НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА АСОІУ

Модульна контрольна робота №2

з дисципліни «Архітектура комп’ютерів»

На тему:

"ОБРОБКА ІНФОРМАЦІЇ В ЕОМ НА ПРОГРАМНОМУ, МІКРОПРОГРАМНОМУ І АПАРАТНОМУ РІВНЯХ"

Виконав:

студент 2 курсу ФІОТ

групи ІП-53

Сулима Олександр

Номер залікової книжки:

5318(10)= 1010001100101 (2)

Перевірив:

Клименко І. А.

Київ – 2016 р.

ЗМІСТ

Зміст 2

1 Завдання 3

2 Опис архітектури і схема алгоритму обчислень 4

2.1 Модель програміста………………………………………………..…4

2.2 Адресний простір оперативної пам’яті (ОП)………………………5

2.3 Адресний простір зовнішніх пристроїв (ЗП)……………………….5

2.4 Схема алгоритму обчислень…………………………………………6

3 Система команд і програма обчислень 7

4 Структура системи і мікропрограма 8

Висновок 14

Список літератури 15

1 ЗАВДАННЯ

Розробити програму обчислення функції: A = (R15 + x) \* y, де операнди вводяться з пристрою введення. Результат (32 розряди) виводиться в пристрій виводу (послідовність даних, що вводяться і виводиться визначається розробником).

Система команд повинна забезпечувати обчислення заданого алгебраїчного вираження на програмному рівні. Операнди мають бути цілими 16-розрядними числами (один знаковий розряд і 15 основних розрядів). Операнди вводяться з пристрою введення в режимі програмного опитування готовності зовнішніх пристроїв. Результат виводитися в пристрій виводу. Кількість типів команд вибирається розробником, виходячи з необхідності реалізації заданих обчислень з урахуванням введення і виведення даних, а також перевірки готовності зовнішніх пристроїв. Введення кожного типу команд у систему повинне бути обґрунтовано. Зокрема, система команд повинна включати команди виконання заданих операцій, введення і виведення даних, команди умовних і безумовних переходів , пересилань даних.

В усіх виконуваних операціях використовується непряма адресація операндів. Множення чисел проводиться четвертим способом множення.

Адреса РС пристрою введення: 052h.

Адреса РД пристрою введення: 054h.

Адреса РС пристрою виведення: 0D2h.

Адреса РД пристрою виведення: 0D4h.

2 ОПИС АРХІТЕКТУРИ І СХЕМА АЛГОРИТМУ ОБЧИСЛЕНЬ

2.1 Модель програміста

Архітектура системи представляється за допомогою моделі програміста, структура якої показана в табл. 2.1.

*Таблиця 2.1 -* Модель програміста

|  |  |  |
| --- | --- | --- |
| Номер  регістру | Призначення регістрів | |
| R0 |  | Регістри загального призначення (РЗП) |
| R1 |  |
| R2 |  |
| R3 |  |
| R4 |  |
| R5 |  |
| R6 |  |
| R7 | Лічильник команд (ЛК) | Робочі регістри |
| R8 | Регістр команд (РК) |
| R9 | Регістр стану (РС) |
| R10 | Допоміжні регістри |
| R11 |
| R12 |
| R13 | Операнд |
| R14 | Адреса операнда |
| R15 | Регістр акумулятор (РА) |

2.2 Адресний простір оперативної пам’яті (ОП) Адресний простір оперативної пам’яті зображений на рис. 2.1.

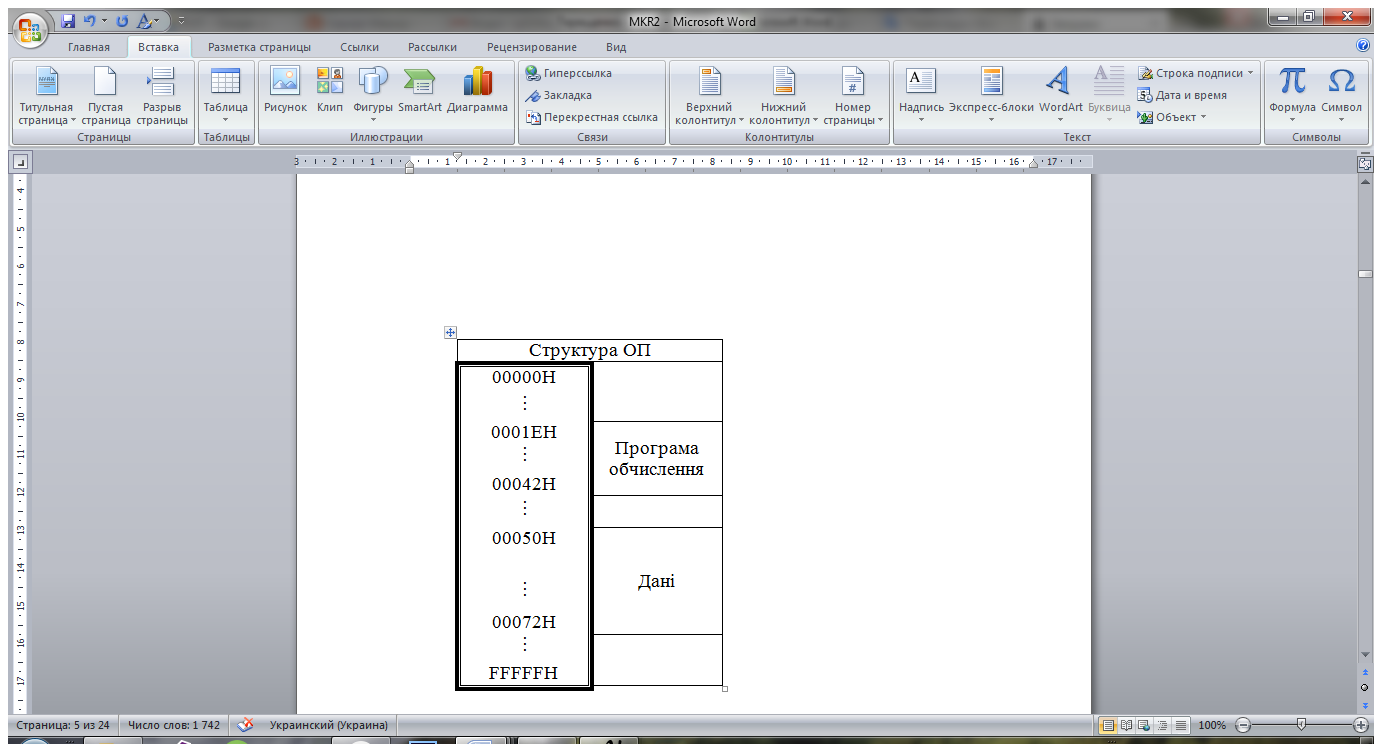
**

Рисунок 2.1 - Адресний простір ОП

2.3 Адресний простір зовнішніх пристроїв (ЗП)

Адресний простір зовнішніх пристроїв зображений на рис. 2.2.

|  |  |
| --- | --- |
| 0052h | РС |
| 0054h | РД |
|  | … |
| 0D2h | РС |
| 0D4h | РД |
|  | … |

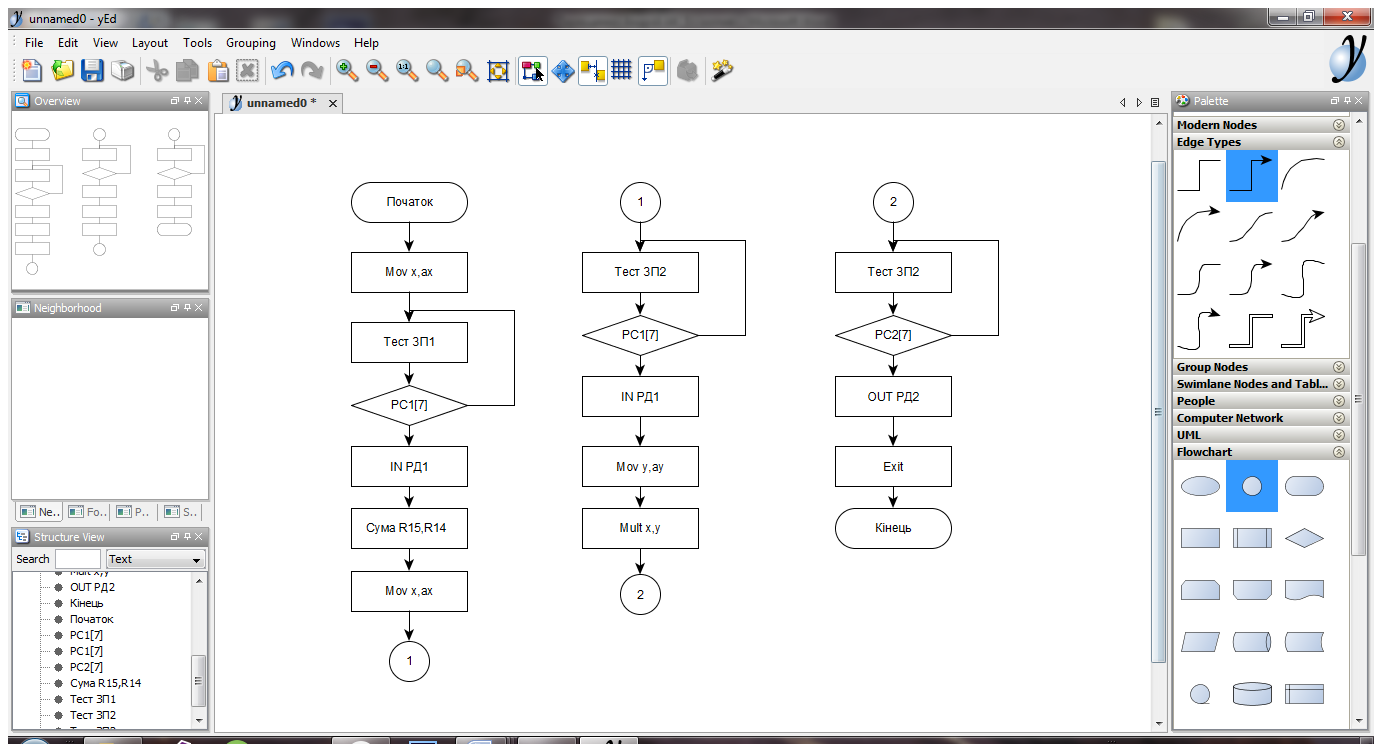
ЗПВ

##### ЗПВив

##### Рисунок 2.2 – Адресний простір ЗП

2.4 Схема алгоритму обчислень

Блок - схема алгоритму обчислень зображена на рис. 2.3.

**

##### Рисунок 2.3 – Алгоритм обчислень

3 Система команд і програма обчислень

Узагальнена схема формату команд зображена на рис.3.1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Одноадресна команда | | | | | |
| Ф | | Операція | ТА | Адреса  в ОП |  |
| 15 | | 14……...11 | 10 | 9………0 |  |
|  |  |  |  |  |  |
|  |  | Формат команди | |  |  |
| 0 | | Одноадресна | |  |  |
| 1 | | Двохадресна | |  |  |

Рисунок 3.1 - Формат команд

Мнемоніки та коди всіх операцій, необхідних для обчислення функції перелічені в таблиці 3.1.

Табл. 3.1. Система команд (для реалізації обчислень потрібні такі команди)

|  |  |  |
| --- | --- | --- |
| Команда | Код операції | Операція |
| Swap ax,reg | 0 0 0 1 | Обмін регістрів |
| jz [mem] | 0 0 1 0 | Умовний перехід |
| test ax,[mem] | 0 0 1 1 | Перевірка готовності ЗП |
| out [mem],ax | 0 1 0 0 | Запис у ЗП |
| in ax,[mem] | 0 1 0 1 | Зчитування з ЗП |
| mov [mem],ax | 0 1 1 0 | Пересилання з пам’яті до акумулятора |
| mul [mem],ax | 0 1 1 1 | Множення |
| Sum | 1 0 0 0 | Сума двох операндів |
| Exit | 1 0 0 1 | Вихід |

4 СТРУКТУРА СИСТЕМИ І МІКРОПРОГРАМА

На рис.4.1 представлена спрощена структура системи на базі мікропрограмувального комплекту ВІС серії 1804 (аналог Am2900 фірми AMD), який дозволяє реалізувати будь-яку систему команд (Intel, DEC, Motorola, Ziploc та ін.) і забезпечити необхідну розрядність процесора.

До складу процесора входять арифметико-логічний пристрій (АЛУ), блок мікропрограмного керування (БМК) і блок регістра стану (БРС).

АЛП містить арифметико-логічний блок (АЛБ) і надоперативний запам'ятовуючий пристрій (НОЗП), до складу якого входять 16 регістрів (R0...R15). Інформація із НОЗП може видаватися одночасно по двох каналах (А и В). Вибір регістрів здійснюється подачею адрес RA і RB на відповідні входи НОЗП. Адреси можуть видаватися з регістрів RA і RB або безпосередньо з БМК, минаючи зазначені регістри (відповідні зв'язки на  
рис. 4.1 умовно не показані).

АЛБ виконує мікрооперації, зазначені а табл.4.1, де через R і S позначені операнди, а через CI - вхідний перенос у молодший розряд. В якості операндів R і S може використовуватися інформація на локальній шині (BUS\_D), а також з регістрів НОЗП по каналах А и В.

Таблиця 4.1 - Мікрооперації АЛБ

|  |  |
| --- | --- |
| Мнемоніка | Мікрооперація в АЛБ |
| ADD | R+S+CI |
| SUB | R-S-1+CI |
| OR | R or S |
| AND | R and S |
| XOR | R xor S |

Ознаки можуть бути записані у відповідні розряди регістра RM та RN, що входить до складу БРС. Ознаки, записані в RM, позначаються як RM\_C, RM\_Z, RM\_N, RM\_V, а ознаки записані в RN – RN\_C, RN\_Z, RN\_N, RN\_V. Логічні ланцюги БРС забезпечують підключення до входу CI АЛБ прямого або інверсного значення ознаки RN\_C, а також сигналів логічного нуля або одиниці. Біт RM\_C бере участь також у деяких мікроопераціях зсуву.

ША

ШД

**ОП**

# ВУ

RDD

RDM

R

W

I

O

# PA

# БД

# RA

# RB

М Р

L1

L2

L3

L4

L5

L6

# БМУ

ЛШ

EWL

EWH

BUS\_D [15…0]

R

W

EWL

I

O

EWH

OEY

RDM

RDD

CT

# БРС

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RM | C | Z | V | N |
| RN | C | Z | V | N |

# Сдв

**Буф Y**

# АЛБ

B

# СОЗУ

(R0-R15)

(RQ)

А В

МУ1

МУ2

EWA

EWB

AdA`

AdB`

Рисунок 4.1 - Структурна схема ЕОМ

5 ЛІСТИНГ МІКРОПРОГРАМИ

\------------Firstly Settings to sxem

LINK l1:ct

LINK l2:rdm

LINK l3:rdd

LINK ewh:16

LINK M:z,z,z,z,z,z,z,14,13,12,11,z

ACCEPT dev[1]:I,052h,054h,12,2

ACCEPT dev[2]:O,0D2h,0D4h,12,2

ACCEPT dev[3]:O,0E2h,0E4h,12,2

ACCEPT dev\_buf[1]:0fffh,0fffh

ACCEPT R7:001Eh

ACCEPT R10:03ffh

ACCEPT R15:000fh

\-----------Program in OP

DW 001Eh:0011110001110010% \mov [072h],r15

DW 0020h:0001110001010110% \test [0056h]

DW 0022h:0001010001011010% \jz [005Ah]

DW 0024h:0010110001010110% \in r15,[0056h]

DW 0026h:0100010001110010% \Sum r14,r15

DW 0028h:0011110001100100% \mov [0064h],r15

DW 002Ah:0001110001010110% \test [0056h]

DW 002Ch:0001010001011100% \jz [005Ch]

DW 002Eh:0010110001010110% \in r15,[0056h]

DW 0030h:0011110001100110% \mov [0066h],r15

DW 0032h:0100110001100100% \mul [0064h]

DW 0034h:0001110001011000% \test [0058h]

DW 0036h:0001010001100000% \jz [0060h]

DW 0038h:0010010001011000% \out [0058h]

DW 003Ah:0000110000000000% \swap r15,r13

DW 003Ch:0001110001011001% \test [0058h]

DW 003Eh:0001010001100010% \jz [0062h]

DW 0040h:0010010001011001% \out [0059h]

DW 0042h:0101010000000000% \exit

\--------------Date in OP

DW 0056h:006Ah \(IN)

DW 0058h:006Ch \(OUT)

DW 0059h:006Fh \(OUT)

DW 005Ah:0020h \ address povernenia

DW 005Ch:0028h

DW 0060h:0032h

DW 0062h:003Ah

DW 0064h:006Eh \(X)

DW 0066h:0070h \(Y)

DW 006Ah:0052h \IN

DW 006Ch:00D2h \OUT

DW 006Fh:00E2h \OUT

DW 006Eh:0000h \X

DW 0070h:0000h \Y

\--------KOP command

ORG 0h \first

{cjp nz,go;}

ORG 2h \adress swap r15,r13

{cjp nz,swap;}

ORG 4h \ jz [mem]

{cjp nz,zjump;}

ORG 6h \ test [mem]

{cjp nz,testdevice;}

ORG 8h \ out [mem],r15

{cjp nz,output;}

ORG 0Ah \ in r15,[mem]

{cjp nz,input;}

ORG 0Eh \ mov [mem],r15

{cjp nz,movToMem;}

ORG 10h \ Sum

{cjp nz,Sum;}

ORG 12h \ mult

{cjp nz,mul;}

ORG 14h \ exit

{cjp nz,exit;}

\-----------------------------

go

\Read command on Op

{or nil,r7,r7;ewl;oey;}

{xor nil,r7,r7;ewh;oey;}

{r;cjp rdm,cp;or r8,z,bus\_d;}

\Analis format command

{and nil,r8,8000h;load rm,flags;}

{cjp not rm\_z,exit;}

\Analis type of adress

{and nil,r8,400h;load rm,flags;}

{cjp rm\_z,exit;}

{and r14,r8,03ffh;}

\transiyion to microoperation

{or nil,r8,z;oey;JMAP;}

\Load from OP

FromMem

{or nil,r14,z;ewl;oey;}

{r;or r14,bus\_d,z;cjp rdm,cp;}

{or nil,r14,z;ewl;oey;}

{r;or r14,bus\_d,z;cjp rdm,cp;}

{crtn nz;}

\ Swap r15,r13

swap

{or r11,r15,z;}

{or r15,r13,z;}

{or r13,r11,z;}

{cjp nz,next;}

mul

{cjs nz,FromMem;}

{or r11,z,17;}

{xor nil,r14,r15; load rm,flags;}

{or nil,r14,r14; cjp not no ,ll1;}

{sub r14, 0,r14,nz;}

ll1 {or nil,r15,r15; cjp not no ,ll2;}

{xor r15,r15,8000h;}

ll2 {cjp not rm\_c,ll3;}

{add r13,r13,r14,z;}

ll3 {or srl,r13,r13,z;}

{or sr.9,r15,r15,z;}

{sub r11,r11,z,z; load rm, flags;cem\_c;cem\_n;}

{cjp not rm\_z, ll2;}

{cjp not rm\_n,fin;}

{xor r15,r15,0FFFFh;}

{xor r13,r13,0FFFFh;}

{add r15,r15,1,z;}

{add r13,r13,0,rm\_c;}

fin {cjp nz,next;}

\Sum r14,r15

Sum

{cjs nz,FromMem;}

{add r15,r15,r14,z;}

{cjp nz,next;}

\ mov [mem],15

movToMem

{or nil,r14,z;ewl;oey;}

{r;or r14,bus\_d,z;cjp rdm,cp;}

{or nil,r14,z;ewl;oey;}

{w;cjp rdm,cp;or nil,r15,z;oey;}

{cjp nz,next;}

\ jz [mem]

zjump

{cjp not rn\_z,no\_j;}

{cjs nz,FromMem;}

{sub r7,r14,1;}

no\_j {cjp nz,next;}

\ in r15,[mem]

input

{cjs nz,FromMem;}

{add r14,r14,2,z;}

{or nil,r14,z;ewl;oey;}

{i;cjp rdd,cp;or r15,bus\_d,z;}

{cjp nz,next;}

\ out [mem],r15

output

{cjs nz,FromMem;}

{add r14,r14,2,z;}

{or nil,r14,z;ewl;oey;}

{o;cjp rdd,cp;or nil,r15,z;oey;}

{cjp nz,next;}

\ test [mem]

testdevice

{cjs nz,FromMem;}

{or nil,r14,z;oey;ewl;}

{i;cjp rdd,cp;or r14,bus\_d,z;}

{or nil,r14,z;load rn,flags;}

{cjp nz,next;}

Next

{add r7,r7,2,z;}

{cjp nz,go;}

\ exit

exit {}

ВИСНОВОК

У ході виконання роботи був розглянутий процес обробки інформації в ЕОМ на різних рівнях – програмному, мікропрограмному й апаратному.

Апаратний рівень є найнижчим рівнем обробки інформації. На цьому рівні обробляються команди, що закладені в АЛБ. До таких команд відносяться команди, що виконують арифметичні і логічні операції, а також операції зсуву.

Мікропрограмний рівень є більш високим. У розробленій мікропрограмі на цьому рівні виконується обробка команд програмного рівня, тобто читання команди , аналіз структури команди, пересилання операндів з оперативної пам'яті в робочі регістри і виконання операцій над ними.

Програмний рівень є найвищим рівнем обробки інформації. На цьому рівні виконуються абстрактні команди, тобто які не виконуються в АЛБ, але які можна описати за допомогою мікрокоманд на мікропрограмному рівні.

СПИСОК ЛІТЕРАТУРИ

1. Конспект лекцій за курсом "Архітектура ЕОМ".
2. Жабін В.І., Жуков І.А., Ткаченко В.В., Клименко І.А. Мікропроцесорні системи – Київ, 2009. – 492 с.
3. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В., Прикладна теорія цифрових автоматів: Навч. Посібник. – К.:Книжкове вид-во НАУ, 2007. – 364 с.