Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

**Лабораторная работа № 7**

“Синтез команд БЭВМ”

Вариант № 6

Выполнил:

Сандов Кирилл Алекссевич

Группа:

P3113

Проверила:

преподаватель Блохина Елена Николаевна

Санкт-Петербург

2023

# Задание

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.

* MSUB M - вычитание аккумулятора из ячейки с записью результата в ячейку памяти без установки N/Z/V/C
* Код операции - 9...
* Тестовая программа должна начинаться с адреса 034516

# Пункт 1

9XXX – прибавление к значению ячейки памяти в адресной части команды значения аккумулятора в дополнительном коде (~AC + 1), не устанавливая флаги результата. Дополнительного декодирования не требует. Далее представлен микрокод данной команды (Таблица 1).

|  |  |  |
| --- | --- | --- |
| **Адрес МП** | **Микрокоманда** | **Действие** |
| E0 | 0201009611 | ~AC + DR + 1 → DR; DR → MEM(AR) |
| E1 | 80C4101040 | GOTO INT @ C4 |

Таблица 1

# Пункт 2

Далее приведён текст тестовой программы (Листинг 1).

|  |
| --- |
| ORG 0x0E0  LD #0x01 ; Тест: 1 - 0 = 1, прямая абсолютная адресация  PUSH  LD #0x00  PUSH  LD $T1\_OP  PUSH  LD #0x01  PUSH  CALL TEST  POP  POP  POP  ST $T1\_RES  POP  ST $T1\_CHK  NOP ; Breakpoint  LD #0x01 ; Тест: 1 - 1 = 0, косвенная относительная адресация  PUSH  LD #0x01  PUSH  LD $T2\_OP  PUSH  LD #0x00  PUSH  CALL TEST  POP  POP  POP  ST $T2\_RES  POP  ST $T2\_CHK  NOP ; Breakpoint  LD #0x00 ; Тест: 0 - 1 = FFFF, косвенная автоинкрементная адресация  PUSH  LD #0x01  PUSH  LD $T3\_OP  PUSH  LD #0xFF  PUSH  CALL TEST  POP  POP  POP  ST $T3\_RES  POP  ST $T3\_CHK  NOP ; Breakpoint  LD #0x02 ; Тест: 2 - 1 = 1, косвенная автодекрементная адресация  PUSH  LD #0x01  PUSH  LD $T4\_OP  PUSH  LD #0x01  PUSH  CALL TEST  POP  POP  POP  ST $T4\_RES  POP  ST $T4\_CHK  NOP ; Breakpoint  LD #0x80 ; Тест: FF80 - 1A = FF66, прямая относительная адресация  PUSH  LD #0x1A  PUSH  LD $T5\_OP  PUSH  LD $T5\_ANS  PUSH  CALL TEST  POP  POP  POP  ST $T5\_RES  POP  ST $T5\_CHK  T5\_ANS: WORD 0xFF66 ; Вспомогательная ячейка для загрузки результата теста 5  NOP ; Breakpoint  LD #0x05 ; Тест: 5 - 1F = FFE6, косвенная относительная, со смещением (SP) адресация  PUSH  LD $T6\_OP  ST SPOPER  LD #0x1F  NOP  SPOPER: NOP  NOP  POP  ST $T6\_RES  CMP #0xE6  BEQ SPPASS  LD #0x00  JUMP SPRES  SPPASS: LD #0x01  SPRES: ST $T6\_CHK  NOP ; Breakpoint  LD $T7\_OP ; Тест: 5 - 3 = 2, прямая загрузка операнда  ST FOPER  LD #0x03  NOP  FOPER: NOP  NOP  LD FOPER  ST $T7\_RES  CMP #0x02  BEQ FPASS  LD #0x00  JUMP FRES  FPASS: LD #0x01  FRES: ST $T7\_CHK  NOP ; Breakpoint  LD #0x01 ; Проверка выполнения всех тестов  AND $T1\_CHK  AND $T2\_CHK  AND $T3\_CHK  AND $T4\_CHK  AND $T5\_CHK  AND $T6\_CHK  AND $T7\_CHK  ST ALL\_CHK  HLT  ORG 0x160 ; Универсальная подпрограмма для тестирования операции  ; Принимает 4 аргумента:  ; 1) Значение ячейки памяти - уменьшаемое  ; 2) Значение аккумулятора - вычитаемое  ; 3) Код операции для тестирования  ; 4) Результат для проверки  ; Возвращает 2 значения:  ; 1) 1, если тест пройден 0, если тест провален  ; 2) Вычисленное тестируемой операцией значение  TEST: LD &0x04 ; Загрузить уменьшаемое  ST X ; Сохранить уменьшаемое  LD &0x02 ; Загурзить КОП  ST OPER ; Сохранить КОП  LD &0x03 ; Загрузить вычитаемое  NOP ; Breakpoint  OPER: NOP ; Исполнить КОП, Address = 166  NOP ; Breakpoint  LD X ; Загрузить результат  ST &0x03 ; Сохранить вычисленное значение  LD &0x01 ; Загрузить результат для проверки  CMP X ; Сравнить с получившимся результатом  BEQ PASSED ; В случае равенства - тест пройден  LD #0x00 ; Загружаем отметку о провале теста  JUMP ST\_RES ; Переходим к сохранению  PASSED: LD #0x01 ; Иначе загружаем отметку о прохождении теста  ST\_RES: ST &0x04 ; Сохранить возвращаемое значение  RET  WORD $X ; Вспомогательная ячейка для косвенной относительной и автоинкрементной адресации, Address = 172  WORD $T1\_RES ; Вспомогательная ячейка для косвенной автодекрементной адресации, Address = 173  ORG 0x180  X: WORD ?  T1\_RES: WORD ?  T2\_RES: WORD ?  T3\_RES: WORD ?  T4\_RES: WORD ?  T5\_RES: WORD ?  T6\_RES: WORD ?  T7\_RES: WORD ?  ORG 0x191  T1\_CHK: WORD ?  T2\_CHK: WORD ?  T3\_CHK: WORD ?  T4\_CHK: WORD ?  T5\_CHK: WORD ?  T6\_CHK: WORD ?  T7\_CHK: WORD ?  NOP  ALL\_CHK:WORD ?  ORG 0x1A0  T1\_OP: WORD 0x9190  T2\_OP: WORD 0x980B  T3\_OP: WORD 0x9A0B  T4\_OP: WORD 0x9B0C  T5\_OP: WORD 0x9E19  T6\_OP: WORD 0x9C00  T7\_OP: WORD 0x9F05 |

Листинг 1

# Пункт 3

Запишем таблицу трассировки циклов исполнения разработанных микрокоманд для одного варианта выполнения каждой микрокоманды (Таблица 2).

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MP до выборки МК** | **Содержимое памяти и регистров процессора после выборки и исполнения микрокоманды** | | | | | | | | | |
| **MR** | **IP** | **CR** | **AR** | **DR** | **SP** | **BR** | **AC** | **NZVC** | **MP (СчМК)** |
| 28 | 813C804002 | 167 | 9190 | 190 | 0000 | 7FB | 0166 | 0000 | 0100 | 3C |
| 3C | 8143204002 | 167 | 9190 | 190 | 0000 | 7FB | 0166 | 0000 | 0100 | 3D |
| 3D | 81E0104002 | 167 | 9190 | 190 | 0000 | 7FB | 0166 | 0000 | 0100 | E0 |
| E0 | 0201009611 | 167 | 9190 | 190 | 0000 | 7FB | 0166 | 0000 | 0100 | E1 |
| E1 | 80C4101040 | 167 | 9190 | 190 | 0000 | 7FB | 0166 | 0000 | 0100 | C4 |

Таблица 2

# Пункт 3

Методика проверки команды с использованием тестовой программы:

1. Загрузить микрокоманду «0201009611» в память микрокоманд по адресу E0;
2. Загрузить микрокоманду «80C4101040» в память микрокоманд по адресу E1;
3. Загрузить тестовую программу в БЭВМ;
4. Изменить режим «Останов» на «Работа» и запустить программу с ячейки 0E0;
5. Дождаться останова;
6. В аккумуляторе будет уменьшаемое;
7. Продолжить выполнение программы;
8. Дождаться останова;
9. В аккумуляторе будет вычитаемое;
10. Продолжить выполнение программы;
11. Дождаться останова;
12. Аккумулятор должен не измениться, как и флаги NZVC, что будет свидетельствовать о правильной работе реализованной команды;
13. Продолжить выполнение программы;
14. В аккумуляторе будет загруженное из ячейки памяти значение разности;
15. Продолжить выполнение программы;
16. Дождаться останова;
17. В аккумуляторе будет 1, если тест пройден успешно (получено верное значение разности), либо 0, если он провален;
18. Продолжить выполнение программы;
19. Повторить шаги с 5 по 18 ещё 6 раз, так как в тестовой программе предусмотрено 7 различных тестов;
20. После выполнения шагов для последнего теста и останова в аккумуляторе будет 1, если все тесты были пройдены, либо 0, если хотя бы один из них провален;
21. Все результаты тестов были сохранены в ячейки памяти 191-197. Финальный результат всего тестирования сохранён в ячейку 199;
22. Все вычисленные значения разности сохранены в ячейки памяти 181-187.

# Заключение

В результате выполнения данной лабораторной работы было изучено микропрограммное устройство БЭВМ. Во-первых, был рассмотрен внутреннее устройство устройства управления. Во-вторых, получены знания о различных типах микрокоманд: операционных и управляющих. В итоге, была реализована собственная команда для БЭВМ, для которой был написан свой микрокод.