**Лабораторна  робота №5**.

**Тема:**   «Дослідження роботи моделі центрального процесора»

**Зміст роботи:**

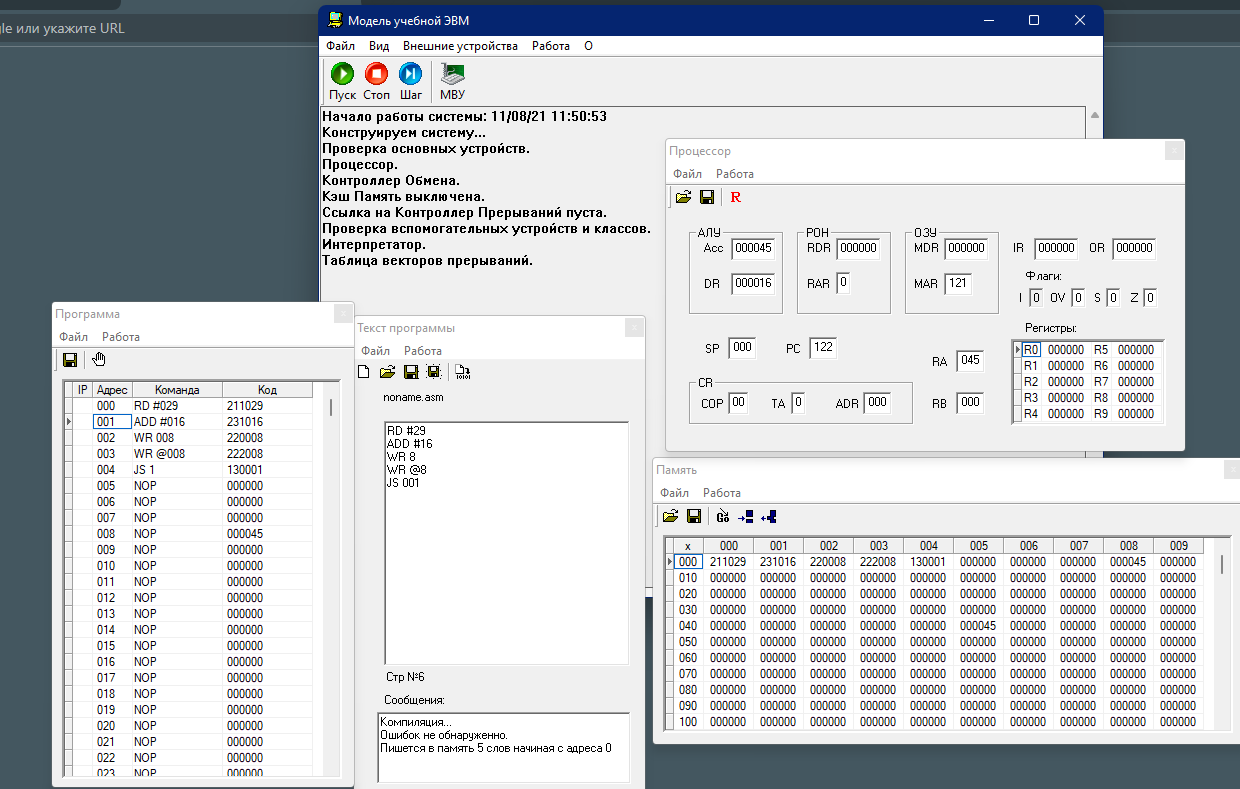
*Навчитися закладати програму в пам’ять та виконувати програмування простих математичних дій.*

**План виконання роботи**

1. Ознайомитися з теоретичними відомостями з архітектури ЕОМ
2. Запустіть програму CompModel.exe.
3. Записати в ОЗУ "програму", що складається з п'яти команд. Номер завдання вибрати з таблиці, відповідно до номеру Вашого варіанту, який співпадає з номером по журналу. Команди розмістити в послідовних комірках пам'яті.

Таблиця 1. Варіанти завдань

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № варіанту | | | **Команда 1** | | **Команда 2** | **Команда 3** | **Команда 4** | | **Команда 5** |
| **1** | **15** |  | RD #07 | | MUL #2 | WR 10 | WR @10 | | JNS 001 |
| **2** | **16** |  | RD #17 | | SUB #9 | WR 16 | WR @16 | | JNS 001 |
| **3** | **17** | **29** | RD #29 | | ADD #16 | WR 8 | WR @8 | | JS 001 |
| **4** | **18** |  | RD #2 | | MUL #6 | WR 11 | WR @11 | | JNZ 00 |
| **5** | **19** | **30** | RD #16 | | WR 8 | DIV #4 | WR @8 | | JMP 002 |
| **6** | **20** |  | RD #4 | | WR 11 | RD @11 | ADD #330 | | JS 000 |
| **7** | **21** | **31** | RD #12 | | WR 9 | RD @9 | SUB #1 | | JS 001 |
| **8** | **22** |  | RD #4 | | SUB #8 | WR 8 | WR @8 | | JNZ 001 |
|  |  |  |  | |  |  |  | |  |
| **9** | **23** | **32** | RD #15 | | ADD #12 | WR 10 | WR @10 | | JS 004 |
| **10** | **24** |  | RD #4 | | ADD #15 | WR 13 | WR @13 | | JMP 001 |
| **11** | **25** | **33** | RD #315 | | SUB #308 | WR 11 | WR @11 | | JMP 001 |
| **12** | **26** |  | RD #988 | | ADD #19 | WR 9 | WR @9 | | JNZ 001 |
| **13** | **27** | **34** | RD #017 | | WR 11 | ADD 11 | WR @11 | | JMP 002 |
| **14** | **28** |  | RD #5 | | MUL #9 | WR 10 | WR @10 | | JNZ 001 |
| **№ комірки пам’яті** | | | | **Код команди на мові асемблера** | | | | **Коментар до виконаної команди** | |
| 000 | | | | RD #29 | | | | Читаю число 29 | |
| 001 | | | | ADD #16 | | | | Додаю число 16 | |
| 002 | | | | WR 8 | | | | Записую у пам’ять за адресою 8 | |
| 003 | | | | WR @8 | | | | Записую за адресою 45 | |
| 004 | | | | JS 001 | | | | Повернення на адресу 1, якщо відповідь <0 | |



1. Виконати наступні завдання:

**Завдання 1.** Обчислити значення **S**:

**S = M + D + X = 44**

де: **M** - місяць Вашого народження;

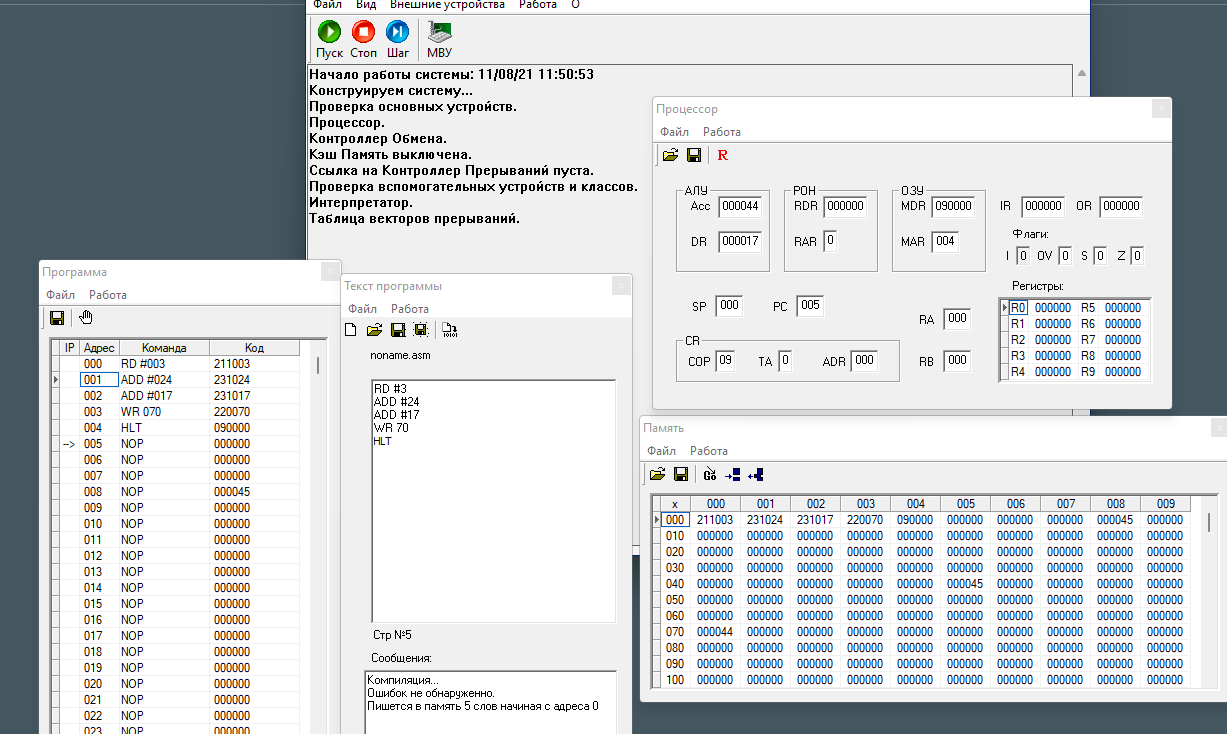
**D** - день Вашого народження;

**X** - номер Вашого варіанту.

Результат зберегти в комірку пам’яті за адресою **070**.

3+24+17=44

|  |  |  |
| --- | --- | --- |
| **№ комірки пам’яті** | **Код команди на мові асемблера** | **Коментар до виконаної команди** |
| 000 | RD #3 | Читаю число 3 |
| 001 | ADD #24 | Додаю число 24 |
| 002 | ADD #17 | Додаю число 17 |
| 003 | WR 70 | Записую у пам’ять за адресою 70 |
| 004 | HLT | Кінець програми |



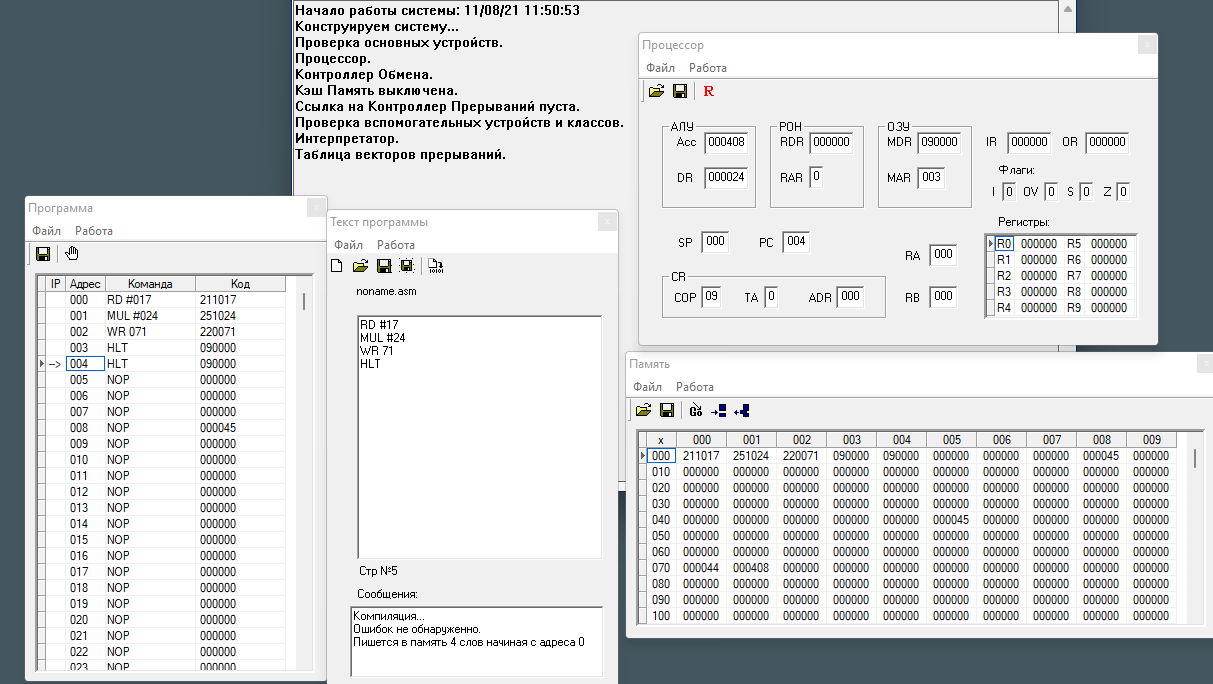
**Завдання 2.** Обчислити значення **A**:

**A = X x D = 408** (**X** помножити на **D**);

Результат зберегти в комірку пам’яті за адресою **071**.

17\*24=408

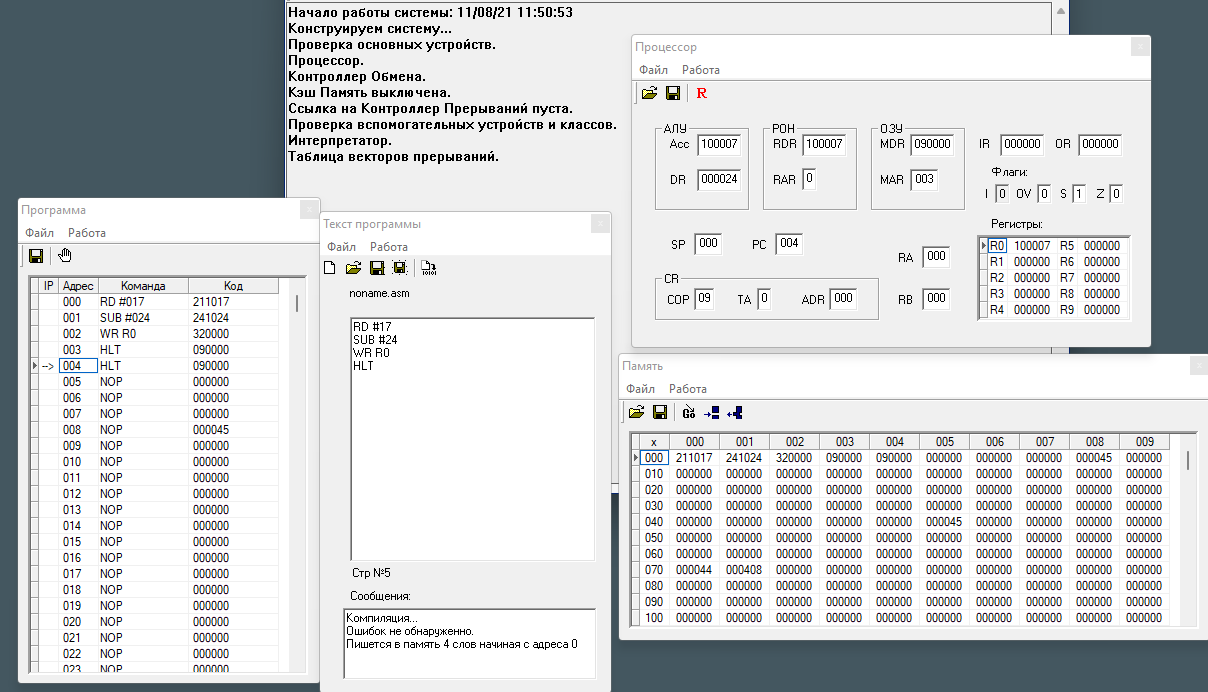
|  |  |  |
| --- | --- | --- |
| **№ комірки пам’яті** | **Код команди на мові асемблера** | **Коментар до виконаної команди** |
| 000 | RD #17 | Читаю число 17 |
| 001 | MUL #24 | Множу до числа 24 |
| 002 | WR 71 | Записую у пам’ять за адресою 71 |
| 003 | HLT | Кінець програми |

****

**Завдання 3**. Обчислити значення:

**B** = **X** – **D = -7**;

|  |  |  |
| --- | --- | --- |
| **№ комірки пам’яті** | **Код команди на мові асемблера** | **Коментар до виконаної команди** |
| 000 | RD #17 | Читаю число 17 |
| 001 | SUB #24 | Віднімаю 24 |
| 002 | WR R0 | Записую у регістр R0 |
| 003 | HLT | Кінець програми |

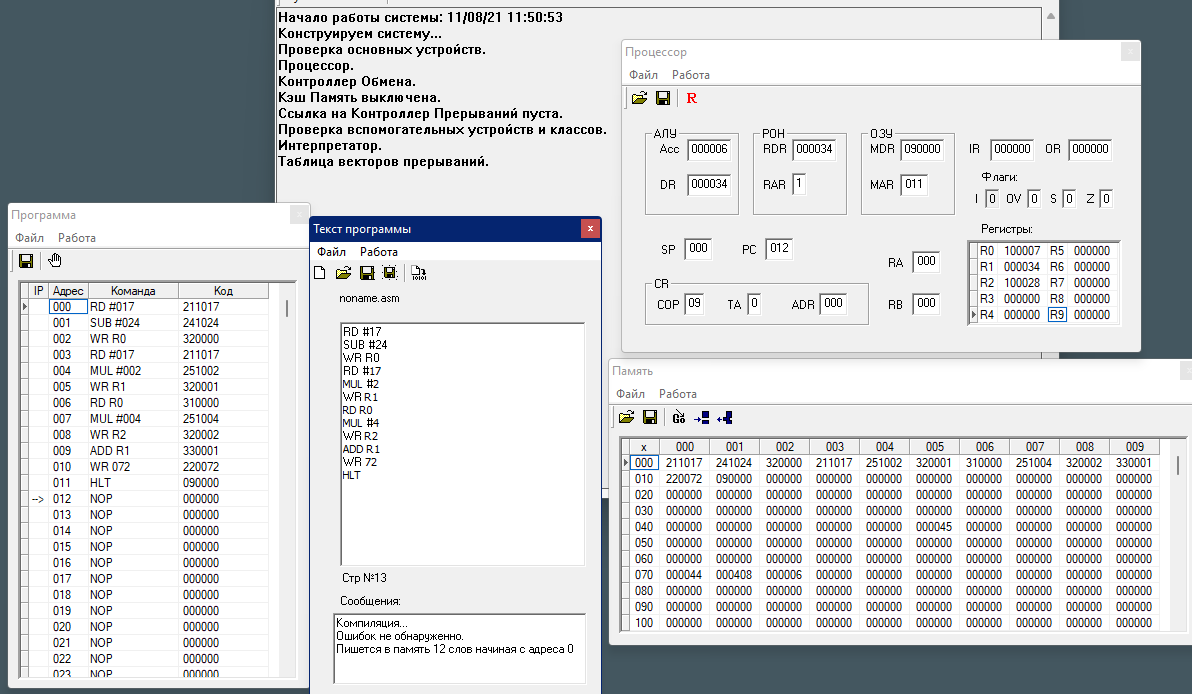


**Завдання 4**. Обчислити значення:

**C = X \* 2 + B \* 4** *(де значення В взяте із попереднього завдання)*

Для зберігання проміжних результатів використовувати регістри **R1** та **R2**. Результат зберегти в комірку пам’яті за адресою **072**.

|  |  |  |
| --- | --- | --- |
| **№ комірки пам’яті** | **Код команди на мові асемблера** | **Коментар до виконаної команди** |
| 000 | RD #17 | Читаю число 17 |
| 001 | MUL #2 | Помножити на 2 |
| 002 | WR R1 | Записую у регістр R1 |
| 003 | RD R0 | Прочитати регістр R0 |
| 004 | MUL #4 | Помножити на 4 |
| 005 | WR R2 | Записую у регістр R2 |
| 006 | ADD R1 | Додати регістр R1 |
| 007 | WR 72 | Записую у пам’ять за адресою 72 |
| 008 | HLT | Кінець програми |



1. Вибрати математичні розрахунки згідно варіанту.
2. Скласти програму обчислення першого математичного розрахунку. При написанні програми спочатку всі змінні розташувати лише в регістрах починаючи із регістра **R1**. Регістр **R0** зарезервувати за стеком. (***За початок стеку відповідає регістр SP***).
3. При складанні програми пам’ятати як працює стек, та розташовувати математичні дії в послідовності за правилом – останній прийшов – перший вийшов.
4. Результат розташувати в комірці пам’яті **100**
5. При виконанні звіту текст з програмою розташувати в таблиці та прокоментувати кожну дію. Таблиця приведена для прикладу:

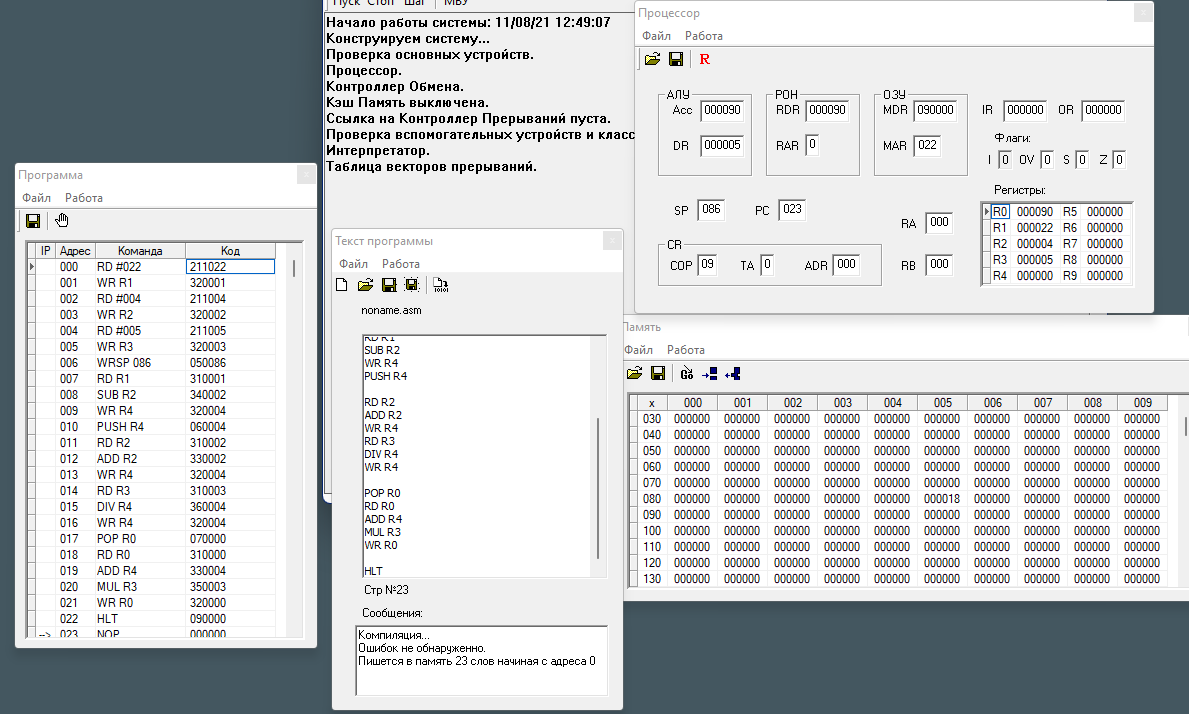
|  |  |  |
| --- | --- | --- |
| **Адреса в пам’яті** | **Команда у**  **мнемокоді** | **Коментар** |
| 000 | *IN* | Зчитування X из регістра IR |
| 001 | *WR 50* | Запис X в 50 комірку пам’яті |

1. У другому математичному розрахунку використовуються від’ємні числа. Знайти у системі команд навчального процесору команди для роботи з від’ємними числами.
2. Скласти програму обчислення другого математичного розрахунку.
3. При складанні програми використовувати виключно регістри починаючи із регістра **R2**. Результат скласти у регістр результату – **R9** та розташувати в комірці пам’яті номер **50**. Стек при цих розрахунках не використовується.
4. Зробити перенос результату обчислення із регістра результату **R9** в регістр **R0**.
5. При виконанні звіту текст з програмою розташувати в таблиці та прокоментувати кожну дію.
6. Таблиця варіантів:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № варіанту | | | | **Перший математичний розрахунок** | **Початок стеку** | **Другий математичний розрахунок** |
| *1* | *11* | *21* | *31* | ((67 – 33) – (22 + 3)) : 3 | *72* | (42 – -4) + -24 |
| *2* | *12* | *22* |  | ((27 + 53) – (12 х 2)) : (5 - 2) | *62* | -6 + (2 х -6) – -10 |
| *3* | *13* | *23* | *32* | (4 х 14) : (12 – 4 х 4) | *60* | 78 + -7 х 8 |
| *4* | *14* | *24* |  | 7 + 7 х 25 – (252) | *90* | (-14 + -14) – 7 |
| *5* | *15* | *25* |  | (57 х 2) : ((12 + 2) х 2) | *95* | 11 – -15 + (-2 х 11) |
| *6* | *16* | *26* | *33* | (31 + 2)2 х ((15 – 31) : 2) | *55* | (55 + -5) –  -12 |
| *7* | *17* | *27* |  | ((22 – 4) + 5 : (4 + 4)) х 5 | *85* | 13 – -5 + (-2 + -5) |
| *8* | *18* | *28* | *34* | (4 х 7 + 4) х (72 : 4) | *65* | (-8 + -5) : -8 |
| *9* | *19* | *29* |  | ((86 – 8) : 4)2 - 86 | *72* | (12 + -15) – -3 |
| *10* | *20* | *30* |  | (75 + 7 + 5 х 7) : (5 – 2) | *81* | -75 + (-2 х 9) |

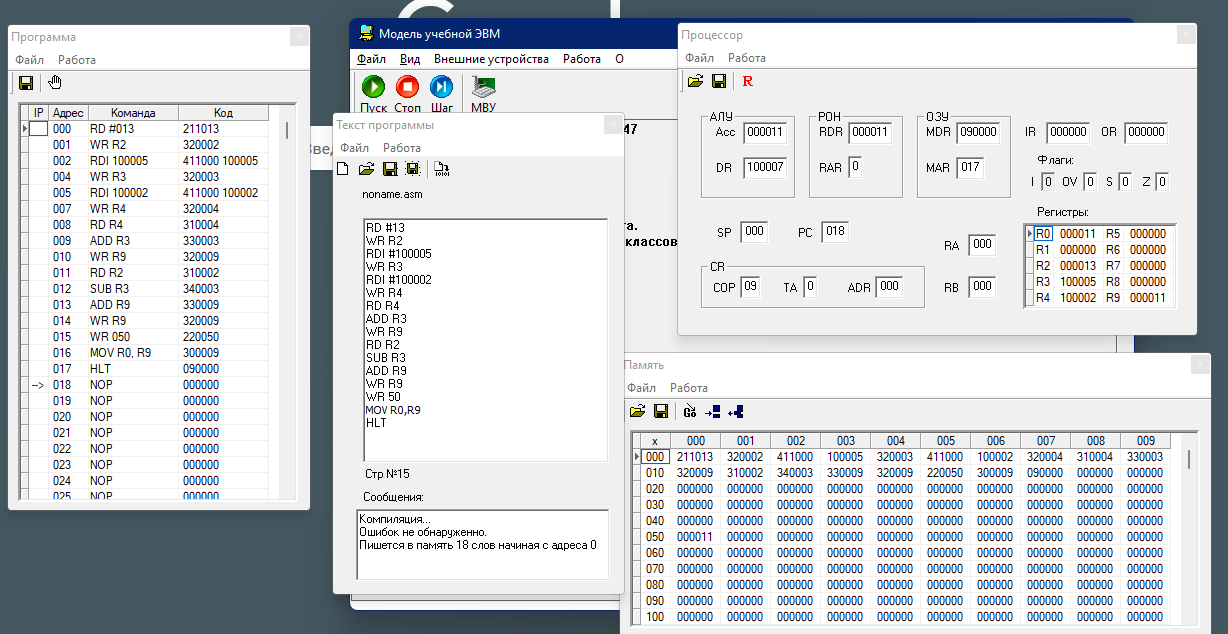
Перший математичний розрахунок:

|  |  |  |
| --- | --- | --- |
| **№ комірки пам’яті** | **Код команди на мові асемблера** | **Коментар до виконаної команди** |
| 000 | RD #22 | Читаю число 22 |
| 001 | WR R1 | Записую у регістр R1 |
| 002 | RD #4 | Читаю число 4 |
| 003 | WR R2 | Записую у регістр R2 |
| 004 | RD #5 | Читаю число 5 |
| 005 | WR R3 | Записую у регістр R3 |
| 006 | WRSP 86 | Початок стеку за адресою 85 |
| 007 | RD R1 | Читаю регістр R1 |
| 008 | SUB R2 | Віднімаю від регістра R2 |
| 009 | WR R4 | Записую у регістр R4 |
| 010 | PUSH R4 | Записую у стек |
| 011 | RD R2 | Читаю регістр R2 |
| 012 | ADD R2 | Додаю регістр R2 |
| 013 | WR R4 | Записую у регістр R4 |
| 014 | RD R3 | Читаю регістр R3 |
| 015 | DIV R4 | Ділю на регістр R4 |
| 016 | WR R4 | Записую у регістр R4 |
| 017 | POP R0 | Витягую зі стека у регістр R0 |
| 018 | RD R0 | Читаю регістр R0 |
| 019 | ADD R4 | Додаю регістр R4 |
| 020 | MUL R3 | Множу на регістр R3 |
| 021 | WR R0 | Записую у регістр R0 |
| 022 | HLT | Кінець програми |



Другий математичний розрахунок:

|  |  |  |
| --- | --- | --- |
| **№ комірки пам’яті** | **Код команди на мові асемблера** | **Коментар до виконаної команди** |
| 000 | RD #13 | Читаю число 13 |
| 001 | WR R2 | Записую у регістр R2 |
| 002 | RDI #100005 | Читаю число -5 |
| 003 | WR R3 | Записую у регістр R3 |
| 004 | RDI #100002 | Читаю число -2 |
| 005 | WR R4 | Записую у регістр R4 |
| 006 | RD R4 | Читаю регістр R4 |
| 007 | ADD R3 | Додаю регістр R3 |
| 008 | WR R9 | Записую у регістр R9 |
| 009 | RD R2 | Читаю регістр R2 |
| 010 | SUB R3 | Віднімаю регістр R3 |
| 011 | ADD R9 | Додаю регістр R9 |
| 012 | WR R9 | Записую у регістр R9 |
| 013 | WR 50 | Записую за адресою 50 |
| 014 | MOV R0,R9 | Перенос регістра R9 в R0 |
| 015 | HLT | Кінець програми |



|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| RD #29 | Читаю число 29 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 --> MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 Acc := ADR | вміст регістру даних пам'яті переміщено в акумулятор |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| ADD #16 | Додаю число 16 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 --> MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 DR := ADR | вміст регістру даних пам'яті переміщено в акумулятор |
| 05 ALU <-- COP | кінець цієї команди та готовність переходу до наступної |
| 06 Start ALU | починається робота арифметично-логічного пристрою |
| 07 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| WR R2 | запис у регістр R2 значення із акумулятора |
| 00      MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 --> MRd | читання даних за вказаною адресою |
| 02      CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03      PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04      RAR := CR5 | регистр команд, содержащий поля копіюється в регистр адреса блока РОН |
| 05      RDR := Acc | Вміст акумулятора копіюється в регистр данных блока РОН |
| 06      RWr | Сигнал для доступу регістрів |
| 07      END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| JS 001 | Повернення на адресу 1, якщо відповідь <0 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 JS | Переход на адрес |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| HLT | Кінець програми |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 HALT | стоп |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| MUL #24 | Помножити на 24 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 DR := ADR | вміст регістру даних пам'яті переміщено в акумулятор |
| 05 ALU <-- COP | кінець цієї команди та готовність переходу до наступної |
| 06 Start ALU | починається робота арифметично-логічного пристрою |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| SUB #24 | Відняти 24 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 DR := ADR | вміст регістру даних пам'яті переміщено в акумулятор |
| 05 ALU <-- COP | кінець цієї команди та готовність переходу до наступної |
| 06 Start ALU | починається робота арифметично-логічного пристрою |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| WRSP 86 | Початок стеку за адресою 85 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 SP := ADR | адрес или непосредственный копіюється в указатель стека, содержащий адрес верхушки стека |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| PUSH R4 | Записую у стек |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 RAR := CR5 | регистр команд, содержащий поля копіюється в регистр адреса блока РОН |
| 05 RRd | Виявлення, з яким регістром працювати |
| 06 MDR := RDR | регистр адреса блока РОН копіюється в регистр данных ОЗУ |
| 07 SP := SP-1 | указатель стека, содержащий адрес верхушки стека, зменшується на 1 |
| 08 MAR := SP | указатель стека, содержащий адрес верхушки стека копіюється в регистр адреса ОЗУ |
| 09 MWr | По сигналу MWr выполняется запись содержимого регистра данных (MDR) в ячейку памяти с адресом, указанным в регистре адреса (MAR) |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| DIV R4 | Ділю на регістр R4 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 RAR := CR5 | регистр команд, содержащий поля копіюється в регистр адреса блока РОН |
| 05 RRd | Виявлення, з яким регістром працювати |
| 06 DR := RDR | регистр адреса блока РОН копіюється в регистр данных ОЗУ |
| 07 ALU <-- COP | кінець цієї команди та готовність переходу до наступної |
| 08 Start ALU | починається робота арифметично-логічного пристрою |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| POP R0 | Витягую зі стека у регістр R0 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 MAR := SP | указатель стека, содержащий адрес верхушки стека копіюється в регистр адреса ОЗУ |
| 05 RRd | Виявлення, з яким регістром працювати |
| 06 RDR := MDR | регистр данных ОЗУ копіюється в регистр данных блока РОН |
| 07 SP := SP+1 | указатель стека, содержащий адрес верхушки стека, збільщується на 1 |
| 08 RAR := CR5 | регистр команд, содержащий поля копіюється в регистр адреса блока РОН |
| 09 RWr | Сигнал для доступу регістрів |
| 05 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

|  |  |
| --- | --- |
| **Сама команда та її мікрооперації** | **Коментар до процесу виконання команди** |
| MOV R0,R9 | Перенос регістра R9 в R0 |
| 00 MAR := PC | передача значення регістру лічильника команд у регістр адресу оперативної  пам’яті |
| 01 MRd | читання даних за вказаною адресою |
| 02 CR := MDR | вміст MDR копіюється в регістр команд (CR) |
| 03 --> PC := PC+1 | содержащий адрес текущей команды, увеличивается на 1 |
| 04 RAR := CR5 | регистр команд, содержащий поля копіюється в регистр адреса блока РОН |
| 05 RRd | Виявлення, з яким регістром працювати |
| 06 RAR := CR4 | Регістр команд 4 копіюється в регистр адреса блока РОН |
| 07 RWr | Сигнал для доступу регістрів |
| 08 END\_COMMAND | кінець цієї команди та готовність переходу до наступної |

Висновок: я навчитися закладати програму в пам’ять та виконувати програмування простих математичних дій.