МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

**ЛАБОРАТОРНАЯ РАБОТА №7**

по дисциплине

«Основы профессиональной деятельности»

Вариант № 3100

***Выполнил:***

Студент группы P3131

Дворкин Борис Александрович

***Преподаватель:***

Клименков Сергей

Викторович

Санкт-Петербург, 2023 г.

Содержание

[Текст задания 3](#_Toc127641422)

[Описание программы 3](#_Toc127641423)

[Вывод 6](#_Toc127641424)

[Таблица трассировки 7](#_Toc127641425)

# Текст задания

Синтезировать цикл исполнения для выданных преподавателем команд. Разработать тестовые программы, которые проверяют каждую из синтезированных команд. Загрузить в микропрограммную память БЭВМ циклы исполнения синтезированных команд, загрузить в основную память БЭВМ тестовые программы. Проверить и отладить разработанные тестовые программы и микропрограммы.

1. **MULSP** - Знаковое умножение младших байтов двух верхних чисел на вершине стека, результат поместить на стек, установить признаки N/Z/V/C
2. Код операции - 0F30
3. Тестовая программа должна начинаться с адреса 006416

# Исходный код синтезируемой команды

**Текст программы**

|  |  |  |  |
| --- | --- | --- | --- |
| Адрес МП | Микрокоманда | Описание | Комментарий |
| E0 | 00**88**00**9**208 | ~0 + SP ➞ SP, AR | Сохранение аккумулятора(AC) на верхушку стека. |
| E1 | 00**01**00**9**010 | AC ➞ DR |
| E2 | 0**2**00000000 | DR ➞ MEM(AR) |
| E3 | 00**80**00**9**208 | ~0 + SP ➞ AR | Сохранение командного регистра (CR) на верхушку стека. |
| E4 | 00**01**00**9**002 | CR ➞ DR |
| E5 | 0**2**00000000 | DR ➞ MEM(AR) |
| E6 | 00**80**00**9**008 | SP ➞ AR | Загрузка первого значения A со стека в DR |
| E7 | 0**1**00000000 | MEM(AR) ➞ DR |
| E8 | 00**10**00**2**001 | LTOH(DR) ➞ AC | Мл.байт числа в старший байт AC |
| E9 | 00**01**81**1**001 | extend sign DR(0..7) ➞ DR, N, Z | Расширение знака A |
| EA | 81**XX**04**1**040 | if PS(Z) = 1 then GOTO XX | Проверка: A != 0 ➞ XX (do nothing) |
| EB | 80**ED**08**1**040 | if PS(N) = 0 then GOTO ED | Проверка: *A > 0* ➞ EC (NEG) |
| EC | 00**01**00**9**501 | ~DR + 1 ➞ DR | Инвертирую знак A, если *A < 0* |
| ED | 00**02**00**2**001 | LTOH(DR) ➞ CR | Сохраняю в старший байт CR |
| EE | 00**80**00**9**408 | SP + 1 ➞ AR | Загрузка второго значения B со стека в DR |
| EF | 0**1**00000000 | MEM(AR) ➞ DR |
| F0 | 00**10**00**1**001 | LTOL(DR) ➞ AC | Мл.байт числа в мл.байт AC |
| F1 | 00**01**81**1**001 | extend sign DR(0..7) ➞ DR, N, Z | Расширение знака B |
| F2 | 81**XX**04**1**040 | if PS(Z) = 1 then GOTO XX | Проверка: B != 0 ➞ XX (do nothing) |
| F3 | 80**F5**08**1**040 | if PS(N) = 0 then GOTO F5 | Проверка: *B > 0* ➞ F4 (NEG) |
| F4 | 00**01**00**9**501 | ~DR + 1 ➞ DR | Инвертирую знак B, если *B < 0* |
| F5 | 00**02**00**1**001 | LTOL(DR) ➞ CR | Сохраняю в младший байт CR |
| F6 | 00**01**00**4**002 | HTOL(CR) ➞ DR | Умножение в цикле  while (B > 0):  A + A + ... + A ➞ **sum** |
| F7 | 00**20**00**9**021 | BR + DR ➞ BR |
| F8 | 00**01**00**8**002 | LTOL(CR) ➞ DR |
| F9 | 00**01**00**9**201 | ~0 + DR ➞ DR, N, Z |
| FA | 80**F6**04**1**040 | if PS(Z) = 0 then GOTO F6 |
| FB | 00**01**80**4**010 | HTOL(AC) ➞ DR, N, Z | Хитрая(а может и нет) проверка различных сценариев умножения.  A > 0, B < 0 ➞ NN (знак меняем)  A < 0, B > 0 ➞ NN (знак меняем)  A > 0, B > 0 ➞ XX (do nothing)  A < 0, B < 0 ➞ XX (do nothing) |
| FC | 80**PP**08**1**040 | if PS(N) = 0 then GOTO PP |
| FD | 00**01**80**1**010 | LTOL(AC) ➞ DR, N, Z |
| FE | 81**XX**08**1**040 | if PS(N) = 1 then GOTO XX |
| FF | 80**XX**08**1**040 | if PS(N) = 0 then GOTO NN |
| PP | 00**01**80**1**010 | LTOL(AC) ➞ DR, N, Z |
| ?? | 80**FF**08**1**040 | if PS(N) = 0 then GOTO XX |
| NN | 00**20**00**9**620 | ~BR + 1 ➞ BR | Меняем знак результата |
| XX | 00**88**00**9**208 | ~0 + SP ? SP, AR | Вернуть со стека старое значение командного регистра |
| ?? | 0**1**00000000 | MEM(AR) ? DR |
| ?? | 00**02**00**9**001 | DR ? CR |
| ?? | 00**88**00**9**208 | ~0 + SP ? SP, AR | Вернуть со стека старое значение аккумулятора |
| ?? | 0**1**00000000 | MEM(AR) ? DR |
| ?? | 00**10**00**9**001 | DR ? AC |
| ?? | 00**01**E0**9**020 | BR ? DR, N, Z, V, C | Положить результат операции на стек |
| ?? | 0**2**00000000 | DR ? MEM(AR) |
| ?? | 80**C4**101040 | GOTO INT @ C4 | Переход к циклу прер. |

**Текст тестовой программы**

|  |
| --- |
| ORG 0xABC  ARG1: WORD 0x0000  ARG2: WORD 0x0000  ARG3: WORD 0x00F3  ARG4: WORD 0x0042  ARG5: WORD 0x0008  ARG6: WORD 0x0013  CHECK1: WORD 0x0  CHECK2: WORD 0x0  CHECK3: WORD 0x0  FINAL: WORD 0x0  RES1: WORD 0x0  RES2: WORD 0xC15A  RES3: WORD 0x0098  START: CLA  CALL TEST1  HLT  CALL TEST2  HLT  CALL TEST3  HLT  LD #0x1  AND CHECK1  AND CHECK2  AND CHECK3  ST FINAL  STOP: HLT    TEST1: LD ARG1  PUSH  LD ARG2  PUSH  WORD 0x0F30  HLT  POP  CMP RES1  BEQ DONE1  ST RES1  POP  POP  CLA  ST CHECK1  RET  DONE1: ST RES1  POP  POP  LD #0x1  ST CHECK1  CLA  RET    TEST2: LD ARG3  PUSH  LD ARG4  PUSH  WORD 0x0F30  HLT  POP  CMP RES2  BEQ DONE2  ST RES2  POP  POP  CLA  ST CHECK2  RET  DONE2: ST RES2  POP  POP  LD #0x1  ST CHECK2  CLA  RET    TEST3: LD ARG5  PUSH  LD ARG5  PUSH  WORD 0x0F30  HLT  POP  CMP RES3  BEQ DONE3  ST RES3  POP  POP  CLA  ST CHECK3  RET  DONE3: ST RES3  POP  POP  LD #0x1  ST CHECK3  CLA  RET |

Таблица трассировки микропрограммы

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МР до выборки МК | Содержимое памяти и регистров процессора после выборки микрокоманды | | | | | | | | | |
| **MR** | **IP** | **AR** | **AR** | **DR** | **SP** | **BR** | **AC** | **NZVC** | **СчМК** |
| E0 |  |  |  |  |  |  |  |  |  | E1 |
| E1 |  |  |  |  |  |  |  |  |  | E2 |
| E2 |  |  |  |  |  |  |  |  |  | E3 |
| E3 |  |  |  |  |  |  |  |  |  | E4 |
| E4 |  |  |  |  |  |  |  |  |  | E5 |
| E5 |  |  |  |  |  |  |  |  |  | E6 |
| E6 |  |  |  |  |  |  |  |  |  | E7 |
| E7 |  |  |  |  |  |  |  |  |  | E8 |
| E8 |  |  |  |  |  |  |  |  |  | E9 |
| E9 |  |  |  |  |  |  |  |  |  | EA |
| ... |  |  |  |  |  |  |  |  |  | ... + 1 |
| end |  |  |  |  |  |  |  |  |  | C4 |

Методика проверки

1. Загрузить тестовую программу в память базовой ЭВМ.
2. Запустить основную программу с адреса 0x058 в режиме работа.
3. Дождаться останова.
4. Проверить значение ячейки памяти FINAL с номером 0x060, если значение 0x0001 – все тесты выполнены успешно.

Комментарии к методике

* Для проверки используется три пары значений: 0000 & 0000, 00F3 & 0042, 0008 & 0013
* Данные значения показывают правильную работу программы с отрицательными, нулевыми и положительными числами.
* В ходе проверки флаги N, Z, V, C меняются с 0 на 1 и с 1 на 0 в двух разных случаях, что говорит о правильном выставлении флагов.
* Результат каждого теста записывается в соответствующую ячейку RES, значение 0x0001 означает успешное выполнение. Любое другое – результат выполнения синтезированной команды
* При успешном выполнении всех тестов значение FINAL станет 0x0001, иначе 0x0000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ячейка с результатом** | | **Первое число** | **Второе число** | **Теоретический результат** | **Полученный результат** |
| RES1 | 0x401 | 0000 (N=0, Z=1) | 0000 (N=0, Z=1) | 0000 (N=0, Z=1) | ? (N=0, Z=1) |
| RES2 | 0x402 | 00F3 (N=1, Z=0) | 0042 (N=0, Z=0) | C15A (N=1, Z=0) | ? (N=1, Z=0) |
| RES3 | 0x403 | 0008 (N=0, Z=0) | 0013 (N=0, Z=0) | 0098 (N=0, Z=0) | ? (N=0, Z=0) |

# Вывод

В ходе выполнения лабораторной работы я изучил алгоритм синтеза собственной команды БЭВМ с помощью горизонтальных микрокоманд, разработал методику проверки сделанной программы, а также получил полное понимание устройства БЭВМ и желание разработать новые команды и модули для уже существующей БЭВМ, либо создать свою с нуля.

# Методика проверки программы

Проверка обработки прерываний:

1. Загрузить текст программы в БЭВМ.

2. Заменить NOP по нужному адресу на HLT.

3. Запустить программу в режиме РАБОТА.

4. Установить «Готовность ВУ-1».

5. Дождаться останова.

6. Записать текущее значение X из памяти БЭВМ:

1. Запомнить текущее состояние счетчика команд.

2. Ввести в клавишный регистр значение 0x010

3. Нажать «Ввод адреса».

4. Нажать «Чтение».

5. Записать значение регистра данных.

6. Вернуть счетчик команд в исходное состояние.

7. Нажать «Продолжение» 2 раза.

8. Записать результат обработки прерывания – полученное значение F(x) из DR контроллера ВУ-1.

9. Нажать «Продолжение».

10. Ввести в ВУ-3 произвольное число, записать его.

11. Установить «Готовность ВУ-3».

12. Дождаться останова.

13. Записать текущее значение X из памяти БЭВМ:

1. Запомнить текущее состояние счетчика команд.

2. Ввести в клавишный регистр значение 0x010

3. Нажать «Ввод адреса».

4. Нажать «Чтение».

5. Записать значение регистра данных.

6. Вернуть счетчик команд в исходное состояние.

14. Нажать «Продолжение» 2 раза.

15. Записать результат обработки прерывания – DR + X из DR контроллера ВУ-3.

16. Рассчитать ожидаемое значение переменной X после обработки прерывания и сравнить.

Проверка основной программы:

1. Загрузить текст программы в БЭВМ.

2. Записать в переменную X максимальное по ОДЗ значение (22)

3. Запустить программу в режиме останова.

4. Пройти нужное количество шагов программы, убедиться, что при уменьшении X на 1, до после момента, когда он равен 2-21, происходит сброс значения в минимальное по ОДЗ.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Прерывание ВУ-1 | | | Прерывание ВУ-3 | | | | |
| AC  (0...7) | Ожидание  -6\*X+3 | DR | AC (0…7) | DR  КВУ-3 | AC  (DR + X) | Результат AC (0...7) |
| 1016 (16) | FFA316 (-93) | FFA316 (-93) | 116 (1) | (127) | 5D16 (126) | 1616 (22) |
| FF16 (-1) | FD16 (-3) | FD16 (-3) | 116 (1) |  | 0016 (0) | 016 (0) |
| 1816 (25) | 7F16 (127) | 7F16 (127) | 116 (1) | (-31) | E216 (-32) | EВ16 (-21) |