Отчет

**по лабораторной работе №6**

**«Обмен данными с ВУ по прерыванию»**

по дисциплине «Основы профессиональной деятельности» вариант 10676

Выполнил: Караганов П.Э., группа P3110

Преподаватель: Блохина Е.Н.

Оглавление

[Текст задания 3](#_Toc198687532)

[Текст исходной программы на языке Ассемблера БЭВМ 3](#_Toc198687533)

[Описание программы 5](#_Toc198687534)

[Описание основной программы 5](#_Toc198687535)

[Описание обработчиков прерывания 5](#_Toc198687536)

[Описание комплекса программ 5](#_Toc198687537)

[Описание подпрограммы 6](#_Toc198687538)

[Описание исходных данных 6](#_Toc198687539)

[Область представления 6](#_Toc198687540)

[Область допустимых значений 6](#_Toc198687541)

[Расположение в памяти БЭВМ 6](#_Toc198687542)

[Методика проверки программы 6](#_Toc198687543)

[Для проверки основной программы: 6](#_Toc198687544)

[Для проверки ВУ-1: 7](#_Toc198687545)

[Для проверки ВУ-3: 7](#_Toc198687546)

[Таблицы результатов работы комплекса 7](#_Toc198687547)

[Проверка основной программы: 7](#_Toc198687548)

[Проверка обработчика прерывания ВУ-1: 7](#_Toc198687549)

[Проверка обработчика прерывания ВУ-3: 7](#_Toc198687550)

[Вывод 8](#_Toc198687551)

# Текст задания

По выданному преподавателем варианту разработать и исследовать работу комплекса программ обмена данными в режиме прерывания программы. Основная программа должна изменять содержимое заданной ячейки памяти (Х), которое должно быть представлено как знаковое число. Область допустимых значений изменения Х должна быть ограничена заданной функцией F(X) и конструктивными особенностями регистра данных ВУ (8-ми битное знаковое представление). Программа обработки прерывания должна выводить на ВУ модифицированное значение Х в соответствии с вариантом задания, а также игнорировать все необрабатываемые прерывания.

1. Основная программа должна декрементировать содержимое X (ячейки памяти с адресом 03F16) в цикле.
2. Обработчик прерывания должен по нажатию кнопки готовности ВУ-1 осуществлять вывод результата вычисления функции F(X)=6X+1 на данное ВУ, a по нажатию кнопки готовности ВУ-3 прибавить содержимое РД данного ВУ к Х, результат записать в X
3. Если Х оказывается вне ОДЗ при выполнении любой операции по его изменению, то необходимо в Х записать максимальное по ОДЗ число.

# Текст исходной программы на языке Ассемблера БЭВМ

|  |
| --- |
| ORG 0x0  V0:         WORD $DEFAULT, 0x180  V1:         WORD $int\_1, 0x180  V2:         WORD $DEFAULT, 0x180  V3:         WORD $int\_3, 0x180  V4:         WORD $DEFAULT, 0x180  V5:         WORD $DEFAULT, 0x180  V6:         WORD $DEFAULT, 0x180  V7:         WORD $DEFAULT, 0x180  DEFAULT:    IRET                    ; Так как MR 4-ёх битыный, то можно не писать отделую логику для прерывания каждого ВУ. Для этого есть 3-ий бит MR      ORG 0x3F  X:          WORD ?  max:        WORD 0x0015             ; 21, максимальное значение X  min:        WORD 0xFFEB             ; -21, минимальное значение X  START:                              ; Инициализация векторов прерывания              DI                      ; Запрет прерывания                CLA              OUT 1                   ; MR КВУ-0 на вектор 0 и запрет прерывания              OUT 5                   ; MR КВУ-2 на вектор 0 и запрет прерывания              OUT 0xB                 ; MR КВУ-4 на вектор 0 и запрет прерывания              OUT 0xE                 ; MR КВУ-5 на вектор 0 и запрет прерывания              OUT 0x12                ; MR КВУ-6 на вектор 0 и запрет прерывания              OUT 0x16                ; MR КВУ-7 на вектор 0 и запрет прерывания              OUT 0x1A                ; MR КВУ-8 на вектор 0 и запрет прерывания              OUT 0x1E                ; MR КВУ-9 на вектор 0 и запрет прерывания              LD #9                   ; Разрешаем прерывание и связываем с вектором №1              OUT 3                   ; (1000 | 0001 = 1001 = 9) в MR КВУ-1              LD #0xB                 ; Разрешаем прерывание и связываем с вектором №3              OUT 7                   ; (1000 | 0011 = 1011 = B) в MR КВУ-3              EI  main:              DI                      ; Запрещаем прерывания для атомаронсти операции              LD X              NOP              DEC              CALL check              ST X              NOP              EI              JUMP main  int\_1:      DI                      ; Обработка прерывания из ВУ-1              LD X              NOP              ASL              ASL              ADD X              ADD X              INC              OUT 2              NOP              EI              IRET  int\_3:      DI                      ; Обработка прерывания из ВУ-3              LD X                    ; тут мы загружаем X в AC, чтобы посмотреть с каким X мы работаем (это чисто для отладки)              NOP              IN 6              SXTB              ADD X              NOP              CALL check              ST X              NOP              EI              IRET  check:                          ; Проверка на ОДЗ  check\_max:  CMP max              BMI check\_min       ; Если AC - max < 0, то прыгаем в проверку на минимум              JUMP ld\_max         ; Иначе прыгаем на загузку макс значения  check\_min:  CMP min             ;              BPL return          ; Если AC - min >= 0, то просто возращаемся  ld\_max:     LD max  return:     RET |

# Описание программы

## Описание основной программы

Основная программа инициализирует вектора прерывания, а потом в цикле уменьшает на 1 содержимое X (ячейки памяти с адресом 03F16).

## Описание обработчиков прерывания

Обработчик прерывания по нажатию кнопки готовности ВУ-1 осуществляет вывод результата вычисления функции F(X)=6X+1 на данное ВУ, a по нажатию кнопки готовности ВУ-3 складывает РД этого ВУ с X, результат записывает в Х.

## Описание комплекса программ

Комплекс программ работает со значением Х: циклически вычитается 1, в случае прерываний - прерывания от ВУ-1 и ВУ-3 обрабатываются, прерывания от остальных ВУ являются необрабатываемыми. Если Х оказывается вне ОДЗ при выполнении любой операции по его изменению, то в Х записывается максимальное по ОДЗ число (для этого вызывается подпрограмма).

## Описание подпрограммы

Подпрограмма проверяет, находится ли Х в ОДЗ. Если нет, сбрасывает Х до максимального по ОДЗ значения.

## Описание исходных данных

* 03F(X) – ссылка на результат
* 040(max), 041(min) – верхняя и нижняя границы ОДЗ для Х соответственно.

## Область представления

* X, max, min – знаковые 8-битные числа

## Область допустимых значений

* FF8016 ≤ 6x + 1 ≤ 007F16 ⟹ FFEB16 ≤ x ≤ 001516
* DR\_ВУ-3 ∈ [00; FF]

## Расположение в памяти БЭВМ

* *Векторы прерывания*: (000) – (00F)
* *Дефолтный обработчик прерывания*: (010)
* *Переменной X*: (03F)
* *Основной программы:* (051) – (058)
* *Обработчика прерывания ВУ-1:* (059) – (064)
* *Обработчика прерывания* ВУ-3: (065) – (06C)
* *Подпрограммы check:* (06D) – (073)
* *Константы (границы ОДЗ):* (040) – (041)
* *Адрес первой выполняемой команды:* (042)

# Методика проверки программы

1. Загрузить текст программы в БЭВМ.
2. Заменить NOP на HLT в той части программы, которую хотите проверить (то есть в метке main для основной программы, в метке int\_1 для теста ВУ-1 и в метке int\_3 для теста ВУ-3, можно заменить сразу везде для теста всего кода).
3. Скомпилировать программу
4. Запустить программу в режиме РАБОТА

## Для проверки основной программы:

1. Дождаться останова
2. Записать значение аккумулятора (1)
3. Нажать «Продолжение»
4. Дождаться останова и записать значение аккумулятора (2)
5. Из (1) вычесть (2) и посмотреть на разницу. Если она равна 110, то программа выполнилась корректно
6. Для дополнительной проверки крайнего случая с ОДЗ: продолжать идти по пунктам 1-5 до момента, пока число в АС не станет меньше минимального допустимого, пронаблюдать изменение аккумулятора в этот момент, записать находящееся в нем значение, сравнить с максимальным по ОДЗ числом.

## Для проверки ВУ-1:

1. Открыть КВУ-1 и установить готовность
2. Дождаться останова
3. Записать значение АС (переменную Х)
4. Нажать «Продолжение»
5. Дождаться останова
6. Записать значение АС (результат выполнения функции 6x+1)
7. Рассчитать ожидаемое значение функции для текущего Х и сверить правильность
8. В случае успеха нажать «Продолжение»

## Для проверки ВУ-3:

1. Открыть КВУ-3 и ввести туда какое-нибудь число
2. Установить готовность ВУ-3
3. Дождаться останова
4. Записать значение АС (значение DR\_ВУ-3)
5. Нажать «Продолжение»
6. Дождаться останова
7. Записать значение АС (результат выполнения функции DR\_ВУ-3 + X)
8. Рассчитать ожидаемое значение и сверить с фактическим

# Таблицы результатов работы комплекса

## **Проверка основной программы:**

|  |  |  |
| --- | --- | --- |
| Старое значение Х | Ожидаемое значение | Новое значение Х |
| 0x00 | 0xFF | 0xFF |
| 0xFB | 0xFA | 0xFA |
| 0xF1 | 0xF0 | 0xF0 |

## Проверка обработчика прерывания ВУ-1:

|  |  |  |
| --- | --- | --- |
| Значение Х | Ожидаемый результат | Фактический результат |
| 0xFD | 0xEF | 0xEF |
| 0xFF | 0xFB | 0xFB |
| 0xF1 | 0xA7 | 0xA7 |

## Проверка обработчика прерывания ВУ-3:

|  |  |  |  |
| --- | --- | --- | --- |
| Старое значение Х | DR ВУ-3 | Ожидаемое значение | Новое значение Х |
| 1111 1111 | 0000 0001 | 0000 0000 | 0000 0000 |
| 1111 1110 | 0001 0101 | 0001 0011 | 0001 0011 |
| 1111 0111 | 0101 1101 | 0101 0100 | 0101 0100 |

# Вывод

В ходе лабораторной работы мной был освоен практический навык взаимодействия с внешними устройствами через механизм прерываний, в частности отработаны методы обмена данными с ВУ-1 и ВУ-3. В процессе работы я детально изучила цикл обработки прерываний и закрепила навыки программирования на языке Ассемблера, что позволило глубже понять принципы низкоуровневого управления аппаратными средствами вычислительной системы.

***Цикл прерывания:***

1. if PS(W) = 0, то останавливаемся
2. if PS(IRQ) = 0, то просто идём дальше в цикл выборки следующей команды
3. SP – 1 -> SP, AR; IP -> DR; DR -> Mem(AR) ⇔ IP -> -(SP) (тут мы просто сохранили IP для того чтобы было куда возвращаться)
4. SP – 1 -> SP, AR; PS -> DR; DR -> Mem(AR) ⇔ PS -> -(SP) (тут мы просто сохранили PS для того чтобы при возвращении мы не сломались)
5. LTOL(CR) -> BR (сохранили младшие 8 бит в BR)
6. SHL(BR) -> BR, AR (умножаем номер вектора прерывания на 2)
7. Mem(AR) -> DR
8. DR -> IP
9. LTOL(BR+1) -> AR
10. MEM(AR) -> DR
11. DR -> PS
12. Идём в цикл выборки команды по-новому IP (мы в обработке прерывания)