Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

**Лабораторная работа №4 (вариант 11)**

по дисциплине: «Ассемблеры и их применение».

Выполнил:

студент 3 курса, гр. ИВТАПбд-31

Кондратьев Павел Сергеевич.

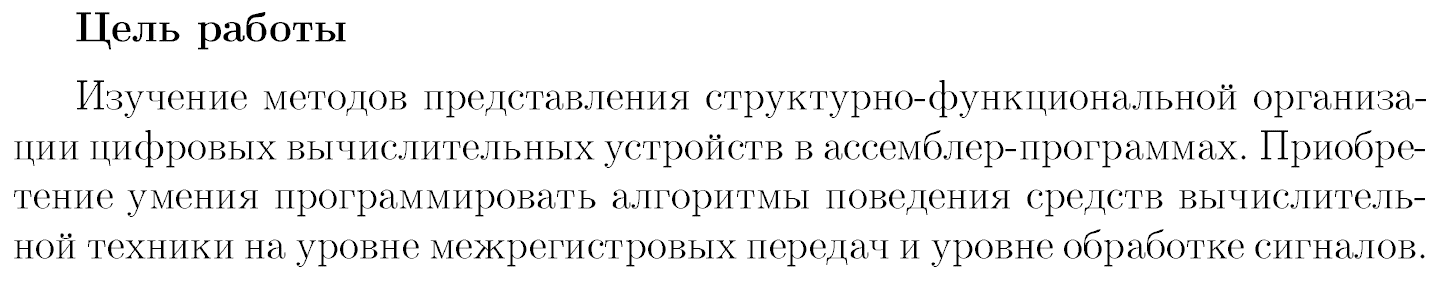
Проверил:

преподаватель кафедры ВТ

Лылова Анна Вячеславовна.

г. Ульяновск, 2019

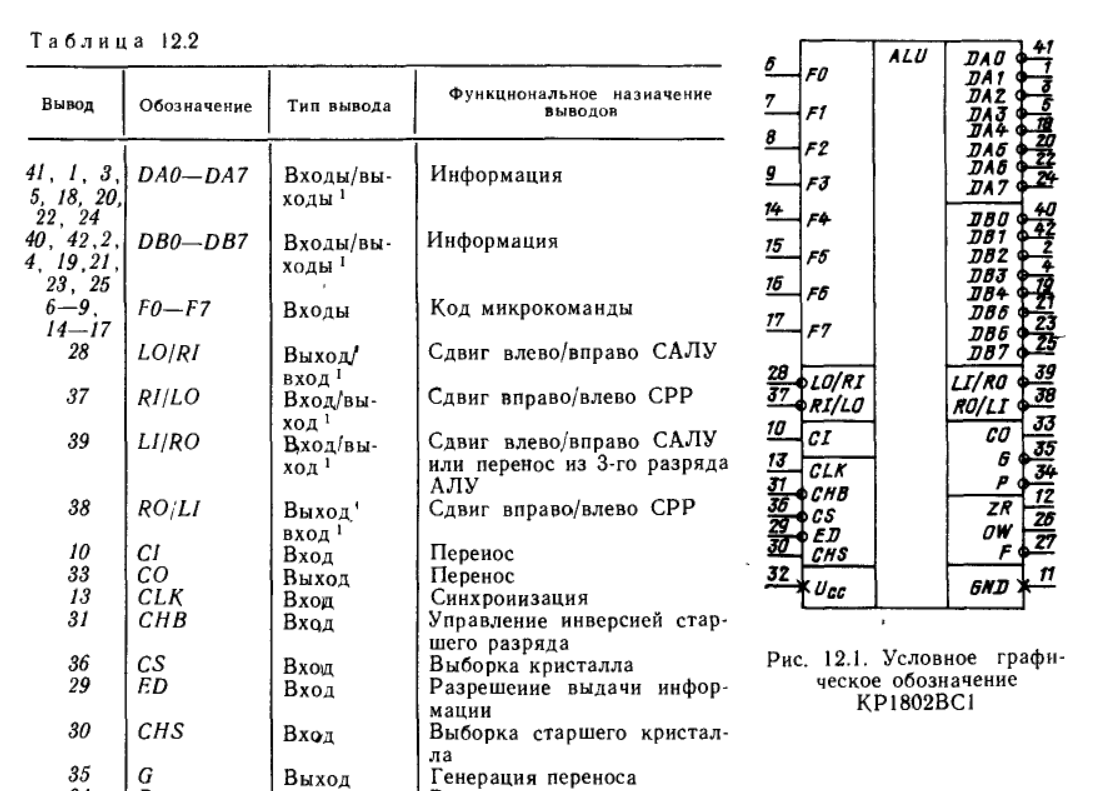
**1) Задание:**

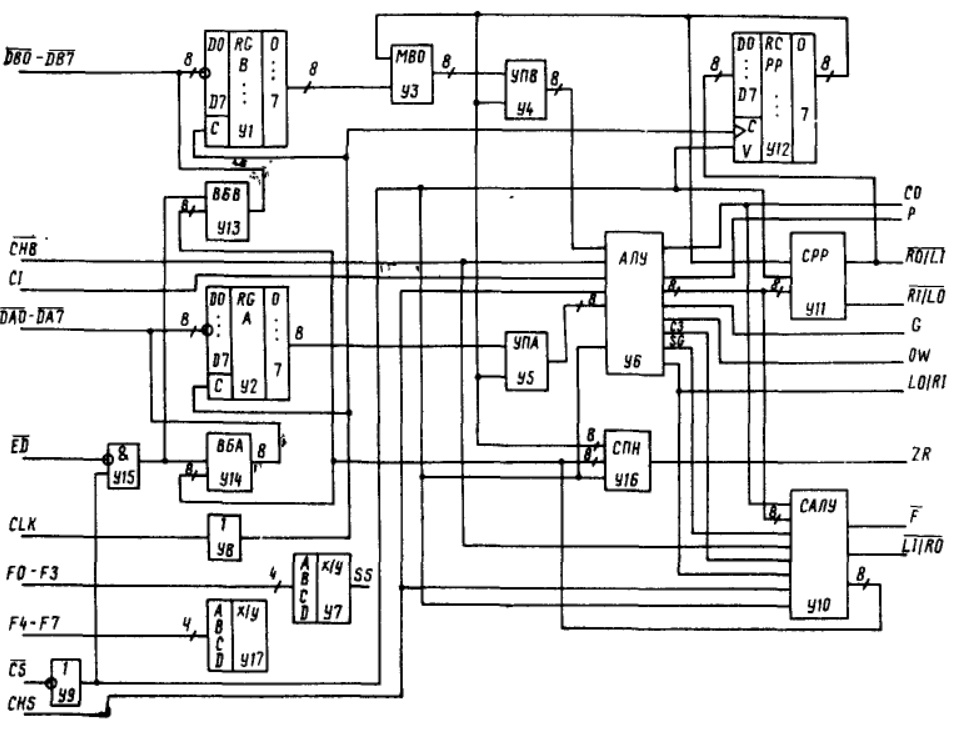


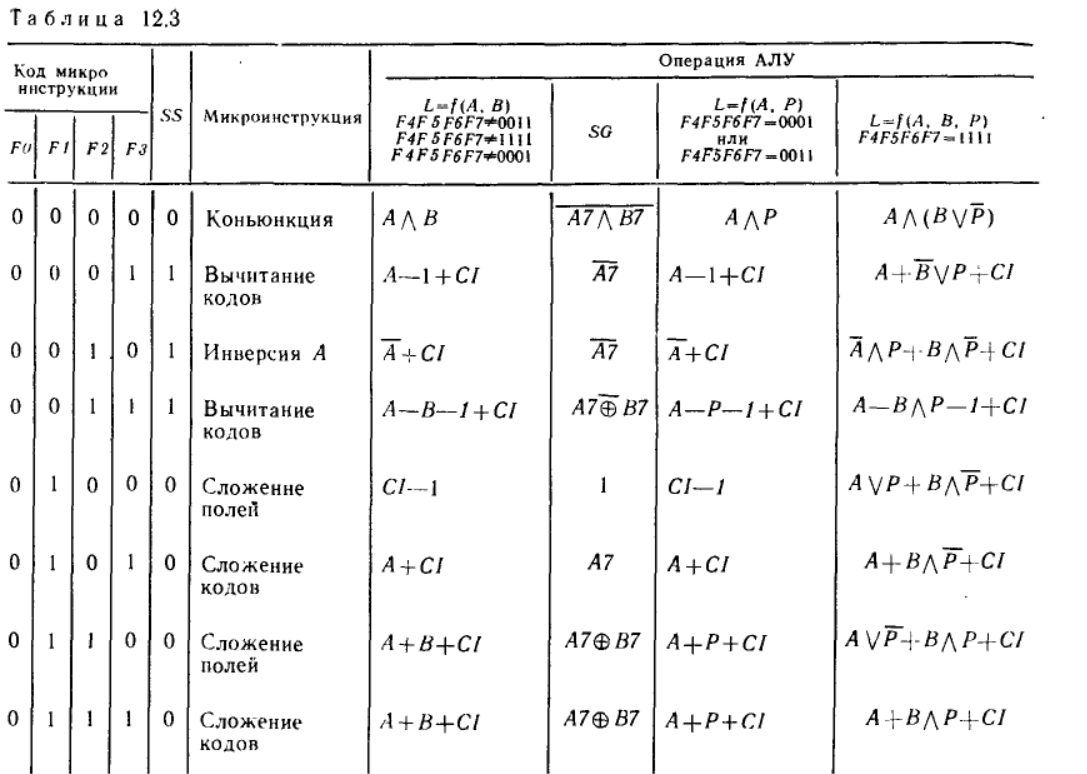
**Задание по варианту**



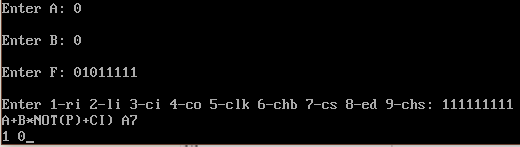
**Описание структуры заданного по заданию**

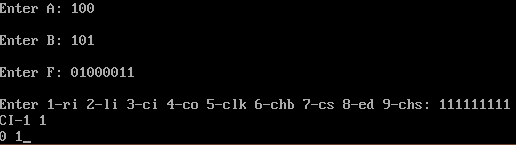


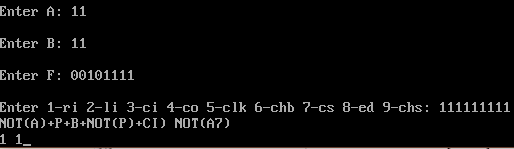


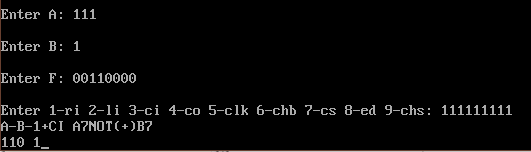


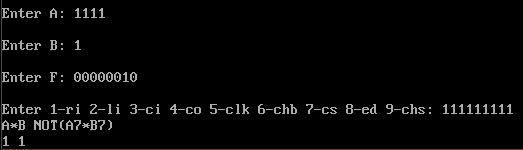
**2) Контрольные примеры**

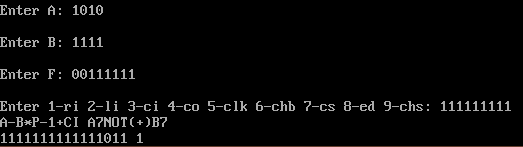












**Код программы:**

.MODEL SMALL

.STACK 200h

.386

; Декларации данных

.DATA

InputA db 13, 10, "Enter A: $"

InputB db 13, 10, "Enter B: $"

InputF db 13, 10, "Enter F: $"

MStep0\_3 db 13, 10, "A\*(B+P) NOT(A7\*B7)$"

MStep1\_3 db 13, 10, "A+NOT(B)+P+CI) NOT(A7)$"

MStep2\_3 db 13, 10, "NOT(A)+P+B+NOT(P)+CI) NOT(A7)$"

MStep3\_3 db 13, 10, "A-B\*P-1+CI A7NOT(+)B7$"

MStep4\_3 db 13, 10, "A+P+B\*NOR(P)+CI 1$"

MStep5\_3 db 13, 10, "A+B\*NOT(P)+CI) A7$"

MStep6\_3 db 13, 10, "A+NOT(P)+B\*P+CI A7(+)B7$"

MStep7\_3 db 13, 10, "A+B\*P+CI A7(+)B7$"

MStep0\_2 db 13, 10, "A\*P NOT(A7\*B7)$"

MStep1\_2 db 13, 10, "A-1+CI NOT(A7)$"

MStep2\_2 db 13, 10, "NOT(A)+P+B+NOT(P)+CI) NOT(A7)$"

MStep3\_2 db 13, 10, "A-B\*P-1+CI A7NOT(+)B7$"

MStep4\_2 db 13, 10, "A+P+B\*NOR(P)+CI 1$"

MStep5\_2 db 13, 10, "A+B\*NOT(P)+CI) A7$"

MStep6\_2 db 13, 10, "A+NOT(P)+B\*P+CI A7(+)B7$"

MStep7\_2 db 13, 10, "A+B\*P+CI A7(+)B7$"

MStep0\_1 db 13, 10, "A\*P NOT(A7\*B7)$"

MStep1\_1 db 13, 10, "A-1+CI NOT(A7)$"

MStep2\_1 db 13, 10, "NOT(A)+P+B+NOT(P)+CI) NOT(A7)$"

MStep3\_1 db 13, 10, "A-B\*P-1+CI A7NOT(+)B7$"

MStep4\_1 db 13, 10, "A+P+B\*NOR(P)+CI 1$"

MStep5\_1 db 13, 10, "A+B\*NOT(P)+CI) A7$"

MStep6\_1 db 13, 10, "A+NOT(P)+B\*P+CI A7(+)B7$"

MStep7\_1 db 13, 10, "A+B\*P+CI A7(+)B7$"

Input db 13, 10, "Enter 1-ri 2-li 3-ci 4-co 5-clk 6-chb 7-cs 8-ed 9-chs: $"

Erorr db 13, 10, "Первый бит F должен быть 0 $"

A dw ?

B dw ?

P dw ?

F dw ?

Ri db ?

LI db ?

CI dw ?

CO db ?

CLK db ?

CHB db ?

myCS db ?

ED db ?

CHS db ?

NL db 13,10,'$'

;========================= Программа =========================

.CODE

BEGIN LABEL NEAR

; инициализация сегментного регистра

mov AX, @DATA

mov DS, AX

mov ah,9

mov dx,offset InputA

int 21h

NEXTA:

mov ah, 01h

int 21h

cmp al, 0dh

je GETOUTA

sub al, 30h

cmp al, 0

je LA

add bl, 1

LA:

shl bx, 1

jmp NEXTA

GETOUTA:

shr bx, 1

mov A, bx

mov ah,9

mov dx,offset InputB

int 21h

NEXTB:

mov ah, 01h

int 21h

cmp al, 0dh

je GETOUTB

sub al, 30h

cmp al, 0

je LB

add cl, 1

LB:

shl cx, 1

jmp NEXTB

GETOUTB:

shr cx, 1

mov B ,cx

mov P ,cx

xor cx, cx

mov ah,9

mov dx,offset InputF

int 21h

;провевка на первый бит, проверить

;mov ah, 01h

;int 21h

;sub al, 30h

;bt cx, 1

;jnc error1

NEXTF:

mov ah, 01h

int 21h

cmp al, 0dh

je GETOUTF

sub al, 30h

cmp al, 0

je LF

add cl, 1

LF:

shl cx, 1

jmp NEXTF

GETOUTF:

shr cx, 1

mov F, cx

xor cx, cx

mov ah,9

mov dx,offset Input

int 21h

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov RI, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov LI, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov CI, ax

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov CO, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov CLK, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov CHB, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov myCS, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov ED, ah

mov ah, 01h

int 21h

sub al, 30h

mov ah, 0

mov CHS, ah

mov ax, 00001111b

and ax, F

cmp ax, 00001111b

jne next1

mov ax, 00001111b

or ax, F

cmp ax, 00001111b

jne step3\_2

; варианты для 1111

; 1 вариант 0000

mov cx, P

not cx

mov dx, B

or cx, dx

and cx, A

mov ah,9

mov dx,offset MStep0\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A1

mov cx, 0

jmp A1next

A1:

mov cx, 1

A1next:

bt B, 7

jc A2

mov bx, 0

jmp A2next

A2:

mov bx, 1

A2next:

and cx, bx

not cx

bt cx, 7

jc A3

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

A3:

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

step3\_2:

; варианты для 1111

; 2 вариант 0001

mov ax, F

shr ax, 4

cmp ax, 0001b

jne step3\_3

mov cx, B

not cx

or cx, P

add cx, A

add cx, CI

mov ah,9

mov dx,offset MStep1\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A1\_1

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

A1\_1:

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

step3\_3:

; варианты для 1111

; 3 вариант 0010

mov ax, F

shr ax, 4

cmp ax, 0010b

jne step3\_4

mov cx, A

not cx

and cx, P

mov bx, P

not bx

and bx, B

add cx, bx

add cx, CI

mov ah,9

mov dx,offset MStep2\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A1\_2

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

A1\_2:

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

step3\_4:

; варианты для 1111

; 4 вариант 0011

mov ax, F

shr ax, 4

cmp ax, 0011b

jne step3\_5

mov cx, A

mov bx, B

and bx, P

sub cx, bx

sub cx, 1

add cx, CI

mov ah,9

mov dx,offset MStep3\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A4\_0

mov cx, 0

jmp A0next

A4\_0:

mov cx, 1

A0next:

bt B, 7

jc A4\_1

mov bx, 0

jmp A4\_0next

A4\_1:

mov bx, 1

A4\_0next:

xor cx, bx

not cx

bt cx, 7

jc A4\_3

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

A4\_3:

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

step3\_5:

; варианты для 1111

; 5 вариант 0100

mov ax, F

shr ax, 4

cmp ax, 0100b

jne step3\_6

mov cx, A

or cx, P

mov bx, P

not bx

and bx, B

add cx, bx

add cx, CI

mov ah,9

mov dx,offset MStep4\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

step3\_6:

; варианты для 1111

; 6 вариант 0101

mov ax, F

shr ax, 4

cmp ax, 0101b

jne step3\_7

mov cx, A

mov bx, P

not bx

and bx, B

add cx, bx

add cx, CI

mov ah,9

mov dx,offset MStep5\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A3\_6

mov dl, '0'

mov ah, 02h

int 21h

jmp A3\_6next

A3\_6:

mov dl, '1'

mov ah, 02h

int 21h

A3\_6next:

jmp TOEND

step3\_7:

; варианты для 1111

; 7 вариант 0110

mov ax, F

shr ax, 4

cmp ax, 0110b

jne step3\_8

mov cx, P

not cx

or cx, A

mov bx, B

and bx, P

add cx, bx

add cx, CI

mov ah,9

mov dx,offset MStep6\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A5\_0

mov cx, 0

jmp Aanext

A5\_0:

mov cx, 1

Aanext:

bt B, 7

jc A7\_1

mov bx, 0

jmp A7\_0next

A7\_1:

mov bx, 1

A7\_0next:

xor cx, bx

call print

jmp TOEND

step3\_8:

; варианты для 1111

; 8 вариант 0111

mov ax, F

shr ax, 4

cmp ax, 0111b

jne TOEND

mov cx, B

and cx, P

add cx, A

add cx, CI

mov ah,9

mov dx,offset MStep7\_3

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A6\_0

mov cx, 0

jmp Aa6next

A6\_0:

mov cx, 1

Aa6next:

bt B, 7

jc A8\_1

mov bx, 0

jmp A8\_0next

A8\_1:

mov bx, 1

A8\_0next:

xor cx, bx

call print

jmp TOEND

next1:

mov ax, 00000001b

and ax, F

cmp ax, 00000001b

jne next1\_2

jmp todo2

next1\_2:

mov ax, 00000011b

and ax, F

cmp ax, 00000011b

jne todonext2

todo2:

; варианты для 0011

; 1 вариант 0000

mov ax, F

shr ax, 4

cmp ax, 0000b

jne next2\_1

mov cx, A

and cx, B

mov ah,9

mov dx,offset MStep0\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A10\_2

mov cx, 0

jmp A31next

A10\_2:

mov cx, 1

A31next:

bt B, 7

jc A2\_30

mov bx, 0

jmp A2\_30next

A2\_30:

mov bx, 1

A2\_30next:

and cx, bx

not cx

bt cx, 7

jc A3\_203

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

A3\_203:

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

next2\_1:

; варианты для 0011

; 2 вариант 0001

mov ax, F

shr ax, 4

cmp ax, 0001b

jne next2\_2

mov cx, A

sub cx, 1

add cx, CI

mov ah,9

mov dx,offset MStep1\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A1\_20

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

A1\_20:

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

next2\_2:

; варианты для 0011

; 3 вариант 0010

mov ax, F

shr ax, 4

cmp ax, 0010b

jne next2\_3

mov cx, A

not cx

add cx, CI

mov ah,9

mov dx,offset MStep2\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A1\_204

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

A1\_204:

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

next2\_3:

; варианты для 0011

; 4 вариант 0011

mov ax, F

shr ax, 4

cmp ax, 0011b

jne next2\_4

mov cx, A

sub cx, P

sub cx, 1

add cx, CI

mov ah,9

mov dx,offset MStep3\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A4\_20

mov cx, 0

jmp A20next

A4\_20:

mov cx, 1

A20next:

bt B, 7

jc A4\_12

mov bx, 0

jmp A4\_20next

A4\_12:

mov bx, 1

A4\_20next:

xor cx, bx

not cx

bt cx, 7

jc A4\_32

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

A4\_32:

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

next2\_4:

; варианты для 0011

; 5 вариант 0100

mov ax, F

shr ax, 4

cmp ax, 0100b

jne next2\_5

mov cx, CI

sub cx, 1

mov ah,9

mov dx,offset MStep4\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

mov dl, '1'

mov ah, 02h

int 21h

next2\_5:

; варианты для 0011

; 6 вариант 0101

mov ax, F

shr ax, 4

cmp ax, 0101b

jne next2\_6

mov cx, A

add cx, CI

mov ah,9

mov dx,offset MStep5\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A3\_62

mov dl, '0'

mov ah, 02h

int 21h

jmp A3\_62next

A3\_62:

mov dl, '1'

mov ah, 02h

int 21h

A3\_62next:

jmp TOEND

next2\_6:

; варианты для 0011

; 7 вариант 0110

mov ax, F

shr ax, 4

cmp ax, 0110b

jne next2\_7

mov cx, A

add cx, P

add cx, CI

mov ah,9

mov dx,offset MStep6\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A5\_20

mov cx, 0

jmp Aa2next

A5\_20:

mov cx, 1

Aa2next:

bt B, 7

jc A7\_12

mov bx, 0

jmp A7\_20next

A7\_12:

mov bx, 1

A7\_20next:

xor cx, bx

call print

jmp TOEND

next2\_7:

; варианты для 0011

; 8 вариант 0111

mov ax, F

shr ax, 4

cmp ax, 0111b

jne TOEND

mov cx, A

add cx, P

add cx, CI

mov ah,9

mov dx,offset MStep7\_2

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A5\_202

mov cx, 0

jmp Aa22next

A5\_202:

mov cx, 1

Aa22next:

bt B, 7

jc A7\_212

mov bx, 0

jmp A7\_205next

A7\_212:

mov bx, 1

A7\_205next:

xor cx, bx

call print

jmp TOEND

todonext2:

mov ax, F

shr ax, 4

cmp ax, 1111b

je TOEND

cmp ax, 0011b

je TOEND

cmp ax, 0001b

je TOEND

; 1 вариант 0000

cmp ax, 0000b

jne todo3\_1

mov cx, A

and cx, B

mov ah,9

mov dx,offset MStep0\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc A1q

mov cx, 0

jmp A1qnext

A1q:

mov cx, 1

A1qnext:

bt B, 7

jc A2q

mov bx, 0

jmp A2qnext

A2q:

mov bx, 1

A2qnext:

and cx, bx

not cx

bt cx, 7

jc Aq3

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

Aq3:

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

todo3\_1:

; 2 вариант 0001

mov ax, F

shr ax, 4

cmp ax, 0001b

jne todo3\_2

mov cx, A

sub cx, 1

add cx, CI

mov ah,9

mov dx,offset MStep1\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc Aq1\_1

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

Aq1\_1:

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

todo3\_2:

; 3 вариант 0010

mov ax, F

shr ax, 4

cmp ax, 0010b

jne todo3\_3

mov cx, A

not cx

add cx, CI

mov ah,9

mov dx,offset MStep2\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc Aw1\_1

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

Aw1\_1:

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

todo3\_3:

; 4 вариант 0011

mov ax, F

shr ax, 4

cmp ax, 0011b

jne todo3\_4

mov cx, A

sub cx, B

sub cx, 1

add cx, CI

mov ah,9

mov dx,offset MStep3\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc Ae4\_20

mov cx, 0

jmp Ae20next

Ae4\_20:

mov cx, 1

Ae20next:

bt B, 7

jc Ae4\_12

mov bx, 0

jmp Ae4\_20next

Ae4\_12:

mov bx, 1

Ae4\_20next:

xor cx, bx

not cx

bt cx, 7

jc Ae4\_32

mov dl, '0'

mov ah, 02h

int 21h

jmp TOEND

Ae4\_32:

mov dl, '1'

mov ah, 02h

int 21h

jmp TOEND

todo3\_4:

; 4 вариант 0100

mov ax, F

shr ax, 4

cmp ax, 0100b

jne todo3\_5

mov cx, CI

sub cx, 1

mov ah,9

mov dx,offset MStep4\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

mov dl, '1'

mov ah, 02h

int 21h

todo3\_5:

; 6 вариант 0101

mov ax, F

shr ax, 4

cmp ax, 0101b

jne todo3\_6

mov cx, A

add cx, CI

mov ah,9

mov dx,offset MStep5\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc As3\_62

mov dl, '0'

mov ah, 02h

int 21h

jmp As3\_62next

As3\_62:

mov dl, '1'

mov ah, 02h

int 21h

As3\_62next:

jmp TOEND

todo3\_6:

; 7 вариант 0110

mov ax, F

shr ax, 4

cmp ax, 0110b

jne todo3\_7

mov cx, A

add cx, P

add cx, CI

mov ah,9

mov dx,offset MStep6\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc Ad5\_20

mov cx, 0

jmp Ada2next

Ad5\_20:

mov cx, 1

Ada2next:

bt B, 7

jc Ad7\_12

mov bx, 0

jmp Ad7\_20next

Ad7\_12:

mov bx, 1

Ad7\_20next:

xor cx, bx

call print

jmp TOEND

todo3\_7:

; 8 вариант 0111

mov ax, F

shr ax, 4

cmp ax, 0111b

jne TOEND

mov cx, A

add cx, P

add cx, CI

mov ah,9

mov dx,offset MStep7\_1

int 21h

call Newline

call print

mov dl, ' '

mov ah, 02h

int 21h

bt A,7

jc Af5\_202

mov cx, 0

jmp Afa22next

Af5\_202:

mov cx, 1

Afa22next:

bt B, 7

jc Af7\_212

mov bx, 0

jmp Af7\_205next

Af7\_212:

mov bx, 1

Af7\_205next:

xor cx, bx

call print

jmp TOEND

;;;;;;;;;;; перенос троки ;;;;;;;;;;;;;;;;

Newline proc

push ax

push dx

mov ah,9

lea dx,NL

int 21h

pop dx

pop ax

ret

;;;;;;;; Вывод числа ;;;;;;;;;;;;;;;;;;;;;

print:

mov ax, cx

xor cx, cx

mov bx, 2

oi2:

xor dx,dx

div bx

push dx

inc cx

test ax, ax

jnz oi2

mov ah, 02h

oi3:

pop dx

add dl, '0'

int 21h

loop oi3

ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

error1:

mov ah,9

mov dx,offset Erorr

int 21h

TOEND:

mov ax, 4c00h

int 21h

END BEGIN

**Вывод:**

В ходе выполнения данной лабораторной работы, я изучил метод представления структурно-функциональной организации цифровых вычислительных устройств в ассемблер-программах. Приобретены умения программировать алгоритмы поведения средств вычислительной техники на уровне межрегистровых передач и уровне обработке сигналов.

**Список литературы:**

* Вострикова З. П. Программирование на языке ассемблера ЕС ЭВМ. М.: Наука, 1985.
* Калашников О. А. Ассемблер? Это просто! Учимся программировать. — [БХВ-Петербург](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%A5%D0%92-%D0%9F%D0%B5%D1%82%D0%B5%D1%80%D0%B1%D1%83%D1%80%D0%B3&action=edit&redlink=1), 2011.
* Юров В., Хорошенко С. Assembler: учебный курс. — СПб.: [Питер](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE))