НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №1.4

з дисципліни“Архітектура комп’ютерів – 3”

Виконав:

студент групи ІВ-83

ЗК ІВ-8321

Поліщук Даніїл

Київ 2021

**Тема**: РОЗРОБЛЕННЯ ПРОГРАМ ОБРОБКИ ДАНИХ ДЛЯ ПРОЦЕСОРНОГО ЯДРА CORTEX М4

**Мета**: Вивчення архітектурних особливостей, системи команд, принципів організації команд умовних та безумовних переходів та переходів на підпрограми, команд роботи з пам’яттю та способів адресації операндів.

**8321 = 0b10000010000001**

**h6 = 0**

**h5 = 0**

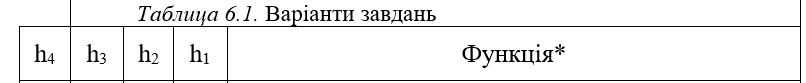
**h4 = 0**

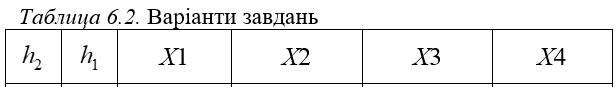
**h3 = 0**

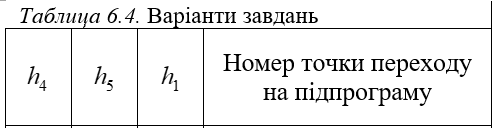
**h2 = 0**

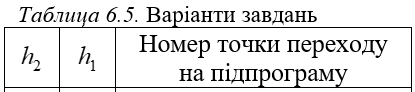
**h1 = 1**

**Варіант:**

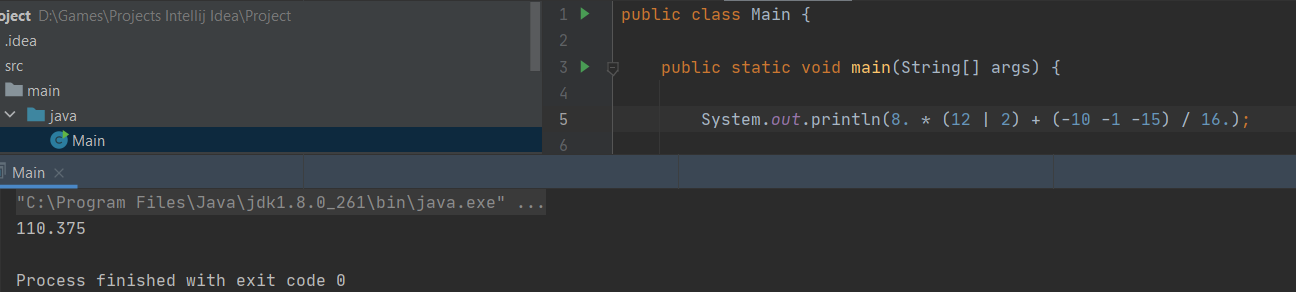
**** ****

**** ****

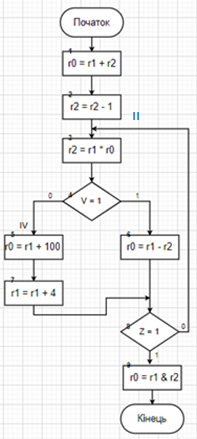
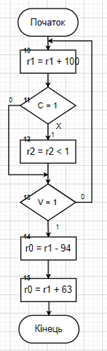
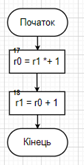
 

**** ****

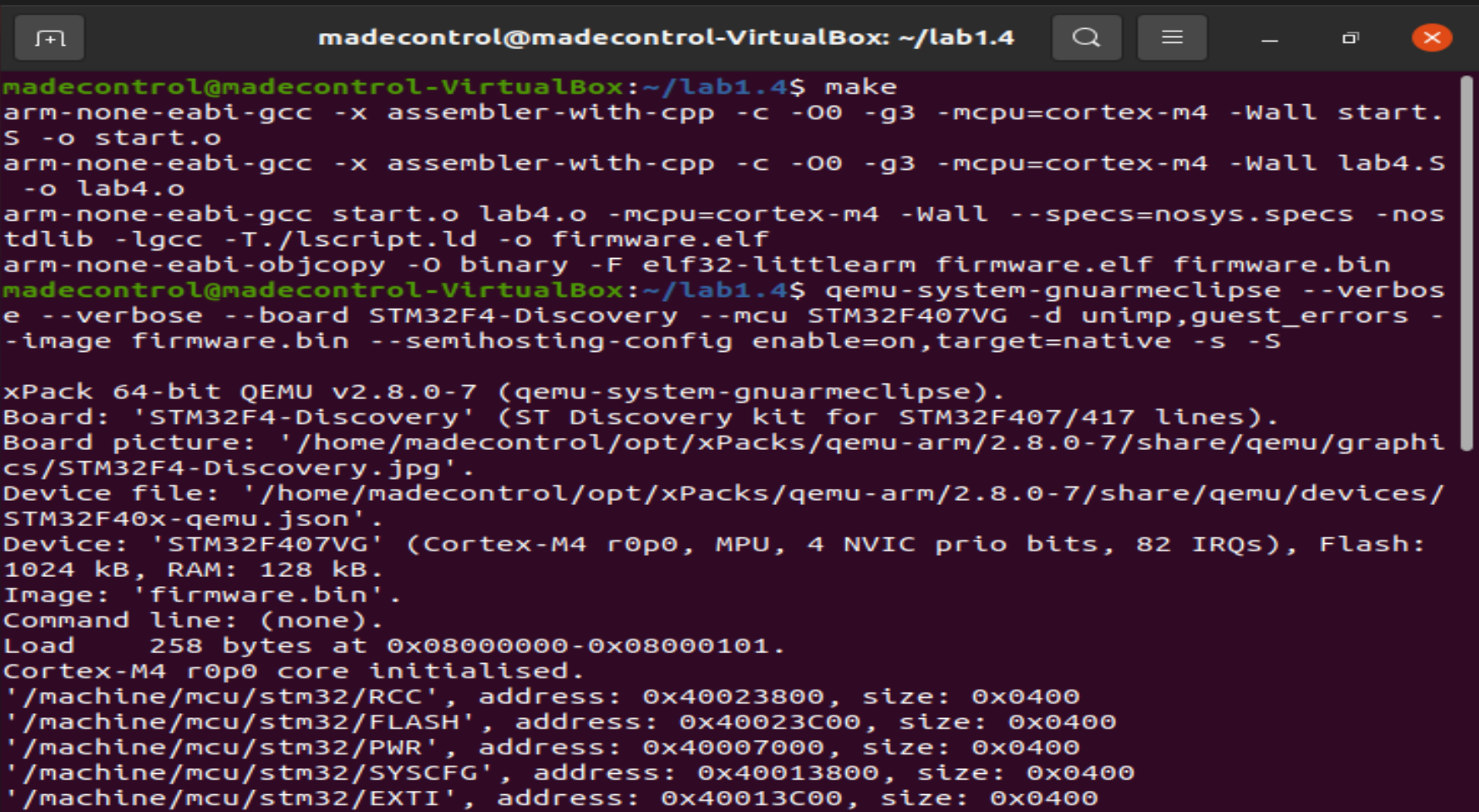
**Роз’язок функції із підставленими значеннями Х1, Х2, Х3, Х4, згідно варіанту**

****

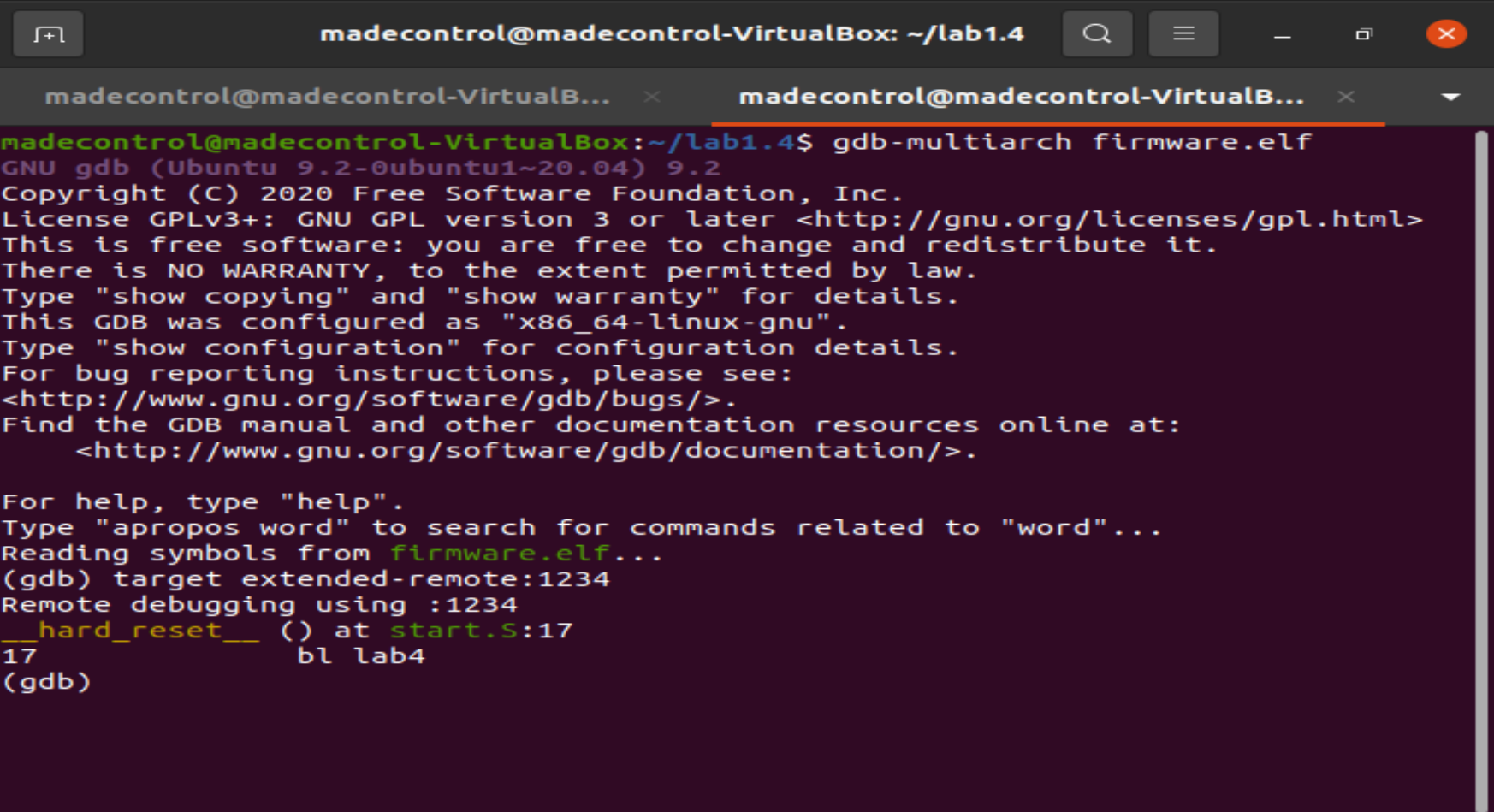
**Алгоритм:**

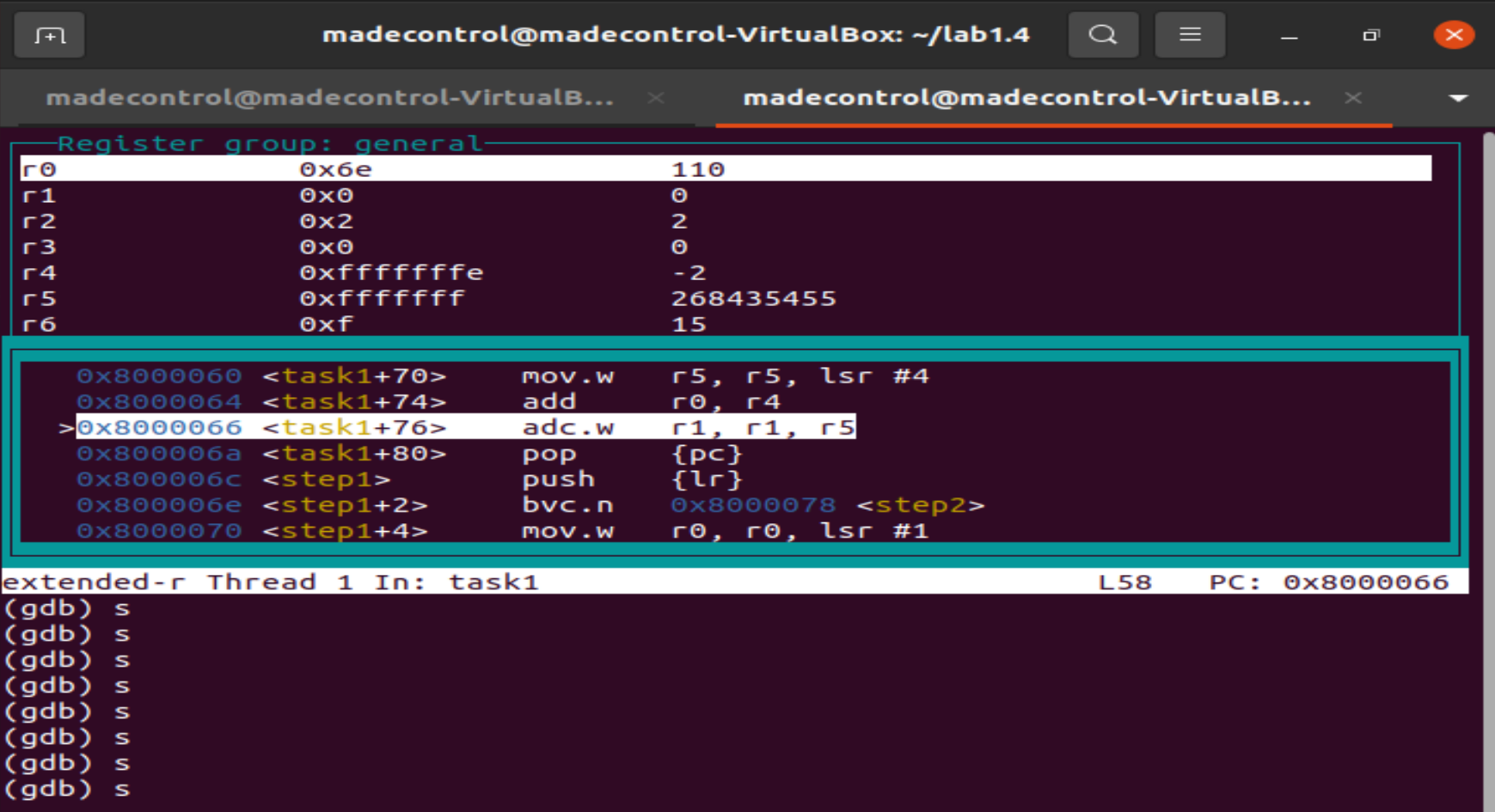
**  **

**Скріншоти роботи:**









**Лістинг коду**

1. **lab4.S**

.global lab4

.syntax unified

#define X1 #12

#define X2 #2

#define X3 #-10

#define X4 #15

#define RB #0b10000001

lab4:

push {lr}

bl task1

bl task2

pop {pc}

task1:

push {lr}

mov r0, X1 // значение x1

mov r1, #0

mov r2, X2 // значение x2

mov r3, #0

mov r4, X3 // значение x3

mov r5, #-1

mov r6, X4 // значениеx x4

mov r7, #0

// calc

// F = 8 \* (12 | 2) + (-10 - 1 - 15) / 16 = 110.375;

// F = 8 \* (x1 | x2) + (x3 - 1 - x4) / 16;

// (x1 or x2)

orr r0, r2

orr r1, r3

// 8 \* (x1 | x2)

lsl r0, #3

lsl r1, #3

// x3 - 1

sub r4, #1

sbc r5, #1

// (x3 - 1 - x4)

sub r4, r6

sbc r5, r7

// (x3 - 1 - x4) / 16

asr r4, #4

lsr r5, #4

// 8 \* (x1 or x2) + (x3 - 1 - x4) / 16;

add r0, r4

adc r1, r5

pop {pc}

// Correction

step1:

push {lr}

bvc step2

lsr r0, #1

lsr r1, #1

step2:

orr r0, #0

bne step4

mov r1, RB

step4:

add r1, RB

and r5, r1, #0xc0000000

beq step7

eor r5, #0xc0000000

beq step7

lsr r0, #1

lsr r1, #1

step7:

bvs endc

push {r0}

push {r1}

pop {r1}

pop {r0}

endc:

pop {pc}

task2:

push {lr}

mov r1, 133

mov r2, 26

first\_chunk:

//1. r0 = r1 + r2

add r0, r1, r2

bl second\_chunk

//2. r2 = r2 - 1

sub r2, #1

b3:

//3. r2 = r1 \* r0

mul r2, r1, r0

//4. V = 1

bvs overflow\_6

//5. r0 = r1 + r2

add r0, r1, r2

//7. r0 = r1 + 4

add r1, r1, #4

bl condition\_8

//6. r0 = r1 - r2

overflow\_6:

sub r0, r1, r2

condition\_8:

//8. Z = 1

bne b3

//9. r0 = r1 & r2

and r0, r1, r2

pop {pc}

second\_chunk:

push {lr}

//10. r1 = r1 ^ r2

b10:

eor r1, r2

//11. C = 1

bls cond\_o

bl third\_chunk

//12. r0 = r1 + r2

add r0, r1, r2

//13. V = 1

cond\_o:

bvs b10

//14. r0 = r1 - 94

sub r0, r1, #94

//15. r0 = r1 + 63

add r0, r1, #63

pop {pc}

third\_chunk:

push {lr}

//17. r0 = r1 \* r2

mul r0, r1, r2

//18. r0 = r1 - 5

sub r0, r1, #5

pop {pc}

**Висновок**

Було створено програмний проект на мові асемблера та перевірено його виконання відлагоджувачем.

Було вивчено архітектурні особливості, системи команд, принципів організації команд умовних та безумовних переходів та переходів на підпрограми.

В результаті, отримано очікувані результати, що показано на скріншотах виконання програми.