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**

по дисциплине

«Основы профессиональной деятельности»

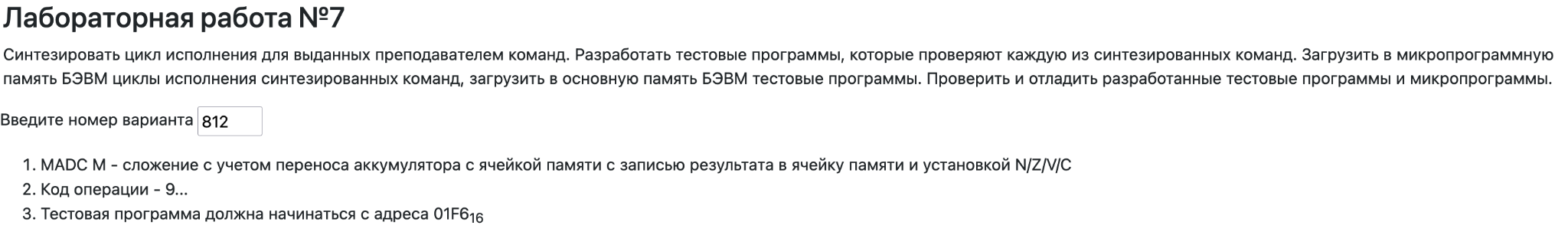
Синтез команд БЭВМ

Вариант № 812

***Выполнил:***Студент группы P3107  
 Чусовлянов Максим Сергеевич

***Принял:***Вербовой Александр Александров

# **Текст задания**



# **Исходный код синтезируемой команды**

| **Адрес ячейки** | **новый код МК** | **Комментарий** |
| --- | --- | --- |
| 3D | 81E0104002 | if CR(12) = 1 then GOTO RESERVED E0  ; Команда 9… обрабатывается микрокомандой с адресом E0 |
| **Цикл исполнения команды MADC (E0—E4)** | | |
| E0 | 80E3011040 | if PS(C) = 0 then GOTO E3  ; Флаг С не выставлен => перейти на E3 (из ADC) |
| E1 | 0001E09411 | AC + DR + 1 -> AC, N, Z, V, C  ; Суммирование AC, DR, C в аккумулятор с установкой флагов (из ADC) |
| E2 | 80E4101040 | GOTO E4  ; перейти на E4 (из ADC) |
| E3 | 0001E09011 | AC + DR -> AC, N, Z, V, C  ; Суммирование AC, DR в аккумулятор с установкой флагов (из ADD) |
| E4 | 0200000000 | DR -> MEM(AR)  ; Запись результата в ячейку памяти |
| E5 | 80C4101040 | GOTO INT @ C4  ; Завершение цикла выполнения команды, переход к циклу прерываний |

# **Трассировка микропрограммы**

| MP до выборки МК | Содержимое памяти и регистров процессора после выборки и исполнения команды | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MR | IP | CR | AR | DR | SP | BR | AC | NZVC | MР |
| E0 | 80E3011040 | 1FD | 91E1 | 1E1 | 0007 | 000 | 01FC | 1234 | 0000 | E3 |
| E3 | 0001E09011 | 1FD | 91E1 | 1E1 | 123B | 000 | 01FC | 1234 | 0000 | E4 |
| E4 | 0200000000 | 1FD | 91E1 | 1E1 | 123B | 000 | 01FC | 1234 | 0000 | E5 |
| E5 | 80C4101040 | 1FD | 91E1 | 1E1 | 123B | 000 | 01FC | 1234 | 0000 | C4 |

# **Код программы проверки команды на языке ассемблера**

ORG 0x1E0

test1\_n1: WORD 0x1234

test1\_n2: WORD 0x0007

test1\_ans: WORD 0x0000

test2\_n1: WORD 0x1717

test2\_n2: WORD 0x1000

test2\_ans: WORD 0x0000

test3\_n1: WORD 0x7FFF

test3\_n2: WORD 0x1234

test3\_ans: WORD 0x0000

test3\_of1: WORD 0x0000

test3\_of2: WORD 0x0000

test1\_res: WORD ?

test2\_res: WORD ?

test3\_res: WORD ?

ORG 0x1F6

start: CLA

LD $test1\_n1

ADD $test1\_n2

*ST* $test1\_ans

LD $test1\_n1

WORD 0x91E1; MADC $test1\_n2

LD $test1\_ans

CMP $test1\_n2

BEQ test1\_p

JUMP test1\_f

test1\_p: LD #0x1

*ST* $test1\_res

JUMP test\_2

test1\_f: CLA

*ST* $test1\_res

JUMP test\_2

test\_2: LD $test2\_n1

ADD $test2\_n2

INC

*ST* $test2\_ans

CLC

CMC

LD $test2\_n1

WORD 0x91E4; MADC $test2\_n2

LD $test2\_ans

CMP $test2\_n2

BEQ test2\_p

JUMP test2\_f

test2\_f: CLA

*ST* $test2\_res

JUMP test\_3

test2\_p: LD #0x1

*ST* $test2\_res

JUMP test\_3

test\_3: LD $test3\_n1

ADD $test3\_n2

INC

*ST* $test3\_ans

BVC SKIP1

LD $test3\_of1

INC

*ST* $test3\_of1

SKIP1: NOP

CLC

CMC

LD $test3\_n1

WORD 0x91E7; MADC $test3\_n2

BVC SKIP2

LD $test3\_of2

INC

*ST* $test3\_of2

SKIP2: NOP

LD $test3\_of1

CMP $test3\_of2

BNE test3\_p

JUMP test3\_f

test3\_f: CLA

*ST* $test3\_res

JUMP main

test3\_p: LD #0x1

*ST* $test3\_res

JUMP main

main: LD $test1\_res

AND $test2\_res

AND $test3\_res

CMP #0x1

BEQ success

LD #0xFF

HLT

success: LD #0x1

HLT

# **Описание тестовых программ**

1. Первый тест проверяет команду MADC без CF: проверка на эквивалентность результату команды ADD тех же чисел. Результат теста в RES1 (0 - ошибка, 1 - тест пройден)

2. Второй тест проверяет результаты после ADD двух чисел + 1 с результатами MADC (с выставленным CF), они должны совпадать.

3. Третий тест рассматривает случай, когда AC = 0x7FFF, DR = 0x1234, C = 1. Проверка что V = 1 при MADC и V=0 при (AC + DR) + 1.

# **Подготовка к проверке**

2. Перенести микрокоманды в БЭВМ

3. Загрузить команды Assembler в БЭВМ

4. Заменить везде NOP на HLT.

5. ПРОВЕРИТ

# **Методика проверки программы**

1. Открыть БЭВМ в формате dual “java -jar –Dmode=dual bcomp-ng.jar”

2. Ввести микрокоманды через консоль

ma

mw 80E3011040

mw 0001E09411

mw 80E4101040

mw 0001E09011

mw 0200000000

mw 80C4101040

3. Скомпилировать и запустить код на ассемблере

4. Удостоверится что после прогона всех тестов в аккумуляторе лежит 0x1 (0xff - ошибка).

5. Для запуска отдельного теста: поставить HLT в нужном месте и вбить адрес метки теста в IR.

# **Вывод**

После проделанной лабораторной работы, я узнал как синтезировать свою команду БЭВМ при помощи микропрограмм. Научился тестировать разработанные микропрограммы на крайние случаи.