Министр науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет информационных технологий и программирования

Лабораторная работа № 2

Исследование работы БЭВМ при выполнении разветвляющихся программ

Выполнил студент группы № M3101
Михеев Артем Романович
Подпись:

Проверил:
Бабич Мария Сергеевна

Санкт-Петербург
2020

Цель работы

Изучение команд переходов, способов организации разветвляющихся программ и исследование порядка функционирования БЭВМ при выполнении таких программ.

Задание, вариант 6

До начала исполнения программы составить теоретическую таблицу трассировки, после чего занести в память БЭВМ заданный вариант программы, и, выполняя её по командам, заполнить таблицу трассировки программы, а также проанализировать сами команды программы для определения назначения программы, реализуемые ею функции и т.д.

Вариант 6:
(первая команда помечена плюсиком)

Адрес	Коды команд
016	0000
017	+C01B
018	001B
019	FF20
01A	00DF
01B	F200
01C	4019
01D	401A
01E	A021
01F	F200
020	F100
021	3023
022	F000
023	C008

Решение

1. Для начала, используя таблицу команд и мнемоник БЭВМ, приведём текст программы вместе с мнемониками и комментариями.

Адрес	Код команды	Мнемоника	Комментарий
017	C01B	BR 1B	1B➔СК ; безусловный переход, значение счетчика команд изменятся на 01B, и следующим шагом исполнение начнется оттуда
018	001B	ISZ 1B	(M)++, if (M) >= 0 then СК+1➔СК ; операция приращения и пропуска, увеличит число в ячейке 01B на 1, и если оно неотрицательно, пропустит одну команду.

019	FF20	-	Не команда, это данные, которые используются программой
01A	00DF	-	Также данные.
01B	F200	CLA	0➔A
01C	4019	ADD 19	(19)+A➔A ; прибавит значение ячейки 019 к аккумулятору и сохранит результат в аккумулятор
01D	401A	ADD 1A	(1A)+A➔A ; прибавит значение ячейки 01A к аккумулятору и сохранит результат в аккумулятор
01E	A021	BMI 21	If A < 0 then 21➔СК ; сравнит значение аккумулятора с нулем, если больше или равно – ничего не произойдет, иначе в счетчик команд запишется 021, исполнение продолжится оттуда
01F	F200	CLA	0➔A
020	F100	NOP	Операция, которая ничего не делает.
021	3023	MOV 23	A➔(23) ; запишет значение аккумулятора в ячейку 023
022	F000	HLT	

Остальные ячейки (016, 023) либо не использовались, либо содержали не команды, а данные, с которыми работала программа.

Соответственно для удобства данную программу можно написать на ассемблере как:

```
ORG 0016
WORD 0000

BEGIN:
        BR 1B
        ISZ 1B
        WORD FF20
        WORD 00DF
        CLA
        ADD 19
        ADD 1A
        BMI 21
        CLA
        NOP
```

```
        MOV RES
        HLT

ORG 0023
RES: WORD C008
```

2. Составим теоретическую таблицу трассировки без исполнения на БЭВМ:

Выполняемая команда		Содержимое регистров процессора после выполнения команды.						Ячейка, содержим. которой изменилось после вып. Программы	
Адрес	Код	СК	РА	РК	РД	А	С	Адрес	Новый код
017	C01B	01B	017	C01B	C01B	0000	0	-	-
01B	F200	01C	01B	F200	F200	0000	0	-	-
01C	4019	01D	019	4019	FF20	FF20	0	-	-
01D	401A	01E	01A	401A	00DF	FFFF	0	-	-
01E	A021	021	01E	A021	A021	FFFF	0	-	-
021	3023	022	023	3023	FFFF	FFFF	0	023	FFFF
022	F000	023	022	F000	F000	FFFF	0	-	-

3. Теперь же составим экспериментальную таблицу трассировки

Выполняемая команда		Содержимое регистров процессора после выполнения команды.						Ячейка, содержим. которой изменилось после вып. Программы	
Адрес	Код	СК	РА	РК	РД	А	С	Адрес	Новый код
017	C01B	01B	017	C01B	C01B	0000	0	-	-
01B	F200	01C	01B	F200	F200	0000	0	-	-
01C	4019	01D	019	4019	FF20	FF20	0	-	-
01D	401A	01E	01A	401A	00DF	FFFF	0	-	-
01E	A021	021	01E	A021	A021	FFFF	0	-	-
021	3023	022	023	3023	FFFF	FFFF	0	023	FFFF
022	F000	023	022	F000	F000	FFFF	0	-	-

Вышло так, что экспериментальная таблица трассировки совпала с теоретической, и это можно обосновать тем, что теоретическая таблица составлялась с существующими знаниями о процессе исполнения и цикле команд.

4. Теперь проанализируем программ:
   Назначение программы и функционал – сложение двух чисел и определение того, является ли результат отрицательным. Программа складывает два числа, записанных в памяти, после чего при отрицательном результате сохраняет его в ячейку 023, иначе в ячейку 023 записывает 0.
   Реализуемые формулы – if A+B < 0 then (023) = A+B else (023) = 0
   Программа располагается в памяти ЭВМ в ячейках 017, 01A..023, адрес первой исполняемой команды – 017, а последней, соответственно, - 022
   Исходные данные программы (переменные A, B) хранятся по адресам 019, 01A.
   Результат исполнения программы записывается по адресу 023.

5. Правильное изменение программы с сохранением формулы для более краткого кода

```
ORG 0017
BEGIN:
        CLA
        ADD NA
        ADD NB
        BMI NEGATIVE_CASE
        CLA
NEGATIVE_CASE:
        MOV RES
        HLT

ORG 0021
NA: WORD FF20
NB: WORD 00DF
RES: WORD 0000
```

   В оригинальной программе в самом начале в СК записывался другой адрес, что можно убрать, просто перенеся основное тело программы в начало, а также была операция NOP, которая ничего не делала (в наборах инструкций настоящих процессоров она также присутствует, но там часто используется в логичных целях, или как минимум для выравнивания памяти).

Выводы

В ходе лабораторной работы пришлось более подробно изучить сам процесс исполнения команд, что потребовалось для создания теоретической таблицы трассировки. Умение статически анализировать код программы, без запуска и отладки, помогает не только в лабораторных заданиях, но и в настоящих проблемах (например, такая техника используется для реверс-инжиниринга).