НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

**Лабораторна робота №4**

з дисципліни **«**Архітектура комп’ютерів 3**»**

Виконав:

студент 3 курсу

групи ІВ-83

Головенець Руслан

Перевірив:

Нікольський С.С.

Київ 2021 р.

**Тема:** Розроблення програм обробки даних для процесорного ядра CORTEX M4

**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.

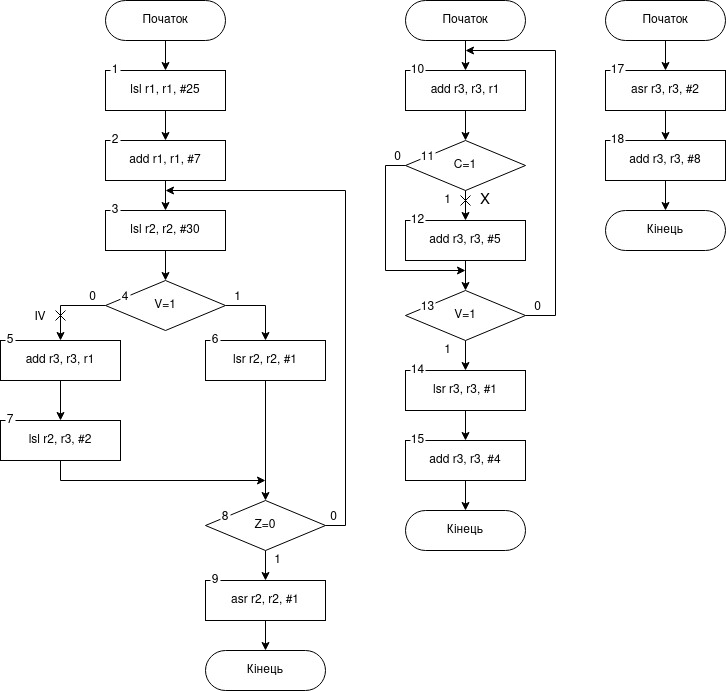
**Варіант:**

Номер залікової книжки: 8305  
8305 = 100000011100012

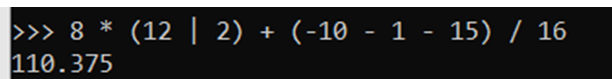
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | *Таблица 6.1.* Варіанти завдань | | | | | | | |
| h4 | h3 | | h2 | | h1 | Функція\* | | | | |
| 0 | 0 | | 0 | | 0 |  | | | | |
| 0 | 0 | | 0 | | 1 |  | | | | |
| 0 | 0 | | 1 | | 0 |  | | | | |
| 0 | 0 | | 1 | | 1 |  | | | | |
| 0 | 1 | | 0 | | 0 |  | | | | |
| 0 | 1 | | 0 | | 1 |  | | | | |
| 0 | 1 | | 1 | | 0 |  | | | | |
| 0 | 1 | | 1 | | 1 |  | | | | |
| 1 | 0 | | 0 | | 0 |  | | | | |
| 1 | 0 | | 0 | | 1 |  | | | | |
| 1 | 0 | | 1 | | 0 |  | | | | |
| 1 | 0 | | 1 | | 1 |  | | | | |
| 1 | 1 | | 0 | | 0 |  | | | | |
| 1 | 1 | | 0 | | 1 |  | | | | |
| 1 | 1 | | 1 | | 0 |  | | | | |
| 1 | 1 | | 1 | | 1 |  | | | | |
| *Таблица 6.2.* Варіанти завдань | | | | | | | | | |
|  | |  | | *X*1 | | | *X*2 | *X*3 | *X*4 |
| 0 | | 0 | | –17 | | | 12 | 17 | 3 |
| 0 | | 1 | | 12 | | | 2 | –10 | 15 |
| 1 | | 0 | | 18 | | | –9 | 23 | 11 |
| 1 | | 1 | | –9 | | | 10 | 31 | –21 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Таблица 6.4.* Варіанти завдань | | | | | | | | | |
|  | |  | |  | Номер точки переходу на підпрограму | | | | |
| 0 | | 0 | | 0 | І | | | | |
| 0 | | 0 | | 1 | ІІ | | | | |
| 0 | | 1 | | 0 | ІІІ | | | | |
| 0 | | 1 | | 1 | ІV | | | | |
| 1 | | 0 | | 0 | V | | | | |
| 1 | | 0 | | 1 | VI | | | | |
| 1 | | 1 | | 0 | VII | | | | |
| 1 | | 1 | | 1 | VIII | | | | |
| *Таблица 6.5.* Варіанти завдань | | | | | | | |
|  |  | | Номер точки переходу на підпрограму | | | | |
| 0 | 0 | | IX | | | | |
| 0 | 1 | | X | | | | |
| 1 | 0 | | XI | | | | |
| 1 | 1 | | XII | | | | |
| *Таблица 6.6.* Варіанти завдань | | | | | | | | |
|  |  | | Умови переходу (ознаки результату виконання операції) | | | | | |
|  |  | | ***CD*1** | | | ***CD*2** | ***CD*3** | |
| 0 | 0 | | Z=1 | | | N=0 | C=1 | |
| 0 | 1 | | N=1 | | | Z=0 | V=1 | |
| 1 | 0 | | V=1 | | | C=1 | Z=1 | |
| 1 | 1 | | C=1 | | | N=0 | V=0 | |

Алгоритм



Очікуваний результат



*task1.S*

.global task1

.syntax unified

#define x1h #0x00000000 #define x1l #0x0000000C #define x2h #0x00000000 #define x2l #0x00000002 #define x3h #0xFFFFFFFF #define x3l #0xFFFFFFF6 #define x4h #0x00000000 #define x4l #0x0000000F #define ZK #0x00002071

task1:

push {lr} mov r1, x1h mov r0, x1l mov r3, x2h mov r2, x2l

//x1 or x2 orr r4, r2, r0 orr r5, r3, r1

//8(x1 or x2) r4, r5 eor r10, r10

lsls r4, r4, #1 adc r10, r10, #0 lsls r4, r4, #1 lsl r10, r10, #1 adc r10, r10, #0 lsls r4, r4, #1 lsl r10, r10, #1 adc r4, r4, r10 lsls r5, r5, #3 add r5, r5, r10

mov r1, x3h mov r0, x3l mov r3, x4h mov r2, x4l

//f = (x3 - 1 - x4) r6, r7 subs r6, r0, #1

sbc r7, r1, #0 subs r6, r6, r2 sbc r7, r7, r3

//f = (x3 - 1 - x4)/16 r6, r7 eor r10, r10

asrs r7, r7, #1 adc r10, r10, #0 asrs r7, r7, #1 lsl r10, r10, #1 adc r10, r10, #0 asrs r7, r7, #1 lsl r10, r10, #1 adc r10, r10, #0 asrs r7, r7, #1 lsl r10, r10, #1 adc r10, r10, #0 lsl r10, r10, #28 lsr r6, r6, #4 add r6, r6, r10

//f = 8(x1 or x2) + (x3 - 1 - x4)/16 adds r6, r6, r4

adcs r7, r7, r5

it vs

blVS \_correct

cmp r6, #0 it eq

blEQ \_addZK

mov r0, ZK add r7, r0

lsrs r0, r7, #30 it eq

blEQ \_cont cmp r0, #3 it eq

blEQ \_cont bl \_correct2

\_cont:

pop {pc}

\_correct:

push {lr} lsr r6, r6, #1

lsrs r7, r7, #1 adc r6, r6, #0

add r7, r7, #0x80000000 pop {pc}

\_addZK:

push {lr} mov r0, ZK lsl r0, r0, #16 mov r6, r0 pop {pc}

\_correct2:

push {lr} lsr r6, r6, #1

asrs r7, r7, #1 it cs

addCS r6, #0x80000000

pop {pc}

*task2.S*

.global task2

.syntax unified

#define a #5

#define b #8

task2:

push {lr}

mov r1, a mov r2, b

//program start

//step1

lsl r1, r1, #25

//step2

add r1, r1, #7

\_step3:

lsls r2, r2, #30

//step4 it vs

//step6

lsrVS r2, r2, #1 bVS \_step8

//else - go to sub-program bl subP1

//step5

add r3, r3, r1

//step7

lsls r2, r3, #2

\_step8:

it eq

blEQ \_step3

//step9

asr r2, r2, #1 pop {pc}

subP1:

push {lr} mov r3, #3

\_step10:

adds r3, r3, r1

blCC \_step13

bl subP2

//step12

add r3, r3, #5

\_step13:

it vc

blVC \_step10

//step14

lsr r3, r3, #1

//step15

add r3, r3, #4

pop {pc} subP2:

push {lr}

//step17

asr r3, r3, #2

//step18

add r3, r3, #8 pop {pc}

