НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дисципліни **«**Архітектура комп’ютерів – 2**»**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-21

Кузьменко Володимир

Київ – 2014 р.

Практична робота 3.1

РОЗРОБКА ПРОГРАМ ПЕРЕДАЧІ ДАНИХ У МК51

Мета роботи: Вивчення системи команд, форматів подання даних та способів адресації операндів; вивчення команд передачі управління, команд пересилки даних та команд вибору банків регістрів; отримання навиків розробки програм на мові асемблеру МК51.

Розробити програму для обчислення функції: F=(X1-X2)/4 + X1/X2, де . Ділення виконати першим способом (із зсувом залишку).

Операнди та результат повинні бути подані: X1 – ПК, X2 – ПК, F – ДК.

2 ПРОГРАМА

Convert\_To\_CF MACRO ADR

MOV A, ADR

CPL A

ADD A, #1

MOV ADR, A

ENDM

Convert\_To\_TF MACRO ADR

Convert\_To\_CF ADR

ENDM

X1 EQU 0C1h//0C1h

X2 EQU 064h//0E4h

Adr\_F EQU 28h

// For subtraction

Adr\_Rez\_Sub EQU 27h

// For division

Bit\_s1\_X1 EQU 0h

Bit\_s1\_X2 EQU 10h

Bit\_s1\_Rez\_Div EQU 20h

Sign\_div EQU 30h

Adr\_s\_X1 EQU 20h

Adr\_X1 EQU 21h

Adr\_s\_X2 EQU 20h

Adr\_X2 EQU 23h

Adr\_s\_Rez\_Div EQU 24h

Adr\_Rez\_Div EQU 25h

Acc0 EQU 0E0h

MOV Adr\_X1, #X1

MOV Adr\_X2, #X2

ACALL Subtraction

MOV P1, Adr\_Rez\_Sub

ACALL Division

MOV P1, Adr\_Rez\_Div

MOV A, Adr\_Rez\_Div

ADD A, Adr\_Rez\_Sub

MOV Adr\_F, A

MOV P1, Adr\_F

JMP Ended

Division:

PUSH Adr\_X2

PUSH Adr\_X1

MOV Adr\_Rez\_Div, #1h

JNB 0Fh, X1\_positive

SETB Sign\_div

CLR 0Fh

X1\_positive:

JNB 1Fh, X2\_positive

JB Sign\_div, sign\_plus

SETB Sign\_div

JMP label0

sign\_plus:

CLR Sign\_div

label0:

CLR 1Fh

X2\_positive:

CLR C

JB Bit\_s1\_X1, X1\_1

// X1 := X1 - X2

MOV A, Adr\_X1

SUBB A, Adr\_X2

MOV Adr\_X1, A

MOV A, Adr\_s\_X1

SUBB A, Adr\_s\_X2

ANL A, #00000001b

MOV Adr\_s\_X1, A

JMP shifting

X1\_1:

// X1 := X1 + X2

MOV A, Adr\_X1

ADD A, Adr\_X2

MOV Adr\_X1, A

MOV A, Adr\_s\_X1

ADD A, Adr\_s\_X2

ANL A, #00000001b

MOV Adr\_s\_X1, A

shifting:

CLR C

// X1 := l(X1).0

MOV A, Adr\_X1

RLC A

CLR Acc0

MOV Adr\_X1, A

JC c1

CLR Bit\_s1\_X1

CLR PSW.1

JMP label1

c1:

SETB Bit\_s1\_X1

SETB PSW.1

label1:

; REZ := l(REZ).NOT RGX[1]

CPL C

MOV A, Adr\_Rez\_Div

RLC A

JB PSW.1, c0

SETB Acc0

JMP label2

c0:

CLR Acc0

label2:

MOV Adr\_Rez\_Div, A

MOV A, Adr\_s\_Rez\_Div

RLC A

ANL Adr\_s\_Rez\_Div, #00000001b

MOV Adr\_s\_Rez\_Div, A

JNB Bit\_s1\_Rez\_Div, X2\_positive

POP Adr\_X1

POP Adr\_X2

JNB Sign\_div, end\_div

Convert\_To\_CF Adr\_Rez\_Div

end\_div:

RET

Subtraction:

PUSH Adr\_X2

PUSH Adr\_X1

JNB 0Fh, X1\_positiv

CLR 0Fh

Convert\_To\_CF Adr\_X1

X1\_positiv:

JNB 1Fh, X2\_positiv

CLR 1Fh

Convert\_To\_CF Adr\_X2

X2\_positiv:

MOV A, Adr\_X1

SUBB A, Adr\_X2

MOV Adr\_Rez\_Sub, A

JNB 3Fh, plus

Convert\_To\_TF Adr\_Rez\_Sub

SETB PSW.1

plus:

CLR C

RRC A

CLR C

RRC A

MOV Adr\_Rez\_Sub, A

JNB PSW.1, end\_sub

Convert\_To\_CF Adr\_Rez\_Sub

end\_sub:

POP Adr\_X1

POP Adr\_X2

RET

Ended:

NOP

END