МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное   
учреждение высшего образования

«Чувашский государственный университет имени И. Н. Ульянова»

Факультет информатики и вычислительной техники

Кафедра вычислительной техники

# **Лабораторная работа 3.**

**АРХИТЕКТУРА И ПРОГРАММИРОВАНИЕ АРИФМЕТИЧЕСКОГО СОПРОЦЕССОРА**

Выполнил: студент группы ИВТ-42-23

Назаров Ярослав Сергеевич

Проверил: доцент Андреева Антонина Аркадьевна

Чебоксары, 2025

Цель работы: Изучить архитектуру сопроцессора, понять его систему команд и освоить программирование математических вычислений на ассемблере.

**Текст задания:**

Найти корни квадратного трехчлена с заданными комплексными коэффициентами.

Программа на языке высокого уровня:

import cmath

# Сопроцессорный стек

fpu\_stack = []

def fpu\_push(value):

    """Помещает значение в стек сопроцессора (эмуляция FLD)"""

    fpu\_stack.append(value)

def fpu\_pop():

    """Извлекает значение из стека сопроцессора (эмуляция FSTP)"""

    return fpu\_stack.pop()

def fpu\_add():

    """Сложение ST(0) + ST(1) -> ST(1); Pop"""

    b = fpu\_pop()

    a = fpu\_pop()

    fpu\_push(a + b)

def fpu\_sub():

    """Вычитание ST(1) - ST(0) -> ST(1); Pop"""

    b = fpu\_pop()

    a = fpu\_pop()

    fpu\_push(a - b)

def fpu\_mul():

    """Умножение ST(0) \* ST(1) -> ST(1); Pop"""

    b = fpu\_pop()

    a = fpu\_pop()

    fpu\_push(a \* b)

def fpu\_div():

    """Деление ST(1) / ST(0) -> ST(1); Pop"""

    b = fpu\_pop()

    a = fpu\_pop()

    fpu\_push(a / b)

def fpu\_sqrt():

    """Корень из ST(0) -> ST(0)"""

    a = fpu\_pop()

    fpu\_push(cmath.sqrt(a))

# Коэффициенты квадратного трехчлена: az^2 + bz + c = 0

a = complex(1, 2)

b = complex(3, -4)

c = complex(2, 1)

# Вычисляем дискриминант: D = b^2 - 4ac

fpu\_push(b)     # ST = b

fpu\_push(b)     # ST = b, b

fpu\_mul()       # ST = b\*b

fpu\_push(4)     # ST = b\*b, 4

fpu\_push(a)     # ST = b\*b, 4, a

fpu\_mul()       # ST = b\*b, 4a

fpu\_push(c)     # ST = b\*b, 4a, c

fpu\_mul()       # ST = b\*b, 4ac

fpu\_sub()       # ST = D = b\*b - 4ac

# Извлекаем дискриминант

D = fpu\_pop()

# Вычисляем корень из дискриминанта

fpu\_push(D)

fpu\_sqrt()

sqrt\_D = fpu\_pop()

# Вычисляем -b

fpu\_push(-b)

# x1 = (-b + sqrt(D)) / (2a)

fpu\_push(sqrt\_D)

fpu\_add()

fpu\_push(2)

fpu\_push(a)

fpu\_mul()

fpu\_div()

x1 = fpu\_pop()

# x2 = (-b - sqrt(D)) / (2a)

fpu\_push(-b)

fpu\_push(sqrt\_D)

fpu\_sub()

fpu\_push(2)

fpu\_push(a)

fpu\_mul()

fpu\_div()

x2 = fpu\_pop()

print(f"x1 = {x1}")

print(f"x2 = {x2}")

**Текстовая последовательная реализация программы:**

**Шаги алгоритма:**

1. **Вычисление дискриминанта D = b² - 4ac**
   * Рассчитать b²:
     + Действительная часть: b\_sq\_re = (b\_re)² - (b\_im)²
     + Мнимая часть: b\_sq\_im = 2 \* b\_re \* b\_im
   * Рассчитать a\*c:
     + Действительная часть: ac\_re = a\_re\*c\_re - a\_im\*c\_im
     + Мнимая часть: ac\_im = a\_re\*c\_im + a\_im\*c\_re
   * Умножить a\*c на 4:
     + four\_ac\_re = 4 \* ac\_re
     + four\_ac\_im = 4 \* ac\_im
   * Получить D:
     + D\_re = b\_sq\_re - four\_ac\_re
     + D\_im = b\_sq\_im - four\_ac\_im
2. **Вычисление квадратного корня из D**
   * Перевести D в полярные координаты:
     + Модуль: |D| = sqrt((D\_re)² + (D\_im)²)
     + Аргумент: θ = arctan2(D\_im, D\_re) (угол в комплексной плоскости)
   * Вычислить корень из D:
     + Модуль корня: sqrt\_mod = sqrt(|D|)
     + Аргумент корня: θ/2
     + Действительная часть: sqrt\_re = sqrt\_mod \* cos(θ/2)
     + Мнимая часть: sqrt\_im = sqrt\_mod \* sin(θ/2)
3. **Формирование числителей для корней**
   * Для первого корня (+√D):
     + root1\_num\_re = -b\_re + sqrt\_re
     + root1\_num\_im = -b\_im + sqrt\_im
   * Для второго корня (-√D):
     + root2\_num\_re = -b\_re - sqrt\_re
     + root2\_num\_im = -b\_im - sqrt\_im
4. **Вычисление знаменателя 2a**
   * denom\_re = 2 \* a\_re
   * denom\_im = 2 \* a\_im
5. **Деление числителей на знаменатель**  
   Для каждого корня:
   * Вычислить квадрат модуля знаменателя:  
     denom\_mod\_sq = (denom\_re)² + (denom\_im)²
   * Действительная часть корня:

root\_re = (root\_num\_re \* denom\_re + root\_num\_im \* denom\_im) / denom\_mod\_sq

* + Мнимая часть корня:

root\_im = (root\_num\_im \* denom\_re - root\_num\_re \* denom\_im) / denom\_mod\_sq

Вывод: Изучена архитектура сопроцессора, понята его система команд и освоено программирование математических вычислений на ассемблере.