МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА ИНФОРМАЦИОННО-СЕТЕВЫХ ТЕХНОЛОГИЙ

КУРСОВАЯ РАБОТА (ПРОЕКТ)
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

доц., канд. техн. наук		Т.Н. Соловьева
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К КУРСОВОЙ РАБОТЕ

ГЕНЕРАТОР ПИЛООБРАЗНЫХ СИГНАЛОВ

по дисциплине: МИКРОКОНТРОЛЛЕРНЫЕ СИСТЕМЫ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4143		Д.В. Пономарев
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ–ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ЗАДАНИЕ НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА

студенту группы 4143 (номер) Пономареву Дмитрию Валерьевичу (фамилия, имя, отчество)

на тему Генератор пилообразных сигналов

Цель проекта: разработка аппаратно-программного комплекса генератора пилообразных сигналов на базе микроконтроллера серии 8051 в системе автоматизированного проектирования Proteus.

Задачи, подлежащие решению: реализация возможности задания частоты и амплитуды сигнала,
генерация пилообразных импульсов напряжения заданной амплитуды и частоты.

Содержание пояснительной записки (основные разделы):
проектирование аппаратного обеспечения,
проектирование программного обеспечения,
тестирование и отладка аппаратно-программного комплекса.

Срок сдачи работы « 30 » ноября 2024

Руководитель

доц., канд. техн. наук		Т.Н. Соловьева
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

Задание принял к исполнению

студент группы № 4143		Д.В. Пономарев
	подпись, дата	инициалы, фамилия

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире цифровая электроника находит применение в самых различных областях, от научных исследований до бытовой техники. Одной из ключевых задач в электронике является генерация сигналов различных форм для тестирования и настройки устройств, а также для применения в составе сложных систем. Одной из наиболее востребованных форм сигналов является пилообразный сигнал, который используется, например, в осциллографах, анализаторах спектра и системах управления.

Актуальность темы разработки генератора пилообразного сигнала обусловлена его широкой применимостью в схемотехнике и необходимости компактных, программируемых решений для реализации такой функции. Использование микроконтроллеров, таких как AT89C51, позволяет создавать универсальные и гибкие генераторы сигналов, которые могут быть адаптированы под различные задачи.

Цель работы: разработать программируемый генератор пилообразного сигнала на базе микроконтроллера AT89C51 с возможностью изменения параметров сигнала (амплитуды и частоты) через матричную клавиатуру.

Задачи работы:

- Разработать алгоритм формирования пилообразного сигнала с фиксированным числом шагов.
- Реализовать функцию управления параметрами сигнала (амплитудой и частотой) через интерфейс матричной клавиатуры.
- Протестировать работу генератора на микроконтроллере AT89C51 и проверить корректность изменения параметров сигнала.
- Обеспечить возможность использования разработанного устройства для тестирования и настройки электронной аппаратуры.

Данный проект демонстрирует возможности интеграции программного

и аппаратного обеспечения для создания функционального устройства, что делает его полезным как для образовательных целей, так и для практического применения.

1. Проектирование аппаратного обеспечения

Обзор существующих решений

Для генерации аналоговых сигналов существует несколько подходов:

1. Использование специализированных микросхем — позволяет генерировать сигналы высокой точности и стабильности, но такие решения часто ограничены функционально или дорогостоящи.
2. Программная генерация с использованием микроконтроллеров — дает большую гибкость и возможность программной настройки параметров сигнала.
3. Применение FPGA — обеспечивает высокую производительность и универсальность, но требует глубоких знаний цифровой схемотехники.

Наиболее оптимальным для данной задачи является использование микроконтроллера (например, AT89C51) в комбинации с цифровым-аналого-преобразователем (DAC), так как это позволяет программно изменять параметры сигнала (амплитуда и частота), сохраняя при этом простоту и доступность реализации.

Выбор аппаратной реализации

Для реализации генератора пилообразного сигнала выбран микроконтроллер AT89C51 и цифровой-аналого-преобразователь LTC1450. Это решение обусловлено следующими факторами:

- AT89C51 предоставляет достаточный объем портов ввода-вывода для подключения клавиатуры и DAC.
- LTC1450 имеет простое управление через цифровые входы и позволяет генерировать стабильные аналоговые сигналы.

Дополнительно используется матричная клавиатура для пользовательского ввода параметров.

Разработка схемы проекта

Схема включает:

1. Микроконтроллер AT89C51 (U1) (рис.1):
   - Управляет работой системы и реализует генерацию сигнала.
   - Порт P1 используется для передачи данных на DAC.
   - Порт P2 используется для подключения матричной клавиатуры.

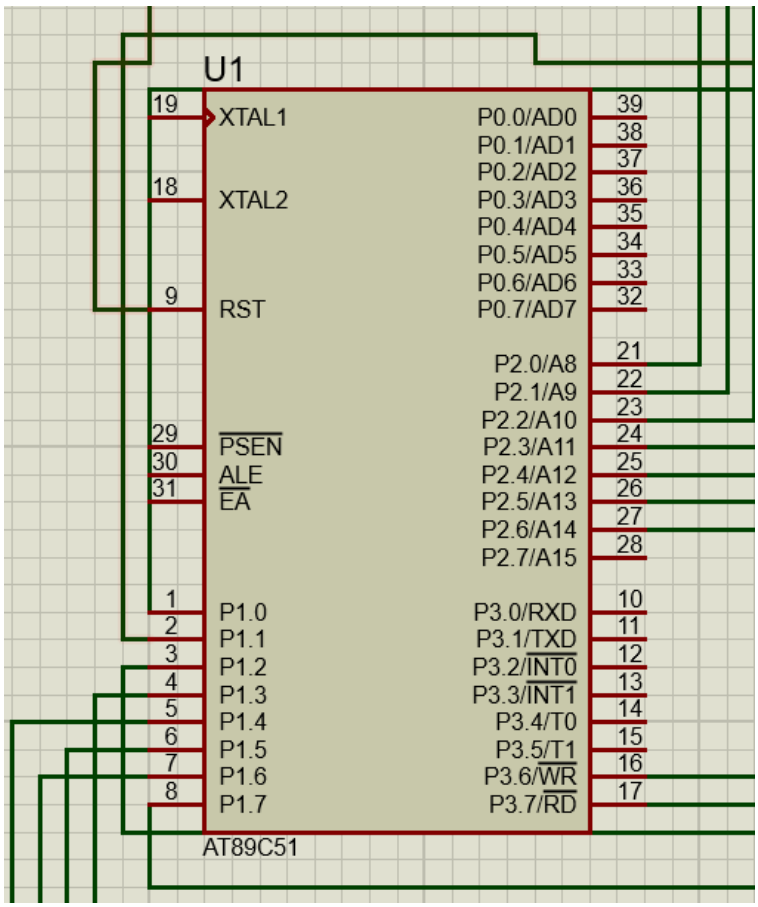

Рисунок 1- Микроконтроллер AT89C51

2. Матричная клавиатура (рис.2):

- Служит для ввода параметров сигнала (амплитуды и частоты).
- Подключена к порту P2 микроконтроллера.

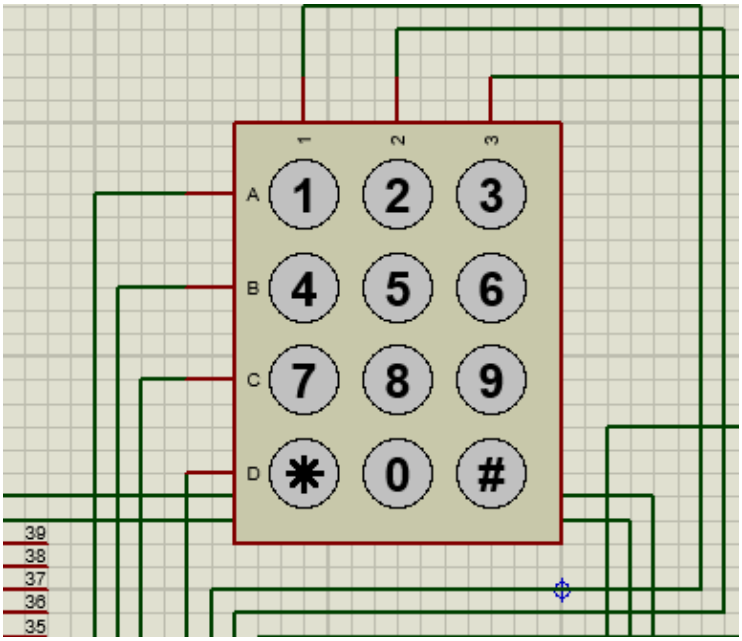

Рисунок 2- Матричная клавиатура

3. DAC LTC1450 (U2) (рис.3):

- Преобразует цифровое значение сигнала в аналоговый сигнал.
- Подключен к выходу порта P1 микроконтроллера.

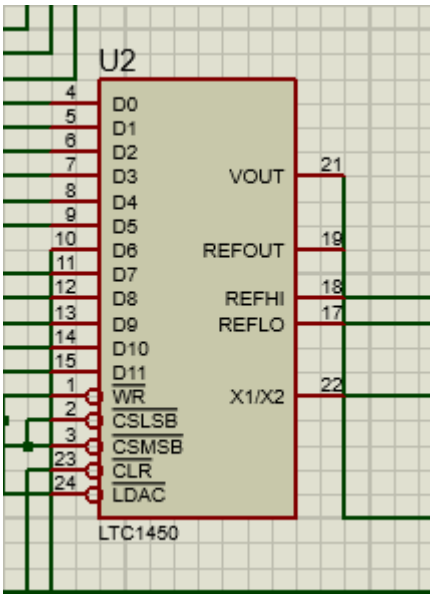

Рисунок 3- DAC LTC1450 (U2)

4. Осциллограф (рис.4):
   1. Обеспечивает вывод на экран.

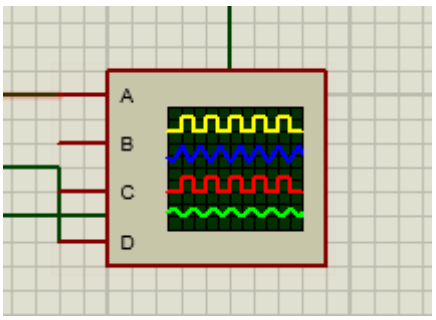

Рисунок 4- Осциллограф

Характеристики элементов и их назначение

1. **AT89C51:**
   1. Характеристики:
      - 8-разрядный микроконтроллер.
      - 4 КБ встроенной памяти для программ.
      - Поддержка портов общего назначения (GPIO).
   2. Назначение:
      - Управление процессом генерации сигнала.
      - Опрос клавиатуры и обработка пользовательского ввода.

3. **LTC1450:**

   1. Характеристики:
      - 12-разрядный ЦАП.
      - Выходное напряжение 0–5 В.

- Простое управление через цифровой интерфейс.

2. Назначение:
   - Преобразование цифрового значения пилообразного сигнала в аналоговый.

3. Матричная клавиатура:

1. Характеристики:
   - 4 строки × 3 столбца.
   - Поддерживает 12 клавиш для ввода чисел и команд.
2. Назначение:
   - Ввод значений амплитуды и частоты пользователем.

3. Осциллограф:

1. Назначение:
   - Используется для визуализации выходного сигнала DAC.

Принцип работы схемы

1. Микроконтроллер генерирует цифровые значения пилообразного сигнала.
2. Параметры сигнала (амплитуда и частота) изменяются пользователем через клавиатуру.
3. Цифровые значения передаются в LTC1450 через порт P1.
4. LTC1450 преобразует цифровое значение в аналоговый сигнал, который подается на выход.

Схема обеспечивает гибкость настройки параметров сигнала и наглядное отображение результата на осциллографе.

2. Проектирование программного обеспечения

Назначение программы

Разрабатываемое программное обеспечение предназначено для управления генератором пилообразного сигнала на базе микроконтроллера AT89C51. Программа обеспечивает генерацию пилообразного сигнала с заданными параметрами (амплитудой и частотой), которые могут быть изменены пользователем через матричную клавиатуру. Основные функции ПО включают:

1. Формирование пилообразного сигнала.
2. Обеспечение взаимодействия с пользователем через интерфейс матричной клавиатуры.
3. Изменение параметров сигнала (амплитуды и частоты) в реальном времени.

Процесс разработки программы

Разработка программы выполнялась поэтапно, начиная с анализа требований и проектирования структуры, заканчивая написанием кода и тестированием. Основные этапы:

1. Определение структуры программы и выделение ключевых функций.
2. Написание модулей для генерации сигнала и обработки ввода с клавиатуры.
3. Интеграция функций в единый алгоритм работы.
4. Тестирование программы на симуляторе и в реальных условиях.

Обобщенная структура программы

Программа состоит из следующих основных модулей:

1. Модуль управления клавиатурой:
   - Реализует опрос матричной клавиатуры для определения нажатых клавиш.
   - Позволяет вводить значения амплитуды и частоты сигнала.

2. Модуль генерации сигнала:
   - Формирует пилообразный сигнал с заданным количеством шагов.
   - Выводит сигнал на порт микроконтроллера.

3. Модуль управления параметрами сигнала:
   - Обрабатывает ввод пользователя и изменяет параметры (амплитуду и частоту) в реальном времени.

4. Главный цикл программы:
   - Координирует работу всех модулей.

Описание отдельных алгоритмов

1. Алгоритм опроса клавиатуры

Осуществляется последовательное сканирование строк и столбцов матричной клавиатуры для определения нажатой клавиши. При нахождении нажатой клавиши выполняется соответствующее действие (например, ввод числа или завершение ввода параметров).

2. Алгоритм ввода параметров

Пользователь вводит числовые значения параметров (амплитуда или частота) с клавиатуры. Ввод завершается нажатием * для амплитуды или # для частоты.

3. Алгоритм формирования пилообразного сигнала

Сигнал формируется за FIXED_STEPS шагов. На каждом шаге значение сигнала вычисляется по формуле:

$$\text{Output} = \frac{\text{count} \times \text{amplitude}}{\text{FIXED\_STEPS}}$$

После каждого шага происходит задержка, определяемая параметром delayTime.

4. Алгоритм главного цикла программы

Главный цикл программы выполняет непрерывный опрос клавиатуры, генерацию сигнала и обработку ввода параметров.

Пример взаимодействия модулей

1. Пользователь нажимает клавишу *, программа переходит в режим изменения амплитуды.
2. Пользователь вводит числовое значение амплитуды, которое сохраняется после повторного нажатия *.
3. Главный цикл программы использует новое значение амплитуды для генерации сигнала.

Программа построена таким образом, чтобы обеспечить гибкость в настройке параметров сигнала, простоту использования и возможность модификации для добавления новых функций.

3. Тестирование и отладка аппаратно-программного комплекса

Тестирование схемы.

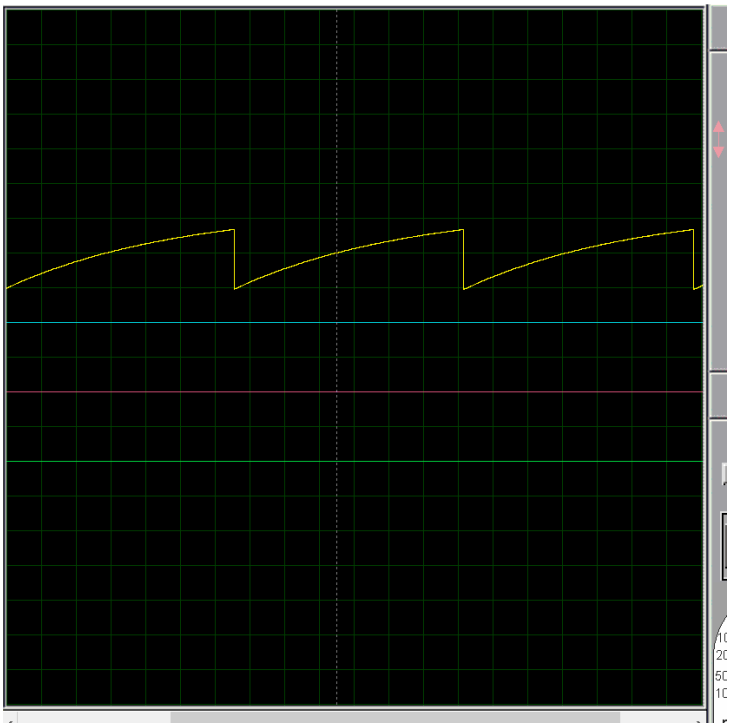

Рисунок 5 - Начальный экран после запуска

Для начала изменим значение амплитуды. Для этого нажимаем * и вводим значение например 100. Результат изменений показан на рисунке 6.

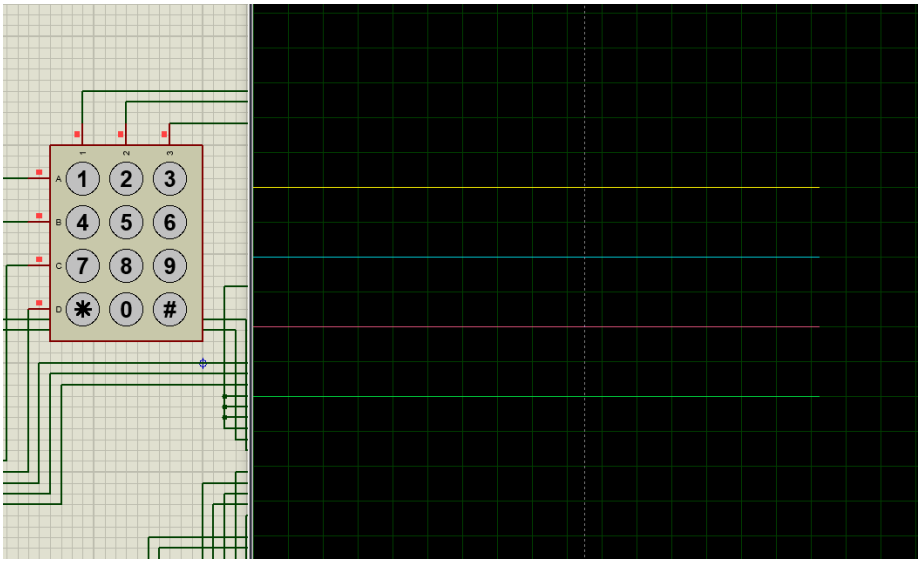

Рисунок 5 – Ожидание ввода занчения

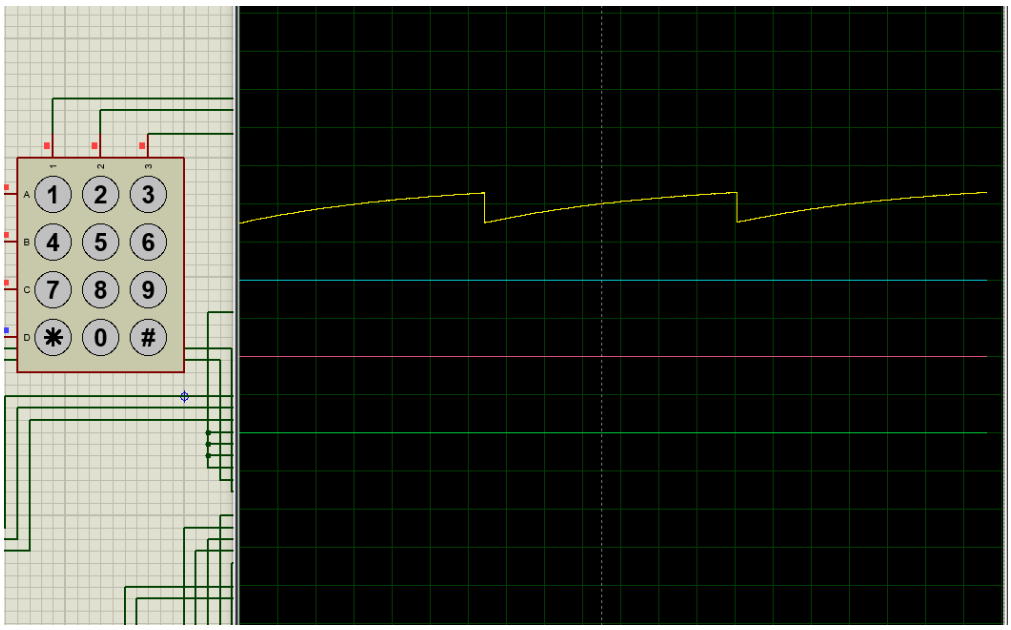

Рисунок 6 – Результат изменения амплитуды

Далее изменим значение частоты. Для этого нажимаем # и вводим значение например 1. Результат изменений показан на рисунке 7.

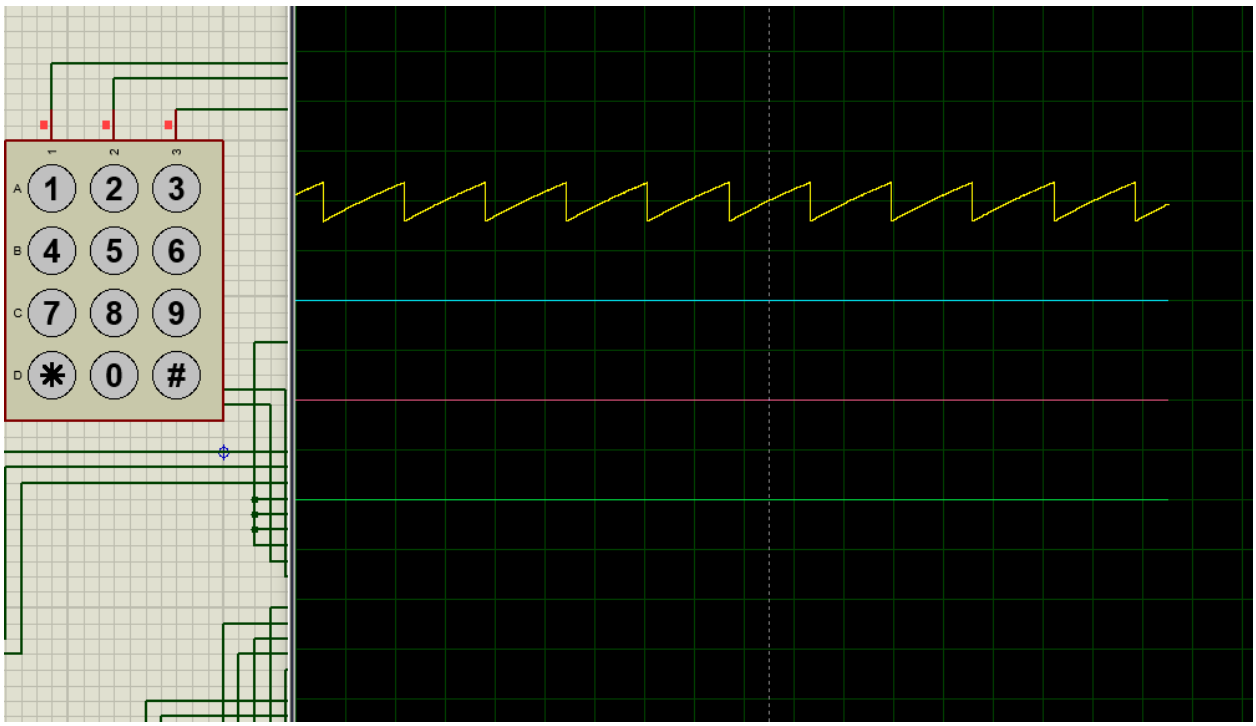

Рисунок 7 – Результат изменения частоты

ЗАКЛЮЧЕНИЕ

Разработанное устройство представляет собой генератор пилообразных сигналов, выполненный на базе микроконтроллера AT89C51 и цифрового-аналого-преобразователя LTC1450. Устройство позволяет генерировать стабильные пилообразные сигналы с настраиваемыми параметрами амплитуды и частоты, что делает его универсальным инструментом для применения в различных областях электроники.

Назначение и функции устройства

- Назначение: генерация пилообразного сигнала с возможностью гибкой настройки параметров.
- Функции:
  1. Программная установка амплитуды сигнала через матричную клавиатуру.
  2. Программная настройка частоты сигнала через матричную клавиатуру.
  3. Генерация аналогового пилообразного сигнала с использованием цифрового значения, преобразованного через ЦАП.
  4. Визуализация выходного сигнала на осциллографе.

Основные технические характеристики

- Тип сигнала: пилообразный.
- Амплитуда сигнала: от 0 до 255 шагов, программно настраиваемая пользователем.
- Частота сигнала: регулируется пользователем, минимальная задержка между отсчетами — 5 мс.
- Элементы управления: матричная клавиатура (12 клавиш: ввод числовых значений и управление режимами).
- Выходное напряжение: 0–5 В (максимальное значение ограничено характеристиками LTC1450).
- Питание устройства: 6 В (блок батарей или сетевой адаптер).

Область применения

Разработанный генератор пилообразных сигналов может применяться в следующих областях:

1. Обучение и исследовательская деятельность:

- Использование в лабораторных работах по электронике и электротехнике для изучения работы генераторов сигналов.
- Исследование параметров пилообразного сигнала в учебных процессах.

2. Разработка и тестирование устройств:
   - Применение в качестве источника тестовых сигналов при разработке аналоговых и цифровых устройств.
   - Проверка работы усилителей, фильтров и других компонентов.

3. Индустриальная автоматика:
   - Использование в системах управления, где требуется генерация пилообразного сигнала для тестирования или управления.

Устройство отличается простотой схемотехнической реализации, доступностью компонентов и гибкостью настроек, что делает его полезным как для образовательных целей, так и для инженерных разработок.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация для AT89C51: https://www.romstore.ru/system/storage/download/AT89C51.pdf
2. Документация для LTC1450: https://www.farnell.com/datasheets/1575719.pdf
3. Учебно-методические материалы к выполнению лабораторной работы №8 по дисциплине «Схемотехника» (2-й семестр изучения дисциплины)// Жаринов. О.О. URL.: https://pro.guap.ru/inside/student/tasks/ad082e0862c9584d5144a1bd553cf8b0/download

ПРИЛОЖЕНИЕ А

Схема

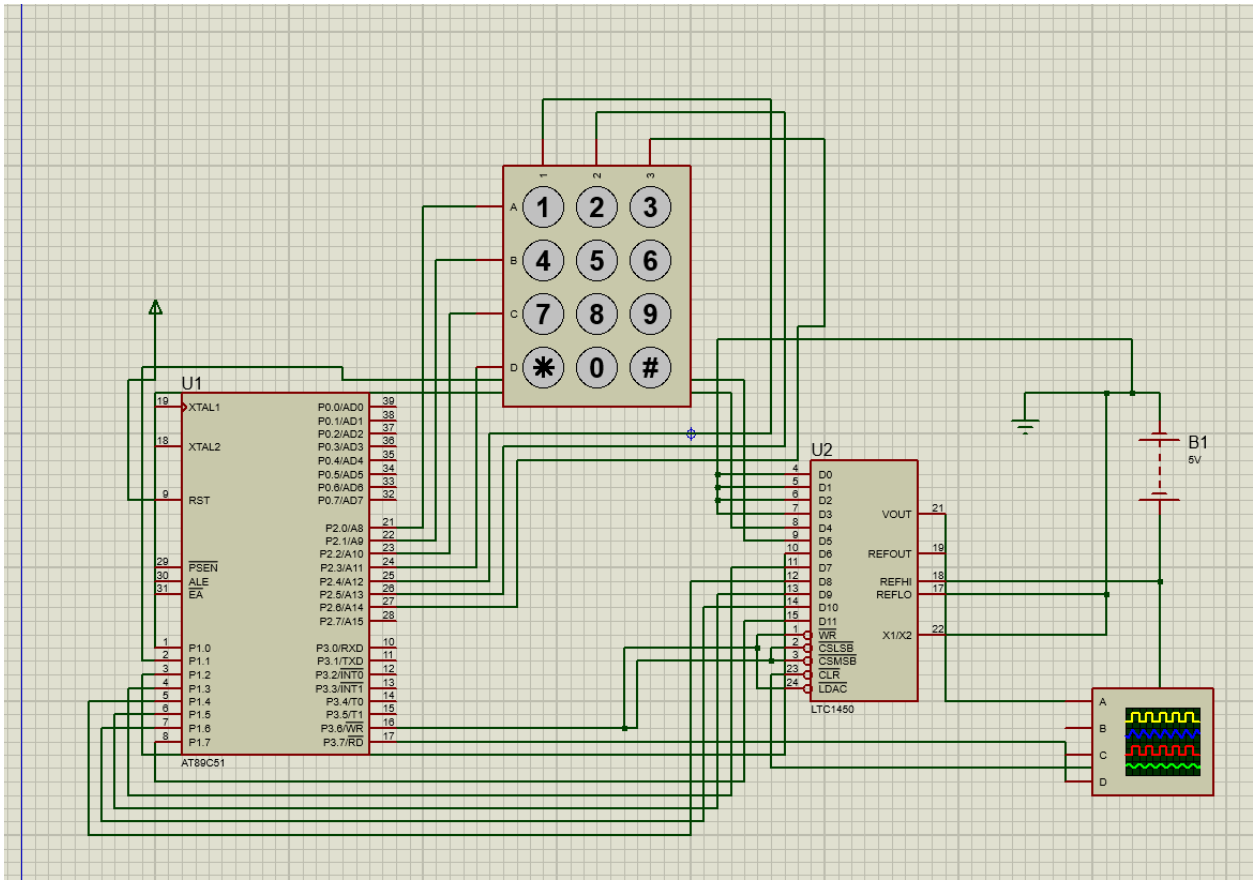

Рисунок 8 – Результат изменения частоты

ПРИЛОЖЕНИЕ Б

Код

```
#include <REG51.h>  // Подключаем заголовочный файл для работы с микроконтроллером 8051
#include <stdio.h>   // Подключаем стандартную библиотеку для ввода/вывода

#define FIXED_STEPS 255  // Константа, определяющая количество шагов для генерации сигнала (255 шагов)

unsigned char count;  // Счетчик для формирования сигнала
xdata unsigned char out;  // Выходной регистр для передачи данных на внешний периферийный компонент
unsigned char amplitude = 220;  // Амплитуда сигнала, по умолчанию 220
unsigned int delayTime = 5;  // Время задержки между изменениями сигнала, по умолчанию 5 миллисекунд
unsigned int mode0 = 0;  // Режим для ввода амплитуды (по умолчанию выключен)
unsigned int mode1 = 0;  // Режим для ввода частоты (по умолчанию выключен)

char keys[4][3] = {  // Массив для хранения значений клавиш на клавиатуре (матрица 4x3)
    {'1', '2', '3'},
    {'4', '5', '6'},
    {'7', '8', '9'},
    {'*', '0', '#'}
};

// Определения линий для работы с клавишами (строки и столбцы)
```

```
sbit r1 = P2^0;
sbit r2 = P2^1;
sbit r3 = P2^2;
sbit r4 = P2^3;

sbit c1 = P2^4;
sbit c2 = P2^5;
sbit c3 = P2^6;

// Функция задержки, используемая для временных пауз в миллисекундах
void delay(unsigned int ms) {
   unsigned int i, j;
   for (i = 0; i < ms; i++)  // Цикл для выполнения задержки
      for (j = 0; j < 120; j++);  // Увеличиваем время задержки, чтобы она была более заметной
}

// Функция для получения нажатой клавиши с клавиатуры
char getKey() {
   unsigned char row, col;

   for (row = 0; row < 4; row++) {  // Перебираем все строки
      r1 = r2 = r3 = r4 = 1;  // Устанавливаем все строки в высокий уровень
      switch (row) {  // Устанавливаем только одну строку в низкий уровень для сканирования
         case 0: r1 = 0; break;
         case 1: r2 = 0; break;
         case 2: r3 = 0; break;
```

```
            case 3: r4 = 0; break;
        }

        // Проверяем, какая клавиша нажата в текущей строке
        if (c1 == 0) { while (c1 == 0); return keys[row][0]; }
        if (c2 == 0) { while (c2 == 0); return keys[row][1]; }
        if (c3 == 0) { while (c3 == 0); return keys[row][2]; }
      }

      return 0;  // Возвращаем 0, если не была нажата никакая клавиша
    }

    // Функция для ввода амплитуды сигнала с клавиатуры
    void inputAmplitude() {
      unsigned char inputValue = 0;  // Переменная для хранения введенного значения амплитуды
      char key;

      while (1) {  // Цикл, пока не введена амплитуда
        key = getKey();  // Получаем нажатую клавишу
        if (key != 0) {  // Если клавиша нажата
          if (key >= '0' && key <= '9') {  // Если это цифра
            inputValue = (inputValue * 10) + (key - '0');  // Формируем число из цифр
          } else if (key == '*') {  // Если нажата клавиша '*'
            amplitude = inputValue;  // Присваиваем введенную амплитуду
            mode0 = 0;  // Отключаем режим ввода амплитуды
            break;  // Выходим из цикла
          }
        }
```

```
        }
    }

    // Функция для ввода частоты сигнала с клавиатуры
    void inputFrequency() {
        unsigned int inputValue = 0;  // Переменная для хранения введенного значения частоты
        char key;

        while (1) {  // Цикл, пока не введена частота
            key = getKey();  // Получаем нажатую клавишу
            if (key >= '0' && key <= '9') {  // Если это цифра
                inputValue = (inputValue * 10) + (key - '0');  // Формируем число из цифр
            } else if (key == '#') {  // Если нажата клавиша '#'
                delayTime = inputValue;  // Присваиваем введенную частоту
                mode1 = 0;  // Отключаем режим ввода частоты
                break;  // Выходим из цикла
            }
        }
    }

    void main(void) {
        char key;

        while (1) {  // Главный цикл программы
            // Генерация сигнала по фиксированным шагам
            for (count = 0; count <= FIXED_STEPS; count++) {
                key = getKey();  // Получаем нажатую клавишу с клавиатуры
                if (key == keys[3][0]) {  // Если нажата клавиша '*'
```

```
                mode0 = 1;  // Включаем режим ввода амплитуды
            } else if (key == keys[3][2]) {  // Если нажата клавиша '#'
                mode1 = 1;  // Включаем режим ввода частоты
            }

            P1 = (count * amplitude) / FIXED_STEPS;  // Генерируем выходной сигнал в зависимости от амплитуды
            out = P1;  // Выводим сигнал на внешний выход
            delay(delayTime);  // Пауза между шагами

            // Проверка, нужно ли вводить амплитуду или частоту
            if (mode0 == 1) {
                inputAmplitude();  // Вводим амплитуду
            } else if (mode1 == 1) {
                inputFrequency();  // Вводим частоту
            }
        }

        P1 = 0;  // Останавливаем сигнал
        delay(delayTime);  // Пауза после завершения генерации сигнала

        // Сброс режима ввода
        mode0 = 0;
        mode1 = 0;
    }
}
```