Министерство высшего образования и науки Российской Федерации

Национальный научно-исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №3  
по дисциплине  
**«Основы профессиональной деятельности».**

Вариант №666.

Работу выполнил:

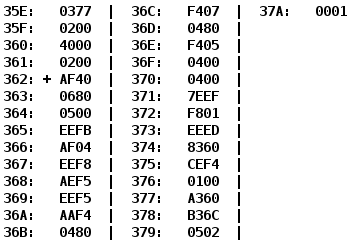
Афанасьев Кирилл Александрович,  
Студент группы P3106.  
Преподаватель:  
Афанасьев Дмитрий Борисович.

Санкт-Петербург, 2023

Задание:

«По выданному преподавателем варианту восстановить текст заданного варианта программы, определить предназначение и составить описание программы, определить область представления и область допустимых значений исходных данных и результата, выполнить трассировку программы.

Вариант 846:

»

Текст исходной программы:

*Таблица: Текст исходной программы*

|  |  |  |  |
| --- | --- | --- | --- |
| Адрес | Код команды | Мнемоника | Комментарий |
| 35E | 0377 | X: WORD 0377 | Ячейка с адресом первого элемента массива |
| 35F | 0200 | Y: WORD 0200 | Ячейка хранения текущего элемента массива |
| 360 | 4000 | L: WORD 4000 | Длина массива (счетчик) |
| 361 | 0200 | R: WORD 0200 | Результат работы программы и промежуточных вычислений |
| 362 | AF40 | LD #40 | Прямая загрузка в AC значения 40(16) |
| 363 | 0680 | SWAB | Обмен байтами в AC. Так как в AC прямо загружено 40(16), результат всегда будет равен 4000 |
| 364 | 0500 | ASL | Сдвиг AC влево. Так как нам известен результат предыдущей операции, результат этой всегда будет равен 8000 (умножение на 2) |
| 365 | EEFB | ST IP-5 | Прямая относительная адресация: ST 366 + FFB = (1)361 (C=1;V=0). Сохраняет содержимое аккумулятора в ячейку результата |
| 366 | AF04 | LD #04 | Прямая загрузка в AC значения 4(16) -- длины массива |
| 367 | EEF8 | ST IP-8 | Прямая относительная адресация: ST 368 + FF8 = (1)360 (C=1;V=0). Сохраняет содержимое аккумулятора в память |
| 368 | AEF5 | LD IP-B | Прямая относительная адресация: LD 369 + FF5 = (1)35E (C=1;V=0). Загружает из ячейки памяти значение в аккумулятор |
| 369 | EEF5 | ST IP-B | Прямая относительная адресация: ST 36A + FF5 = (1)35F (C=1;V=0). Сохраняет содержимое аккумулятора в ячейку памяти. |
| 36A | AAF4 | LD (IP-C)+ | Косвенная относительная автоинкрементная адресация: Значение адреса берется из ячейки 35F, далее мы загружаем по полученному адресу значение для команды, а значение адреса увеличиваем на 1. Полученное значение загружается в аккумулятор |
| 36B | 0480 | ROR | Значение AC сдвигается вместе с Carry флагом вправо. |
| 36C | F407 | BCS IP+7 | Если Carry флаг установился (то бишь 0й бит был 1) -- перемещаемся на ячейку IP + 7 + 1. Иначе продолжаем выполнение |
| 36D | 0480 | ROR | Если мы продолжили выполнение, то значение AC снова сдвигается вместе с Carry флагом вправо. |
| 36E | F405 | BCS IP+5 | Если Carry флаг установился (то бишь 1й бит был 1) -- перемещаемся на ячейку IP + 5 + 1. Иначе продолжаем выполнение |
| 36F | 0400 | ROL | Если мы продолжили выполнение, сдвигаем AC и C влево |
| 370 | 0400 | ROL | Сдвигаем AC и C влево. В AC теперь исходное значение из массива |
| 371 | 7EEF | CMP IP-11 | Прямая относительная адресация: CMP 372 + FEF = (1)361 (C=1; V=0). Производит вычитание значения ячейки из аккумулятора, результат операции -- установка флагов, соответствующие флагам результата операции |
| 372 | F801 | BLT IP + 1 | Если не произошло переполнение, но 15й разряд обратился в 1, или наоборот, это означает, что значение AC меньше значения из ячейки, мы перейдем на ячейку IP + 1 + 1. Иначе продолжим выполнение |
| 373 | EEED | ST IP-13 | Прямая относительная адресация: ST 374 + FED = (1)361. Если мы продолжили выполнение, обновляем значение ячейки по относительному адресу |
| 374 | 8360 | LOOP 360 | По абсолютному адресу 0x360 уменьшаем счетчик на 1. Если он оказался <= 0 -> перейдем на IP + 1 + 1, иначе продолжим выполнение |
| 375 | CEF4 | JUMP 36A | Прямая относительная адресация: JUMP 376 + FF4 = (1)36A. Если цикл не закончился, мы безусловно перейдем в ячейку IP-C |
| 376 | 0100 | HLT | Если оказалось, что цикл завершил работу, мы останавливаем работу ЭВМ и возвращаем управление оператору |
| 377 | A360 | WORD A360 | Элемент массива |
| 378 | B36C | WORD B36C | Элемент массива |
| 379 | 0502 | WORD 0502 | Элемент массива |
| 37A | 0001 | WORD 0001 | Элемент массива |

*Окончание таблицы*

Описание программы:

* Реализуемая функция: R = Y & (Z + X)
* Назначение программы: складывает 2 числа и выполняет операцию поразрядного умножения с третьим числом.
* Исходные данные должны располагаться в ячейках памяти: Y – 19E, Z – 19F, X – 19D. ОПИ и ОДЗ расписано в следующем пункте.
  + Исходные данные нужны для подсчета значения результата R в реализуемой программе по формуле.
  + Для хранения промежуточного результата используется ячейка памяти R’ – 1A1
* Результат работы программы будет находиться в ячейке 1A0.
* Вся программа располагается в памяти в ячейках между адресами 195 и 1A1 включительно. Первая команда располагается по адресу 195. Последняя – 19C.

Область представления и допустимых значений.

R = Y & (Z + X)

ОПИ:

* R – набор из 16 логических однобитовых значений.
* Z, X – знаковые 16-рязрядные числа.
* Y – набор из 16 логических однобитовых значений.

Результат арифметической операции (Z + X) трактуется как логическое значение

* (Z + X) – набор из 16 логических однобитовых значений.

ОДЗ:  
R:

X, Y, Z:

Таблица трассировки выполнения команд:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполняемая команда | | Содержимое регистров процессора после выполнения команды | | | | | | | | Ячейка, содержимое которой изменилось | |
| Адрес | Код | IP | CR | AR | DR | SP | BR | AC | NZVC | Адрес | Код |
| 195 | A19F | 196 | A19F | 19F | E1A0 | 0000 | 0195 | E1A0 | 1000 |  |  |
| 196 | 419D | 197 | 419D | 19D | A19F | 0000 | 0196 | 833F | 1001 |  |  |
| 197 | E1A1 | 198 | E1A1 | 1A1 | A180 | 0000 | 0197 | 833F | 1001 | 1A1 | 833F |
| 198 | 0200 | 199 | 0200 | 198 | 0200 | 0000 | 0198 | 0000 | 0101 |  |  |
| 199 | 319E | 19A | 319E | 19E | 319E | 0000 | CEA1 | 319E | 0001 |  |  |
| 19A | 21A1 | 19B | 21A1 | 1A1 | A180 | 0000 | 0198 | 2180 | 0001 |  |  |
| 19B | E1A0 | 19C | E1A0 | 1A0 | 011E | 0000 | 019B | 011E | 0001 | 1A0 | 011E |
| 19C | 0100 | 19D | 0100 | 19C | 0100 | 0000 | 019C | 011E | 0001 |  |  |

Вариант с меньшим числом команд:

Вместо того, чтобы очищать аккумулятор и выполнять логическое «ИЛИ», мы можем сразу применить операцию загрузки.

Таким образом, команды 0200 и 319E можно заменить на A19E. Чтобы убрать образовавшиеся пропуски в памяти, часть данных переместить на 1 ячейку вверх в памяти. Таким образом, получаем следующую программу:

*Таблица: Текст программы с меньшим числом команд*

|  |  |  |  |
| --- | --- | --- | --- |
| Адрес | Код команды | Мнемоника | Комментарий |
| 195 | A19E | LD 19E | Загрузка в AC значения ячейки 19E. Является точкой входа в программу. |
| 196 | 419C | ADD 19C | Добавить содержимое ячейки 19C к AC |
| 197 | E1A0 | ST 1A0 | Сохраняет содержимое AC в ячейку 1A0 |
| 198 | A19D | LD 19D | Загрузка в AC значения ячейки 19D. |
| 199 | 21A0 | AND 1A0 | Над каждым разрядом регистра AC выполняется логическое «И» с соответствующим разрядом ячейки 1A0. |
| 19A | E19F | ST 19F | Сохраняет содержимое AC в ячейку 19F |
| 19B | 0100 | HLT | Остановка работы программы. |
| 19C | A19F | X | Переменная X |
| 19D | 319E | Y | Переменная Y |
| 19E | E1A0 | Z | Переменная Z |
| 19F | 21A1 | R | Результат работы программы будет храниться здесь. |
| 1A0 | 0200 | R’ | Ячейка промежуточного результата. |

*Окончание таблицы*

Вывод

Во время выполнения данной лабораторной работы я ознакомился с базовой структурой ЭВМ, научился определять элементарную функцию, вычисляемую программой, область допустимых значений и представления переменных.