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 №2
"Исследование работы БЭВМ"
по курсу
"Основы профессиональной деятельности"

Группа: **P3131**
Студент: Друян Эдуард Андреевич
Вариант: **3109**

Санкт-Петербург
2021 г.

Цель работы

Изучение приемов работы на базовой ЭВМ и исследование порядка выполнения арифметических команд и команд пересылки.

Задание

По выданному преподавателем варианту определить функцию, вычисляемую программой, область представления и область допустимых значений исходных данных и результата, выполнить трассировку программы, предложить вариант с меньшим числом команд. При выполнении работы представлять результат и все операнды арифметических операций знаковыми числами, а логических операций набором из шестнадцати логических значений.

Подготовка к выполнению работы

Познакомиться с устройством и системой команд базовой ЭВМ (см. приложение В, п.п. 1.1 - 1.6), порядком выполнения машинных команд (п.1.9), инструкцией по работе с моделью базовой ЭВМ (см. приложение Г). Изучить представления в БЭВМ числовых и логических значений.

Порядок выполнения работы

Восстановить текст заданного варианта программы, отделить ячейки данных от кода программы, написать назначение программы и реализуемую функцию, которую представить в виде формулы.

Во время допуска к работе получить у преподавателя исходные данные для переменных, согласовать вариант программы для исполнения, занести в память базовой ЭВМ заданный вариант программы и, выполняя ее по командам, заполнить таблицу трассировки выполненной программы. Занесение программы с данными, а также запуск программы в пультовом режиме продемонстрировать преподавателю.

Содержание отчета по работе

1. Текст исходной программы.

Adress	Code	Mnemonics	Comments
097	0200	CLA	0 -> AC
098	30A3	OR	AC | M -> AC
099	30A0	OR	AC | M -> AC
09A	E0A2	ST	AC -> M
09B	A0A1	LD	M -> AC
09C	60A2	SUB	AC - M -> AC
09D	E09F	ST	AC -> M
09E	0100	HLT	Halt
09F	0200	-	res
0A0	0200	-	v_0
0A1	0200	-	v_1
0A2	30A0	-	v_2
0A3	0200	-	v_3

2. Описание программы:

- *назначение программы и реализуемые ею функции (формула);*

Назначение программы: арифметическое и побитово-логическое вычисление по заданной формуле.

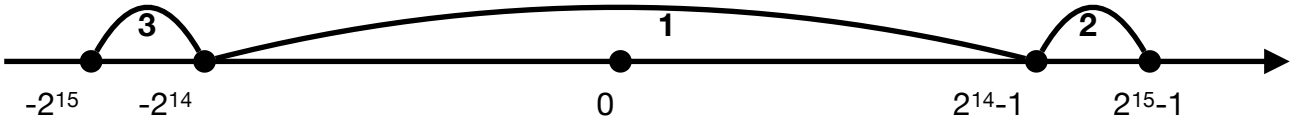
Формула:

$$res = v_1 - (v_3 \,|\, v_0),$$

`v2 = (v3 | v0)`, где `res`, v_1, v_3, v_0 - заданные переменные.

- *область представления и область допустимых значений исходных данных и результата;*

Область представления:
`res`, v_1, v_2 - знаковые, 16-ти разрядные числа;
v_3, v_0 - наборы из 16 логических 1-битовых значений.
Область допустимых значений:
Рассмотрю 3 случая для v_1 (снизу значения, сверху номер случая):



Случай 1:

$$\begin{cases} v_1 \in [-2^{14},\ 2^{14}-1]; \\ -v_2 \in [-2^{14},\ 2^{14}],\ v_2 \in [-2^{14},\ 2^{14}]; \\ res \in [-2^{15},\ 2^{15}-1]; \\ \left[\begin{array}{c} v_2[15] = v_2[14] = 0; \\ v_2[14] = 1,\ v_2[i] = 0,\ i \in \{0,1,...,13,15\}; \\ v_2[15] = v_2[14] = 1. \end{array}\right. \end{cases}$$

Случай 2:

$$\begin{cases} v_1 \in [2^{14},\ 2^{15}-1]; \\ -v_2 \in [-2^{15}+1,\ 0],\ v_2 \in [0,\ 2^{15}-1]; \\ res \in [-2^{15}+2^{14}+1,\ 2^{15}-1]; \\ v_2[15] = 0. \end{cases}$$

Случай 3:

$$\begin{cases} v_1 \in [-2^{15},\ -2^{14}-1]; \\ -v_2 \in [0,\ 2^{15}-1],\ v_2 \in [-2^{15}+1,\ 0]; \\ res \in [-2^{15},\ 2^{15}-2^{14}-2]; \\ v_2[15] = 1; \\ v_2 \neq -2^{15}. \end{cases}$$

- *расположение в памяти ЭВМ программы, исходных данных и результатов;*

Указано в комментариях текста исходной программы.

- *адреса первой и последней выполняемой команд программы.*

Первая команда расположена по адресу: `097`.
Последняя команда расположена по адресу: `09E`.

3. Таблица трассировки.

Executing command		Content of processor registers after command execution								Modified cell	
Adress	Code	AC	BR	DR	CR	AR	IP	PS	NZVC	Adress	Code
097	0200	0000	0097	0200	0200	097	098	004	0100	-	-
098	30A3	0200	FDFF	0200	30A3	0A3	099	000	0000	-	-
099	30A0	0200	FDFF	0200	30A0	0A0	09A	000	0000	-	-
09A	E0A2	0200	009A	0200	E0A2	0A2	09B	000	0000	0A2	0200
09B	A0A1	0200	009B	0200	A0A1	0A1	09C	000	0000	-	-
09C	60A2	0000	009C	0200	60A2	0A2	09D	005	0101	-	-
09D	E09F	0000	009D	0000	E09F	09F	09E	005	0101	09F	0000
09E	0100	0000	009E	0100	0100	09E	09F	005	0101	-	-

4. Вариант программы с меньшим числом команд.

Adress	Code	Mnemonics	Comments
020	3029	OR	AC | v3 -> AC
021	3027	OR	AC | v0 -> AC
022	6028	SUB	AC - v1 -> AC
023	0780	NEG	^AC + 1 -> AC
024	E026	ST	AC -> res
025	0100	HLT	Halt
026	0200	-	res
027	0200	-	v_0
028	0200	-	v_1
029	0200	-	v_3

Выводы

Изучил некоторые приемы работы на базовой ЭВМ и исследовал порядок выполнения арифметических команд и команд пересылки.

PROGRAM EXECUTION BY TICK

Start
NZVC(0100)
PS(004)
Control Unit: Operator panel
0 -> AC,BR,DR,CR,SP,AR
    PS -> CU: Operator panel
    CU: Execution
    PS -> CU: Interrupt
    -
    PS -> Control Unit

097 CLA 0200 0 -> AC
Instruction Fetch:
{IP (097)-> BR; IP (097)-> AR}
{BR (098)-> IP; M(AR) (0200)-> DR}
DR (0200)-> CR
    CR -> CU: IF (3 free-taps)
    CR -> CU: EX (4 free-taps)
Execution:
(0000)-> AC
    PS -> CU: EX
    PS -> CU: Interrupt
    -
    PS -> CU

098 OR 30A3 AC | M -> AC
Instruction Fetch:
{IP (098)-> BR; IP (098)-> AR}
{BR (099)-> IP; M(AR) (30A3)-> DR}
DR (30A3)-> CR
    CR -> CU: IF (3 free-taps)
    CR -> CU: OF (0 free-taps)
Operand Fetch:
DR (0A3)-> AR
M(AR) (0200)-> DR
    CR -> CU: EX (2 free-taps)
Execution:
AC(0000) | DR(0200) -> BR(FDFF)
BR (0200)-> AC
    PS(000)
    NZVC(0000)
    PS -> CU: EX
    PS -> CU: Interrupt
    -
    PS -> CU

099 OR 30A0 AC | M -> AC
Instruction Fetch:
{IP (099)-> BR; IP (099)-> AR}
{BR (09A)-> IP; M(AR) (30A0)-> DR}
DR (30A0)-> CR
    CR -> CU: IF (3 free-taps)
    CR -> CU: OF (0 free-taps)
Operand Fetch:
DR (0A0)-> AR
M(AR) (0200)-> DR
    CR -> CU: EX (2 free-taps)
Execution:
AC(0200) | DR(0200) -> BR(FDFF)
BR (0200)-> AC
    PS -> CU: EX
    PS -> CU: Interrupt
    -
    PS -> CU

09A ST E0A2 AC -> M
Instruction Fetch:
{IP (09A)-> BR; IP (09A)-> AR}
{BR (09B)-> IP; M(AR) (E0A2)-> DR}
DR (E0A2)-> CR
    CR -> CU: IF (4 free-taps)
    CR -> CU: OF (1 free-taps)
    CR -> CU: EX (0 free-taps)
Execution:
DR (0A2)-> AR
AC (0200)-> DR
DR (0200) -> M(AR)
    PS -> CU: EX
    PS -> CU: Interrupt
    -
    PS -> CU

09B LD A0A1 M -> AC
Instruction Fetch:
{IP (09B)-> BR; IP (09B)-> AR}
{BR (09C)-> IP; M(AR) (A0A1)-> DR}
DR (A0A1)-> CR
    CR -> CU: IF (2 free-taps)
    CR -> CU: OF (1 free-taps)
Operand Fetch:
DR (0A1)-> AR
M(AR) (0200)-> DR
    CR -> CU: EX (2 free-taps)
Execution:
DR (0200)-> AC
    PS -> CU: EX
    PS -> CU: Interrupt
    -
    PS -> CU

09C SUB 60A2 AC – M -> AC
Instruction Fetch:
{IP (09C)-> BR; IP (09C)-> AR}
{BR (09D)-> IP; M(AR) (60A2)-> DR}
DR (60A2)-> CR
    CR -> CU: IF (2 free-taps)
    CR -> CU: OF (0 free-taps)
Operand Fetch:
DR (0A2)-> AR
M(AR) (0200)-> DR
    CR -> CU: EX (3 free-taps)
Execution:
AC(0200) – DR(0200) -> AC(0000)
    PS(005)
    NZVC(0101)
    PS -> CU: EX
    PS -> CU: Interrupt
    –
    PS -> CU

09D ST E09F AC -> M
Instruction Fetch:
{IP (09D)-> BR; IP (09D)-> AR}
{BR (09E)-> IP; M(AR) (E09F)-> DR}
DR (E09F)-> CR
    CR -> CU: IF (4 free-taps)
    CR -> CU: OF (1 free-taps)
    CR -> CU: EX (0 free-taps)
Execution:
DR (09F)-> AR
AC (0000)-> DR
DR (0000)-> M(AR)
    PS -> CU: EX
    PS -> CU: Interrupt
    –
    PS -> CU

09E HLT 0100 Halt
Instruction Fetch:
{IP (09E)-> BR; IP (09E)-> AR}
{BR (09F)-> IP; M(AR) (0100)-> DR}
DR (0100)-> CR
    CR -> CU: IF (2 free-taps)
    CR -> CU: EX (3 free-taps)
    PS -> CU: EX
    –

SHORT PROGRAMM#1 (losing –2^15)
----
020 OR 3029 AC | v3 -> AC
021 OR 3027 AC | v0 -> AC
022 SUB 6028 AC – v1 -> AC
023 NEG 0780 ^AC + 1 -> AC
024 ST E026 AC -> res
025 HLT 0100 Halt
026 –   xxxx res
027 –   xxxx v0
028 –   xxxx v1
029 –   xxxx v3

6 commands, 4 variables.

SHORT PROGRAMM#2
----
030 OR 303B AC | v3 -> AC
031 OR 3038 AC | v0 -> AC
032 ST E03A AC -> v2
033 LD A039 v1 -> AC
034 SUB 603A AC – v2 -> AC
035 ST E037 AC -> res
036 HLT 0100 Halt
037 –   xxxx res
038 –   xxxx v0
039 –   xxxx v1
03A –   xxxx v2
03B –   xxxx v3

7 commands, 5 variables.