1 Семестр.

Лабораторная работа №1. Построение логических вентилей по КМОП логике.

Цель: построить из транзисторов по КМОП логике в программе Logisim следующие логические вентили:
И(AND), ИЛИ(OR), НЕ(NOT), И-НЕ(NAND), ИЛИ-НЕ(NOR).
Примечание: при построении элементов И-НЕ(NAND) и ИЛИ-НЕ(NOR) использовать только две пары транзисторов разных типов.

Задания:

1. **Вентиль НЕ:**

   Используйте один транзистор p-типа и один транзистор n-типа.

   p-тип подключите к источнику питания, n-тип к земле.

   Вход пускаем на затворы обоих транзисторов, выход – на соединение стоков.

2. **Вентиль И:**

   Используйте два транзистора n-типа последовательно и два транзистора p-типа параллельно.

   n-типы подключите к источнику питания, p-типы к земле.

   Каждый вход пускаем на пары затворов n-типа и p-типа, выход на соединение стоков n-типов и p-типов.

3. **Вентиль ИЛИ:**

   Используйте два транзистора n-типа параллельно и два транзистора p-типа последовательно.

   n-типы подключите к источнику питания, p-типы к земле.

   Каждый вход пускаем на пары затворов n-типа и p-типа, выход на соединение стоков n-типов и p-типов.

4. **Вентиль И-НЕ:**

   Используйте два транзистора n-типа последовательно и два транзистора p-типа параллельно.

   p-типы подключите к источнику питания, n-типы к земле.

Каждый вход пускаем на пары затворов n-типа и p-типа, выход на соединение стоков n-типов и p-типов.

5. **Вентиль ИЛИ-НЕ:**

   Используйте два транзистора n-типа параллельно и два транзистора p-типа последовательно.

   p-типы подключите к источнику питания, n-типы к земле.

   Каждый вход пускаем на пары затворов n-типа и p-типа, выход на соединение стоков n-типов и p-типов.

Лабораторная работа №2. Компьютерная арифметика.

Цель: научиться строить и анализировать сумматоры, «вычитатор» и «умножатор» в программе Logisim. Исследовать восьмибитное число, используя сдвиги.

Задания:

1. **Сумматор двух двухбитных чисел:**

   Составить таблицу истинности.

   Минимизировать с помощью карты Карно.

   Построить в Logisim.

2. **Построение сумматоров:**

   Полусумматор: использовать элементы И и Исключающее ИЛИ, обозначить входы (A, B) и выходы (Sum, CarryOut)

   Полный сумматор: использовать два полусумматора и элемент ИЛИ, указать CarryInput, Sum, CarryOut.

   8-битный сумматор: соединить полные сумматоры, указать CarryInput, CarryOut.

3. **Инвертор:**
   Использовать Исключающие ИЛИ для подачи на них входного значения и значения инвертирования.

4. **«Вычитатор»:**

   Подать входные значения A и B.

   Использовать инвертор, 8-битный сумматор, контакт для вычитания и Исключающее ИЛИ

5. **«Умножатор» двух четырехбитных чисел:**

   Использовать частичные произведения и 4-битные сумматоры.

   Реализовать схему умножения.

6. **Исследование числа посредством сдвигов:**

   Изучить логический, арифметический и циклический сдвиги.

   Исследовать число 10000001.

   Представить результаты в беззнаковом и знаковом десятичном виде.

Лабораторная работа №3. Шифратор, Дешифратор, Мультиплексор, Демультиплексор.

Цель: научиться строить и анализировать основные комбинационные схемы (шифратор, дешифратор, мультиплексор, демультиплексор) в программе Logisim.

Задания:

1. **Построить Шифратор:**

   Преобразует входные сигналы в код.

   Использовать логические элементы ИЛИ.

   Реализовать шифровку чисел от 0-15 в двоичный код.

2. **Построить Дешифратор:**

   Преобразует код в уникальный выход.

   Использовать логические элементы И и ИЛИ.

   Реализовать дешифровку двоичного кода в числа от 0-F.

   Осуществить вывод данных на семисегментный индикатор.

3. **Построить Мультиплексор:**

   Выбирает один из нескольких входов и передает его на выход.

   Использовать логические элементы И, ИЛИ и НЕ.

   Конечное значение должно выходить через элемент ИЛИ.

4. **Построить Демультиплексор:**

   Передает входной сигнал на один из нескольких выходов.

   Использовать логические элементы И, НЕ.

   Конечные значения должны выходить из каждого элемента И отдельно.

Лабораторная работа №4. Построение триггеров и последовательных схем.

Цель: научиться строить и анализировать различные типы триггеров, счетчики, регистры и память в программе Logisim.

Задания:

1. **Построить синхронный RS-триггер:**

   Синхронный и асинхронный.

   Использовать логические элементы И-НЕ.

   Входные значения Set, Reset, Clock.

2. **Построить асинхронный RS-триггер:**

   Синхронный и асинхронный.

   Использовать логические элементы И-НЕ.

   Входные значения Set, Reset.

3. **Построить JK-триггер из RS-триггеров:**

   Использовать RS-триггеры и логические элементы И, НЕ.

   Входные значения Jump, Kill и Clock.

4. **Построить D-триггер из RS-триггера:**

   Использовать логические элемент НЕ.

   Входные значения Data и Clock.

5. **Построить T-триггер из RS-триггеров:**

   Использовать логический элемент НЕ и RS-триггеры.

   Добавить вход Toggles.

6. **Построить счетчик:**

   Использовать JK-триггеры.

   Реализовать счет по модулю 2^n.

7. **Построить 8-битный регистр:**

   Использовать D-триггеры.

   Входные значения числа, Clock, Reset.

   Реализовать параллельный ввод и вывод данных.

8. **Построить память:**

   Использовать восьмибитные регистры по переднему фронту.

   Реализовать матрицу памяти при помощи мультиплексора и демультиплексора.

   Добавить кнопку Reset.

9. **Модифицировать сумматор:**

   Добавить память с использованием регистров.

   Добавить кнопки Clock, Reset.

   Реализовать хранение результата.

2 Семестр.

Лабораторная работа №1. Моделирование перекрёстка со светофорами и реализация собственного конечного автомата.

Цель: научиться строить блок-схемы, таблицы истинности, графы и таблицы переходов для перекрёстка со светофорами, а также реализовывать собственные конечные автоматы.

Задание 1. Перекрёсток со светофорами и пешеходными переходами.

1. **Создание Счетчика:**

   Построить счетчик на основе JK-триггеров для управления последовательностью цветов светофора.

2. **Полноценный Светофор:**

   Использовать счетчик для переключения между состояниями светофора (красный, желтый, зеленый).

   Добавить логику для пешеходных переходов, активируемых по кнопке.

3. **Перекрёсток:**

   Создать схему перекрестка со светофорами и пешеходными переходами.

   Синхронизировать светофоры для безопасного движения.

4. **Граф Переходов:**

   Определить состояния светофоров и переходов.

   Нарисовать граф с переходами между состояниями.

5. **Блок-Схема:**

   Создать блок-схему алгоритма работы перекрестка.

6. **Таблица Переходов:**

   Составить таблицу с текущим состоянием и следующим состоянием.

   Указать условия переходов.

7. **Таблица Истинности:**

   Составить таблицу истинности для логических условий переходов.

Задание 2. Свой конечный автомат. *Пример: торговый автомат (Vending machine).*

1. **Граф Переходов:**

   Определить состояния (ожидание ввода, обработка монет, выбор товара, выдача товара).

   Нарисовать граф с переходами между состояниями.

2. **Блок-Схема:**

   Создать блок-схему алгоритма работы автомата.

3. **Таблица Переходов:**

   Составить таблицу с текущим состоянием и следующим состоянием.

   Указать условия переходов (ввод монет, выбор товара, выдача товара).

4. **Реализация в Logisim:**

   Добавьте триггеры (D-триггеры) для хранения текущего состояния.

   Используйте логические элементы (И, ИЛИ, НЕ) для обработки условий переходов.

   Создайте входы для монет, выбора товара и кнопки выдачи.

   Создайте выходы для индикации состояний и выдачи товара.

   Свяжите входы и выходы с соответствующими логическими элементами и триггерами.

Лабораторная работа №2. Построение АЛУ с аккумулятором.

Цель: научиться проектировать и реализовывать действия над арифметико-логическим устройством (АЛУ) с аккумулятором в программе Logisim.

Задание:

1. **Построить АЛУ:**
    - Реализовать основные арифметические и логические операции (сложение, вычитание, И, ИЛИ, НЕ).
    - Использовать восьмибитные входы и выходы.
2. **Реализовать Аккумулятор:**
    - Создать регистр для хранения результата операций АЛУ.
    - Реализовать возможность загрузки данных в аккумулятор и выгрузки данных из него.
3. **Интегрировать АЛУ и Аккумулятор:**
    - Связать выход АЛУ с входом аккумулятора.
    - Реализовать управление загрузкой и сохранением данных в аккумуляторе.

Лабораторные работы №3-4. Построение процессоров Гарвардской архитектуры и архитектуры Фон Неймана.

Цель: научиться проектировать и совершать действия над процессорами по архитектуре Фон Неймана или Гарвардской архитектуре в программе Logisim.

Задание:

1. **Выбор Архитектуры:**
    - Выберите архитектуру (Фон Неймана или Гарвардская).
2. **Построение Процессора:**
    - Реализовать основные компоненты процессора: АЛУ, регистры, память, шины данных и адреса.
    - Определить набор команд и их выполнение.
3. **Интеграция Компонентов:**
    - Связать компоненты в соответствии с выбранной архитектурой.
    - Реализовать управление потоком данных и команд.

Методика:

1. **Выбор Архитектуры:**
    - Определитесь с архитектурой (Фон Неймана или Гарвардская).
2. **Построение Процессора:**
    - **АЛУ:** Добавьте 8-битное АЛУ с основными операциями (сложение, вычитание, И, ИЛИ, НЕ).
    - **Регистры:** Создайте регистры общего назначения (РОН) и специальные регистры (например, PC, IR).
    - **Память:** Реализуйте ОЗУ и ПЗУ (для Гарвардской архитектуры — отдельно для данных и команд).
    - **Шины:** Создайте шины данных и адреса.
3. **Интеграция Компонентов:**
    - **Фон Неймана:** Свяжите АЛУ, регистры и память через единую шину данных и адреса.
    - **Гарвардская:** Реализуйте отдельные шины для данных и команд, свяжите их с соответствующими компонентами.
4. **Управление Потоком Данных и Команд:**
    - Реализуйте микрокод или логику управления для выполнения команд.
    - Определите последовательность выполнения команд (выборка, декодирование, исполнение).

Лабораторная работа №5. Начало работы с ассемблером (практикум LamPanel).

1. Введите программу
   DATA 01D0
   DATA 3536
   DATA 0101
   DATA FFFF
   Используя дисассемблер, запишите эту программу на языке ассемблера. Запишите содержимое
   памяти, в которой располагается эта программа, в
   виде последовательности символов.
2. Как вы думаете, какой код будет иметь команда MOV R3,R2? Проверьте свой ответ с помощью программы.
3. Блок данных программы выглядит так.
   A:
   DATA 1234
   B:
   DATA 4321
   SUM:
   DATA 0
   Напишите программу, которая складывает
   переменные A и B и записывает результат в переменную SUM.
   4. Напишите программу, которая преобразует
   строчные буквы в заглавные, используя байтовые
   операции. Что произойдет, если среди исходных
   данных уже есть заглавные буквы?
   5. Усовершенствуйте программу так, чтобы цикл
   останавливался не после заданного количества букв, а тогда,
   когда очередной прочитанный байт равен 0.
   6. Поскольку в компьютере с архитектурой фон
   Неймана программа и данные расположены в одной области
   памяти, программа может менять
   свой собственный код. Напишите какую-нибудь
   программу, которая изменяет сама себя во время
   работы.

Лабораторная работа №6. Проектирование клавиатуры на языке ASCII.

Цель: научиться строить и пользоваться клавиатурой на языке ASCII в программе Logisim.

Задание:

1. Определение структуры клавиатуры

Клавиатура на языке ASCII представляет собой набор кнопок, каждая из которых соответствует определенному символу. Нажатие кнопки генерирует соответствующий ASCII-код, который передается в схему.

2. Создание схемы для клавиш

Каждая клавиша на клавиатуре должна быть представлена в Logisim как отдельный компонент, например, в виде кнопок (Button). Эти кнопки будут активировать соответствующие сигналы при нажатии.

3. Кодирование ASCII-символов

Для преобразования нажатий клавиш в ASCII-коды вам потребуется использовать компонент ROM или таблицу истинности.

Использование ROM:

- В ROM можно записать соответствие между нажатием клавиши (адрес в памяти) и её ASCII-кодом (значение по этому адресу).
- Загрузите ROM с необходимыми данными, где адреса представляют клавиши, а данные — их ASCII-коды.

- Таблица истинности:

- Альтернативно, вы можете использовать логические схемы и мультиплексоры для реализации таблицы истинности, которая будет преобразовывать входной сигнал (например, код клавиши) в соответствующий ASCII-код.

4. Вывод ASCII-кода

Используйте дисплей (Hex Display или LED Display) для отображения сгенерированного ASCII-кода.

5. Добавление функциональных клавиш

Для создания полноценной клавиатуры вы можете добавить специальные клавиши, такие как Backspace, Enter, Shift и Caps Lock.

6. Тестирование схемы

После завершения проектирования схемы протестируйте её работу: Проверьте, что нажатие каждой клавиши генерирует правильный ASCII-код. Убедитесь, что все специальные клавиши работают корректно(например, Enter завершает строку, Backspace удаляет символ).