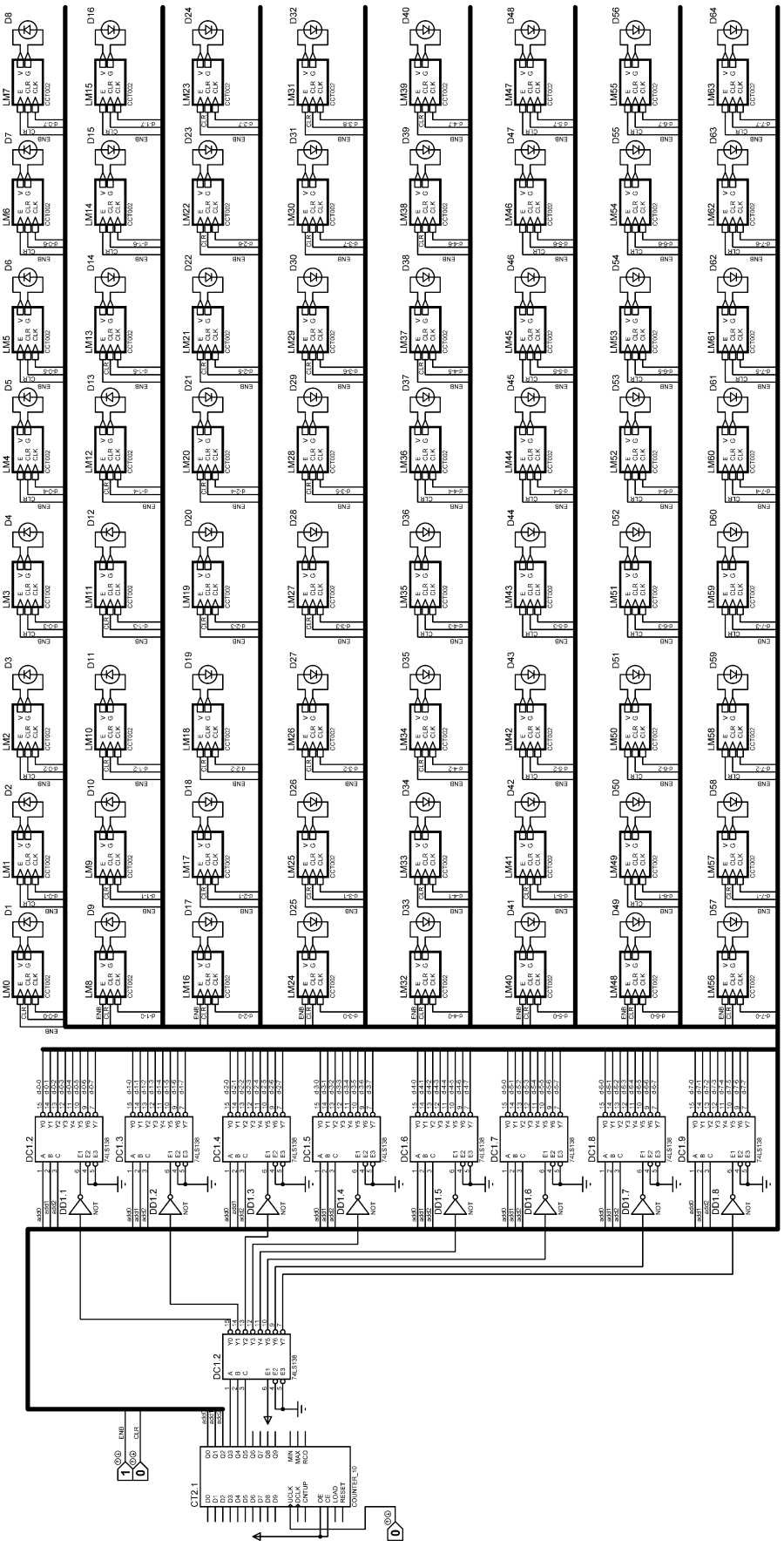

Рисунок 16 — Схема сетки диодов для индикации вывода

## 2.8 Расчет элементов микросхемы

Для построения микросхемы были использованы логические элементы, запоминающие устройства, комбинационные микросхемы — компоненты, не требующие расчета; генераторы: постоянного напряжения и прямоугольных импульсов. Первый класс выполняет роль источника сигнала логической единицы, поэтому должны выдавать напряжение 5 В. Второй класс генерирует тактовые сигналы, обеспечивающие тактирование схемы.

## 2.9 Выбор реальных элементов для схемы

Перечень использованных элементов, их реальные аналоги приведены в Приложении А. количество требуемых компонентов указано с учетом того, что некоторые логические элементы выпускаются в одном корпусе по несколько штук.

## 2.10 Расчет мощности устройства

Для определения потребляемой мощности устройства необходимо вычислить сумму потребляемых мощностей для каждого из использованных элементов – эти значения можно получить из документов соответствующих реальных компонентов. В таблице 2 указаны мощности каждого элемента.

Таблица 2 — Мощности использованных компонентов

| Компонент | Потребляемая мощность, мВт |
|---|---|
| НЕ | 54 |
| 2-И | 50 |
| 2-ИЛИ | 54 |
| 3-И | 54 |
| 4-И | 54 |
| 2-М2 | 54 |
| 4-ИЛИ-НЕ | 59 |
| 3-ИЛИ | 70 |
| 4-разрядный счетчик | 34 |
| 8-разрядный счетчик | 36 |
| Дешифратор 3-8 | 87 |
| 7-канальный мультиплексор | 150 |
| 2-канальный мультиплексор | 150 |
| Тристабильный буфер | 54 |

| | |
|---|---|
| 4-разрядный сумматор | 58 |
| 4-разрядный компаратор | 180 |
| 4-разрядный универсальный регистр сдвига | 120 |
| 8-разрядный универсальный регистр сдвига | 98 |
| Статическое ОЗУ 1Кх8 | 200 |
| Синхронный D-триггер | 60 |
| Двуцветный LED-диод | 10 |

(54·12 + 50·35 + 54·3 + 54·1 + 54·1 + 54·11 + 59·2 + 70·2 + 87·9 + 34·13 + 36·1 + 150·54 + 150·43 + 54·3 + 58·6 + 180·8 + 120·7 + 98·15 + 200·1 + 60·144 + 10·60) мВт = 33031 мВт. Следовательно, потребляемая мощность смоделированного устройства равна 33 Вт.

# ЗАКЛЮЧЕНИЕ

В рамках курсовой работы было исследовано применение кодирования Хаффмана для сжатия данных, а также было разработано устройство, которое может осуществлять кодирование данных с помощью этого метода. В ходе работы были выполнены следующие задачи:

- Изучены теоретические основы кодирования Хаффмана и принцип его работы. Получено представление о том, как метод может быть применен для сжатия данных;
- Разработаны логические блоки электронной схемы кодирования данных с использованием кода Хаффмана

В результате выполненной работы можно сделать вывод о том, что реализация устройства кодирования данных с помощью кода Хаффмана является актуальной задачей, поскольку кодирование Хаффмана является одним из наиболее эффективных методов сжатия данных;

Важным результатом работы является разработка электронной схемы, которая может осуществлять кодирование данных с помощью кода Хаффмана. Схема реализована с использованием цифровых элементов без использования микроконтроллеров и учитывает особенности работы этого конкретного метода сжатия данных;

Полученные результаты подтверждают возможность применения устройства для решения задач сжатия данных в различных областях — от технологий хранения и передачи данных до обработки медицинских изображений и видео файлов;

В процессе выполнения работы были получены ценные знания и навыки в области электроники и схемотехники.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Энциклопедия о высоких технологиях / Статическая оперативная память [Электронный ресурс]. — Режим доступа: http://allht.ru/inf/pc/mem_sram.html (Дата обращения: 04.04.2023)
2. Википедия / Код Хаффмана [Электронный ресурс]. — Режим доступа: https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%A5%D0%B0%D1%84%D1%84%D0%BC%D0%B0%D0%BD%D0%B0 (Дата обращения 10.03.2023)
3. Сайт сетевого издания Compression / Всё о коде Хаффмана [Электронный ресурс]. — Режим доступа: http://www.compression.ru/download/articles/huff/simakov_2002_huffcode.html (Дата обращения 01.04.2023)
4. Сайт сетевого издания iXBT / Современная оперативная память [Электронный ресурс]. – Режим доступа: https://www.ixbt.com/mainboard/ram-faq-2006.shtml (Дата обращения: 01.05.2023)

| Зона | Поз. обознач. | Наименование | Кол. | Примечание |
|---|---|---|---|---|
| | | Комбинационные микросхемы | | |
| | DC1 | Дешифратор 3-8 74LS138 | 9 | |
| | CT1 | Четырехразрядный бинарный счетчик 74LS93 | 13 | |
| | CT2 | Восьмиразрядный бинарный счетчик 74LS193 | 1 | |
| | MX1 | Семиканальный мультиплексор CD4051 | 54 | |
| | MX2 | Двухканальный мультиплексор CD4052 | 43 | |
| | DD3 | Тристабильный буфер 74LS125 | 3 | В корпусе 4 шт. |
| | DD8 | Четырехразрядный сумматор 74LS283 | 6 | |
| | DD9 | Четырехразрядный компаратор 74LS85 | 8 | |
| | | | | |
| | | Регистры | | |
| | RG1 | Универсальный 4-разрядный сдвиговый регистр SN74LS194 | 7 | |
| | RG2 | Универсальный 8-разрядный сдвиговый регистр SN74LS597 | 15 | |
| | | | | |
| | | Логические элементы | | |
| | DD1 | НЕ 74LS04 | 12 | В корпусе 6 шт. |
| | DD2 | 2-И 74LS00 | 35 | В корпусе 4 шт. |
| | DD5 | 2-ИЛИ 74LS32 | 3 | В корпусе 4 шт. |
| | DD10 | 3-И 74LS11 | 1 | В корпусе 3 шт. |
| | DD11 | 4-И 74LS21 | 1 | В корпусе 2 шт. |
| | DD12 | 2-М2 74LS86 | 11 | В корпусе 4 шт. |
| | DD14 | 4-ИЛИ-НЕ 74HC4002D | 2 | В корпусе 2 шт. |
| | DD15 | 3-ИЛИ 74LS4075 | 2 | В корпусе 3 шт. |
| | | | | |
| | | ОЗУ | | |
| | RAM1 | Асинхронное статичекое ОЗУ 1Кх8 7130SA | 1 | |
| | | | | |
| | | | | |

| Изм | Лист | № докум. | Подпись | Дата |
|---|---|---|---|---|
| Разраб. | | Лагов С.П. | | |
| Провер. | | Ковынёв Н.В. | | |
| Т. контр. | | | | |
| Реценз. | | | | |
| Н. контр. | | Ковынёв Н.В. | | |
| Утверд. | | | | |

МГТУ.000001.001 ПЭ

Устройство кодирования данных с помощью кода Хаффмана

| Лит | Масса | Масштаб |
|---|---|---|
| | | |

| Лист 1 | Листов |
|---|---|

Приложение А

МГТУ им. Н.Э. Баумана
Группа: ИУ8-62

| Зона | Поз. обознач. | Наименование | Кол. | Примечание |
|---|---|---|---|---|
| | | Триггеры | | |
| | DD6 | D-триггер SN74LS74 | 144 | |
| | | | | |
| | | Диоды | | |
| | D | Двуцветный LED-диод CLVBA-FKA-CA | 64 | |
| | | | | |
| | | Источники питания | | |
| | G1 | Генератор постоянного напряжения HF65W-DSM-A | 1 | |
| | G2 | Генератор прямоугольных импульсов NS047 | 1 | |

| Изм | Лист | № докум. | Подпись | Дата | МГТУ.000001.001 ПЭ | | | |
|---|---|---|---|---|---|---|---|---|
| Разраб. | | Лагов С.П. | | | Устройство кодирования данных с помощью кода Хаффмана | Лит | Масса | Масштаб |
| Провер. | | Ковынёв Н.В. | | | | | | |
| Т. контр. | | | | | | Лист 2 | Листов 2 | |
| Реценз. | | | | | Приложение А | МГТУ им. Н.Э. Баумана Группа: ИУ8-62 | | |
| Н. контр. | | Ковынёв Н.В. | | | | | | |
| Утверд. | | | | | | | | |