Министерство образования Республики Беларусь

Учреждение образования "Белорусский государственный университет информатики и радиоэлектроники"

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе 1

По дисциплине: Модели решения задач в интеллектуальных системах

На тему: Вычисления попарного произведения (деления) компонентов двух векторов чисел

Выполнил ст. гр. 221731: Ашурбаева С.М.

Проверил: Ивашенко В.П.

Минск 2025

# Лабораторная работа по дисциплине

# Модели решения задач в интеллектуальных системах

# Выполнино студентами группы 221731 БГУИР Ашурбаева С.М.

# Этот файл содержит реализацию базовой системы.

# Вариант 5

# был сделан код который решает алгоритм вычисления целочисленного частного пары 4-разрядных

# чисел делением с восстановлением частичного остатка.

# 02.04.2025 дата окончания проекта

# выполнение математическая операций.

import matplotlib.pyplot as plt

def int\_to\_bin(num, width=4):

    """Возвращает двоичную строку длины width для числа num."""

    return format(num & ((1 << width) - 1), f'0{width}b')

def bin\_to\_int(bin\_str):

    """Преобразует двоичную строку в целое число."""

    return int(bin\_str, 2)

def pipeline\_divide(bin\_dividend, bin\_divisor, width=4):

    """

    Деление с конвейерным алгоритмом восстановления остатка для двух 4-битных чисел.

    Возвращает список этапов (описание, двоичное остатка, двоичное частного),

    а также итоговое q и r.

    """

    dividend = bin\_to\_int(bin\_dividend)

    divisor = bin\_to\_int(bin\_divisor)

    if divisor == 0:

        raise ValueError("Деление на ноль не допускается!")

    remainder = 0

    quotient = 0

    steps = []

    for i in range(width):

        # Сдвигаем остаток и подаем следующий старший бит делимого

        current\_bit = (dividend >> (width - 1 - i)) & 1

        remainder = (remainder << 1) | current\_bit

        desc = f"Этап {i+1}: бит={current\_bit}"

        # Восстановление остатка

        if remainder >= divisor:

            before = remainder

            remainder -= divisor

            q\_bit = 1

            desc += f", {before} - {divisor} = {remainder}"

        else:

            q\_bit = 0

            desc += f", не вычитаем (остаток={remainder})"

        quotient = (quotient << 1) | q\_bit

        steps.append((desc,

                      format(remainder, f'0{width+1}b'),

                      format(quotient, f'0{width}b')))

    # Финальный этап

    steps.append(("Завершение",

                  format(remainder, f'0{width+1}b'),

                  format(quotient, f'0{width}b')))

    return steps, quotient, remainder

def main():

    print("Конвейерное деление 4-битных чисел с восстановлением остатка")

    try:

        n = int(input("Введите количество пар (каждая пара: делимое и делитель от 0 до 15): "))

    except ValueError:

        print("Ошибка: введите целое число.")

        return

    results = []

    for idx in range(1, n+1):

        try:

            a = int(input(f"\nПара {idx}. Делимое A (0–15): "))

            b = int(input(f"Пара {idx}. Делитель B (1–15): "))

        except ValueError:

            print("Ошибка: введите целое число.")

            return

        if not (0 <= a < 16 and 1 <= b < 16):

            print("Ошибка: числа должны быть в диапазоне 0–15 (делитель не ноль).")

            return

        bin\_a = int\_to\_bin(a, 4)

        bin\_b = int\_to\_bin(b, 4)

        print(f"A = {bin\_a} ({a}), B = {bin\_b} ({b})")

        steps, q, r = pipeline\_divide(bin\_a, bin\_b, 4)

        for desc, rem\_bin, q\_bin in steps:

            print(f"{desc} | остаток: {rem\_bin} | частное (bin): {q\_bin}")

        print(f"Результат: частное = {int\_to\_bin(q,4)} ({q}), остаток = {format(r,'05b')} ({r})")

        results.append((idx, q, r))

    # Построение графика

    xs = [x[0] for x in results]

    ys\_q = [x[1] for x in results]

    ys\_r = [x[2] for x in results]

    plt.figure(figsize=(8, 4))

    plt.plot(xs, ys\_q, marker='o', label="Частное")

    plt.plot(xs, ys\_r, marker='s', label="Остаток")

    plt.title("Результаты деления 4-битных чисел")

    plt.xlabel("Номер пары")

    plt.ylabel("Значение")

    plt.xticks(xs)

    plt.legend()

    plt.grid(True)

    plt.tight\_layout()

    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Вывод:

