МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ ФГБОУ ВО «ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ И.С.ТУРГЕНЕВА»

Кафедра «Программная инженерия»

**Отчет**

По лабораторной работе №1  
на тему:  
“Линейное исполнение программ. Арифметические и поразрядные логические операции над целыми двоичными числами  
по дисциплине: “Архитектура ЭВМ и систем”

Работу выполнил:  
Студент группы 01-ИТ

Колчев Д.Н.

Проверил:  
Доцент кафедры  
программной инженерии

Конюхова О.В.

Орёл, 2020

1. **Структура программы (-a - 5b) : (c+8)**

d\_s segment ; начало data сегмента

a db -9 ; создание ячеек памяти  
b db 1 ; передача значений  
c db -4

d\_s ends ; конец data сегмента

c\_s segment ; начало code сегмента  
assume ds:d\_s, cs:c\_s  
begin:

mov ax, d\_s ; (-a - 5b) : (c+8)  
mov ds, ax ; add, sub, mul, div

mov ch, a ; Передали в рег 'ch' значение яч. 'a'  
neg ch ; Сделали знач рег 'ch' отриц.  
mov al, 5 ; Передали рег 'al' значение 5 для умнож.  
mov bl, b ; Передали рег 'bh' значение яч 'b'  
mul bl ; 'bl\*al' = 'ax' (=5)  
sub ch, al ; 'ch-al' = 'ch' (=4)  
mov al, ch ; Передали рег 'al' значение рег 'ch' (4)  
mov dh, c ; Передали рег 'dh' значение ячейки 'c' (-4)  
add dh, 8 ; Прибавили к значению рег 'dh' значение 8 (4)  
div dh ; 'ax/dh' = 'al'(1), 'ah'(0) [ax = 0001]   
  
or al, 11000001b ; Устанавливает бит 0, 6, 7, значение '1'  
not al ; Инвертировать значения регистра.  
and al,11111011b ; Сбрасывает 2 бит, значение '0'.

mov ah, 4ch ; (9 - 5) : (-4 + 8) = 1  
int 21h ; прерывание, вызывающее завершение программы

c\_s ends ; конец code сегмента  
end begin ; конец программы

1. **Контрольные вопросы:**
2. **Понятие сегмента, характеристики сегмента, организация сегмента.**

Сегмент – это прямоугольная область памяти, характеризующаяся начальным адресом и длиной. Начальный адрес (адрес начала сегмента) – это номер (адрес) ячейки памяти, с которой начинается сегмент. Длина сегмента – это количество входящих в него ячеек памяти. Сегменты могут иметь различную длину. Все ячейки, расположенные внутри сегмента, перенумеровываются, начиная с нуля. Адресация ячеек внутри сегмента ведется относительно начала сегмента; адрес ячейки в сегменте называется смещением или эффективным адресом - EA (относительно начального адреса сегмента).

1. **На какие сегменты разбита память компьютера? В какие регистры записываются начальные адреса сегментов?** Память компьютера разбита на: сегмент данных (DS) для хранения операндов, сегмент кода (CS) для хранения операторов программы, сегмент стека (SS) – дополнительную память для временного размещения информации, а также дополнительный сегмент данных (ES).  
    Начальные адреса сегментов помещаются микропроцессором в соответствующие сегментные регистры (DS,SS,CS,E).
2. **Какие регистры процессора используются при выполнении арифметических операций?** При выполнении арифметических операций используются регистры: AX, BX, CX, DX.
3. **На какие флаги воздействуют арифметические команды?** Арифметические команды воздействуют на статусные (условные) флаги. К ним относятся:  
    CF (Carry Flag) – флаг переноса. Устанавливается в 1, если при выполнении арифметических и некоторых операций сдвига возникает «перенос» из старшего разряда.  
    PF (Parity Flag) – флаг чётности. Проверяет младшие 8 битов результатов над данными. Чѐтное число единиц приводит к установке этого флага в 1, нечѐтное – в 0.  
    AF (Auxiliary Carry Flag) – флаг логического переноса в двоично-десятичной арифметике. Устанавливается в 1, если арифметическая операция приводит к переносу или займу четвёртого справа бита однобайтового операнда.  
    Используется при арифметических операциях над двоично-десятичными кодами и кодами ASCII.  
    ZF (Zero Flag) – флаг нуля. Устанавливается в 1, если результат операции равен 0, в противном случае ZF обнуляется.  
    SF (Sign Flag) – флаг знака. Устанавливается в 1, если результат арифметической операции является отрицательным, в 0, если результат положительный.  
    OF (Overflow Flag) – флаг переполнения. Устанавливается в единицу при арифметическом переполнении, когда результат выходит за пределы разрядной сетки.
4. **Особенности выполнения команд сложения и вычитания. Требования к операндам этих команд.** Команды сложения и вычитания применяются для выполнения арифметических операций над целыми двоичными числами.  
    Особенностью данных команд является то, что они записывают результат на место первого операнда (приёмника), вследствие чего операнд –приёмник должен находиться в регистре процессора. Операнд –источник может иметь любой режим адресации.
5. **Особенности выполнения операции умножения. Особенности выполнения операции деления. Распределение регистров.** Особенностью операций умножения и деления является наличие в команде всего одного операнда, который может быть задан с помощью любого режима адресации данных, кроме непосредственного (находиться в памяти или в регистре). Второй операнд задан неявно; его местоположение фиксировано и зависит от размера операндов. Знаки результатов в операциях со знаком определяются по алгебраическим правилам.  
    Первый сомножитель указывается в самой команде, второй сомножитель должен быть помещён в регистр AL, а произведение после выполнения команды умножения помещается в регистр AX. Процесс умножения двухбайтовых сомножителей аналогичен, за исключением того, что второй сомножитель должен быть помещён в регистр AX, а произведение командой умножения сохраняется по частям в двух регистрах: DX и AX.  
    Делитель может находиться в регистре или в памяти и иметь размер 8, или 16 битов. Местонахождение делимого фиксировано:  
    1) если делитель имеет размер 1 байт, то делимое должно помещаться в регистр AX;  
    2) если делитель является двухбайтовым, то старшие два байта делимого должны размещаться в регистре DX, а младшие два байта – в регистре AX.  
    Результатом команды деления являются частное и остаток от деления, которые также размещаются в строго определённых регистрах.
6. **Основные логические операции и принципы их выполнения.** 1) Инвертировать (NOT OPR) - всех битов операнда.  
    2) Объединить по «ИЛИ» (OR DST,SRC) - Выполнение операции логического «ИЛИ» над соответствующими парами битов источника и приёмника.  
    3) Объединить по «И» (AND DST,SRC) - Выполнение операции логического «И» над соответствующими парами битов источника и приёмника.  
    4) Сложить по MOD2 («исключающее ИЛИ») (XOR DST,SRC) - Выполнение операции логического «Исключающее ИЛИ» над соответствующими парами битов источника и приёмника  
    5) Проверить (TEST OPR1,OPR2) - Выполнение операции логического сравнения над соответствующими парами битов двух операндов.
7. **Правила формирования масок для установки и сброса битов.** Логические команды наиболее часто используются для селективных (выборочных) установок, инвертирования, сброса или проверки битов в операнде – получателе в соответствии с двоичным значением операнда – источника. Такие действия часто встречаются в операциях над битами регистров и данных ввода-вывода. При этом операнд источник называют маской, а сама операция называется маскированием.  
    Команда логического сравнения TEST выполняет операцию логического умножения над операндами; при этом сами операнды не изменяются. Результат операции формируется во временной памяти: бит результата равен 1, если соответствующие биты операндов равны 1, в противном случае бит результата равен 0. Команда NOT не воздействует на флажки. Остальные команды сбрасывают OF и CF, оставляют АF не определенным и устанавливают СF, ZF, PF по обычным правилам.
8. **Каким образом выполняются логические команды над словами?**  Если размерность операндов равна слову (16 бит), то логическая операция выполняется сначала над нулевыми битами операндов и ее результат записывается на место бита 0 результата. Далее команда последовательно повторяет эти действия.