**3 ЛАБОРАТОРНАЯ РАБОТА № 3**

**ПОДПРОГРАММЫ И СТЕК. КОМАНДНЫЙ ЦИКЛ ПРОЦЕССОРА**

**3.1 Цель работы**

Реализация подпрограмм. Исследование работы микрокоманд.

**3.2 Краткие теоретические сведения**

В программировании часто встречаются ситуации, когда одинаковые действия необходимо выполнять многократно в разных частях программы (например, вычисление функции sinx). При этом с целью экономии памяти не следует многократно повторять одну и ту же последовательность команд - достаточно один раз написать так называемую подпрограмму (в терминах языков высокого уровня - процедуру) и обеспечить правильный вызов этой подпрограммы и возврат в точку вызова по завершению подпрограммы. Для вызова подпрограммы необходимо указать ее начальный адрес в памяти и передать (если необходимо) параметры - те исходные данные, с которыми будут выполняться предусмотренные в подпрограмме действия. Адрес подпрограммы указывается в команде вызова call, а параметры могут передаваться через определенные ячейки памяти, регистры или стек. Возврат в точку вызова обеспечивается сохранением адреса текущей команды (содержимого регистра PC) при вызове и использованием в конце подпрограммы команды возврата ret, которая возвращает сохраненное значение адреса возврата в PC.

**3.3 Выполнение лабораторной работы**

Данное задание выполнено согласно варианту 2. Найти самое большое отрицательное число. В таблице 3.1 приведен код программы и описание действий к каждой команде.

Таблица 3.1 – код программы

|  |  |
| --- | --- |
| RD #100 | Чтение стартового адреса первого массива |
| WR R1 | Заносим его в регистр |
| RD #5 | Чтение количества |
| WR R2 | Заносим в регистр |
| CALL M | Вызов процедуры обработки |
| WR R6 | Заносим результат |
| RD #110 | Чтение стартового адреса второго массива |
| WR R1 | Заносим его в регистр |
| RD #5 | Чтение количества |
| WR R2 | Заносим в регистр |
| CALL M | Вызов процедуры обработки |
| WR R7 | Заносим результат |
| RD #120 | Чтение стартового адреса третьего массива |
| WR R1 | Заносим его в регистр |
| RD #5 | Чтение количества |
| WR R2 | Заносим в регистр |
| CALL M | Вызов процедуры обработки |
| ADD R6 | Чтение промежуточных результатов |
| ADD R7 | Чтение промежуточных результатов |
| DIV #3 | Делим на три |
| OUT | Выводим результат |
| HLT | Остановка конец программы |
| M: RD @R1 | Чтение первого числа массива |
| WR R3 | Заносим его значения во временный регистр |
| L2: RD @R1+ | Читаем следующий за ним элемент |
| WR R4 | Заносим его во временный буфер РОН 4 |
| JNS L1 | Если число положительное переходим на метку L1 |
| SUB R3 | Сравниваем два элемента |
| JNS L1 | Если R4 < R3 |
| MOV R3,R4 | заносим новое макс отрицательное |
| L1: JRNZ R2,L2 | Сравнение выполнение прыжка пока не ноль |
| RD R3 | Чтение результата |
| RET | Возврат в точку вызова |

Результаты исследования по второй части работы таблица 3.2.

Таблица 3.2 результат исследования микро команд.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ячейки | | АУ | | CR | | | ОЗУ | | Микро команда | Мнемокод | Адрес |
| 16 | @16 | DR | ACC | ADR | TA | ADR | MDR | MAR |  | RD #17 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00 MAR := PC |  |  |
|  |  | 0 | 0 |  |  |  | 211017 | 000 | 01 --> MRd |  |  |
|  |  |  |  | 21 | 1 | 017 |  |  | 02 CR := MDR |  |  |
|  |  |  |  |  |  |  |  |  | 03 PC := PC+1 |  |  |
|  |  |  | 17 |  |  |  |  |  | 04 Acc := ADR |  |  |
|  |  |  |  |  |  |  |  |  | END\_COMMAND |  |  |
|  |  |  |  |  |  |  |  |  | MAR := PC | SUB #9 | 1 |
|  |  |  |  |  |  |  |  |  | MRd |  |  |
|  |  |  |  |  |  |  | 241009 | 001 | CR := MDR |  |  |
|  |  |  |  | 24 | 1 | 009 |  |  | PC := PC+1 |  |  |
|  |  | 9 |  |  |  |  |  |  | DR := ADR |  |  |
|  |  |  |  |  |  |  |  |  | ALU <-- COP |  |  |
|  |  |  | 8 |  |  |  |  |  | Start ALU |  |  |
|  |  |  |  |  |  |  |  |  | END\_ |  |  |
|  |  |  |  |  |  |  |  |  | MAR := PC | WR 16 | 2 |
|  |  |  |  |  |  |  | 220016 | 002 | MRd |  |  |
|  |  |  |  | 22 | 00 | 016 |  |  | CR := MDR |  |  |
|  |  |  |  |  |  |  |  |  | PC := PC+1 |  |  |
|  |  |  |  |  |  |  |  |  | MAR := ADR |  |  |
|  |  |  |  |  |  |  | 8 | 016 | MDR := Acc |  |  |
| 8 |  |  |  |  |  |  |  |  | MWr |  |  |
|  |  |  |  |  |  |  |  |  | END\_COMMAND |  |  |
|  |  |  |  |  |  |  |  |  | MAR := PC | WR @16 | 3 |
|  |  |  |  |  |  |  | 222016 | 003 | MRd |  |  |
|  |  |  |  | 22 | 2 | 016 |  |  | CR := MDR |  |  |
|  |  |  |  |  |  |  |  |  | PC := PC+1 |  |  |
|  |  |  |  |  |  |  |  |  | MAR := ADR |  |  |
|  |  |  |  |  |  |  | 008 | 016 | MRd |  |  |
|  |  |  |  |  |  |  |  |  | RA := MDR |  |  |
|  |  |  |  |  |  |  |  |  | MAR := RA |  |  |
|  |  |  |  |  |  |  |  |  | MDR := Acc |  |  |
|  | 8 |  |  |  |  |  |  |  | MWr |  |  |
|  |  |  |  |  |  |  |  |  | MAR := PC | JNS | 4 |
|  |  |  |  |  |  |  | 140001 | 004 | MRd |  |  |
|  |  |  |  | 14 | 0 | 001 |  |  | CR := MDR |  |  |
|  |  |  |  |  |  |  |  |  | PC := PC+1 |  |  |
|  |  |  |  |  |  |  |  |  | JNS |  | 1 |

Выполнение второй части задания 2.

ADD R3

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR5

05 RRd

06 DR := RDR

07 ALU <-- COP

08 Start ALU

09 END\_COMMAND

ADD @R3

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR5

05 RRd

06 RA := RDR

07 MAR := RA

08 MRd

09 DR := MDR

10 ALU <-- COP

11 Start ALU

12 END\_COMMAND

ADD @R3+

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR5

05 RRd

06 RA := RDR

07 MAR := RA

08 MRd

09 DR := MDR

10 INC\_GR

11 ALU <-- COP

12 Start ALU

13 END\_COMMAND

ADD -@R3

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR5

05 DEC\_GR

06 RRd

07 RA := RDR

08 MAR := RA

09 MRd

10 DR := MDR

11 ALU <-- COP

12 Start ALU

13 END\_COMMAND

JRNZ R3,M

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR2

05 DEC\_GR

06 JRNZ

07 END\_COMMAND

MOV R4,R2

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR5

05 RRd

06 RAR := CR4

07 RWr

08 END\_COMMAND

CALL M

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 MDR := PC

05 SP := SP-1

06 MAR := SP

07 MWr

08 PC := ADR

09 END\_COMMAND

RET: PUSH R3

00 MAR := PC

01 --> MRd

02 CR := MDR

03 PC := PC+1

04 RAR := CR5

05 RRd

06 MDR := RDR

07 SP := SP-1

08 MAR := SP

09 MWr

10 END\_COMMAND

POP R5

00 MAR := PC

01 MRd

02 CR := MDR

03 PC := PC+1

04 MAR := SP

05 MRd

06 --> RDR := MDR

07 SP := SP+1

08 RAR := CR5

09 RWr

10 END\_COMMAND

**3.3 Выводы**

Поскольку процессор может выполнить за один такт только одно действие, а большинство команд требуют больше, по мимо команд основных существует набор скрытых микрокоманд, которые выполняет процессор за один такт. В данной лабораторной работе был рассмотрен сам механизм выполнения команд на микрокомандном уровне.