**Реферат по лекции «Ассемблер» - Каспер Наталья**

Ассемблер работает с ячейками памяти и регистрами.

**Регистры** – ячейки памяти, которые расположены непосредственно в процессоре. Работа с регистрами выполняется быстрее, чем с ячейками оперативной памяти => регистры используют как в программах на ассемблере, так и компиляторами языков высокого уровня.

**Виды регистров:**

1. Регистры общего назначения
2. Флаги
3. Указатель команд
4. Регистры сегментов

Существует 8 регистров **общего назначения**. Размер каждого регистра – 32 бита.

**AX** – аккумулятор – для арифметических операций

**BX** – регистр базы – для указателя на данные

**CX** – счетчик – для сдвигов и циклов

**DX** – регистр данных – для арифметических операций и операций ввода/вывода

**SP** – регистр указателя стека – для указателя вершины стека

**BP** – регистр указателя базы кадра стека – для индикатора основания стека

**SI** – индекс источника – для указателя отправителя (источника)

**DI** – индекс приемника (получателя) – для указателя получателя

Язык ассемблера используется для работы со стеком, при этом:

1. для записи в стек используется команда **push**
2. для чтения из стека используется команда **pop**
3. вызов метода – **call** (при этом аргументы не указываются в скобках, а передаются через стек)
4. загрузка в регистр – **mov**
   1. move eax, offset world – загрузка слова world в регистр eax
   2. offset – указывает на то, что грузится адрес
5. компиляция – **ml имя\_файла.masm**
6. **.data** – секция для объявления сообщения
7. Слинковать прогу – **link имя\_скомпилированного\_файла**

**Математический сопроцессор**

1. **FINIT** – инициализация сопроцессора
2. **FLD aa** – загрузка в ST(0) операнда с именем аа
3. **FMUL** – умножение ST(0)\*ST(1) с занесением результата в ST(0)
4. **FLDPI** – загрузка в ST(0) числа *пи*
5. **FST aa** – запись результата в ячейку памяти аа
6. **FWAIT** – ожидание завершения работы сопроцессора

**Система команд сопроцессора**

1. **Команды передачи данных**

Предназначены для организации обмена между регистрами стека, вершиной стека сопроцессора и ячейками ОП.

|  |  |
| --- | --- |
| **fid/fild** источник | Загрузка числа из памяти в вершину стека |
| **fst/fist** приемник | Сохранение числа из вершины стека сопроцессора в память  (поле top не изменяеится) |
| **fstp/fistp** приемник | Сохранение числа из вершины стека сопроцессора в память  поле top увел.на 1=>вершина-след.больший по физ.номеру регистр |
| **fxsh** st(i) | Обмен знач.между тек.вершиной стека и регистром стека сопроц.ST(i) |

1. **Команды загрузки констант**

|  |  |
| --- | --- |
| fldz | загрузка 0 в вершину стека сопроцессора |
| fld1 | загрузка 1 в вершину стека сопроцессора |
| fldpi | загрузка числа *pi* в вершину стека сопроцессора |
| fldl2t | загрузка 2сс логарифма десяти в вершину стека сопроцессора |
| fldl2e | загрузка 2сс логарифма *е* в вершину стека сопроцессора |
| fldlg2 | загрузка 10сс логарифма 2ух в вершину стека сопроцессора |
| fldln2 | загрузка натурального логарифма 2ух в вершину стека сопроцессора |

1. **Команды сравнения данных**

Такие команды выполняют сравнение значений числа, которое находится в вершине стека и операнда, указанного в команде

|  |  |
| --- | --- |
| fcom/ficom | *без операндов:* сравнивает значения регистров st(0) и st(1)  *есть операнды:* сравнивается значение в регистре st(0) со значением в памяти |
| fcomp/ficomp | сравнивает значение в вершине стека st(0) со значение операнда, который находится в регистре или в памяти => выталк.знач из st(0) |
| ftst | *без операндов:* сравнивает значение в st(0) с 0  \*операнды несовместимы => c3=1, c2=1, c0=1;  \*операнд\_2 > операнд\_1 => c3=0, c2=0, c0=1;  \*операнд\_2 < операнд\_1 => c3=0, c2=0, c0=0;  \*операнд\_2 = операнд\_1 => c3=1, c2=0, c0=0; |
| fsrsw | записывает содержимое регистра swr в регистр ax |
| sahf | содержимое регистра ah записывается в младший байт регистра flafs/eflags |

1. **Арифметические команды**

Такие команды позволяют выполнять: сложение, вычитание, умножение, деление. Также существует несколько дополнительных команд, которые предназначены для повышения эффективности использования арифметических команд. Перечислим несколько основных команд:

|  |  |
| --- | --- |
| fiadd источник | складывает значения ST(0) и целочисленного источника |
| fisub источник | вычитает значение целочисленного источника из ST(0) |
| fimul источник | умножает значение целочисленного источника на содержимое ST(0) |
| fidiv источник | делит содержимое ST(0) на значение целочисленного источника |
| fadd | складывает значения из регистров ST(0) и ST(1) |
| fsub | вычитает из значения в регистре ST(1) значение регистра ST(0) |
| fmul | умножает значение в ST(1) на содержимое регистра в ST(0) |
| fdiv | делит содержимое регистра ST(1) на значение регистра сопроцессора ST(0) |
| fsqrt | вычисление квадратного корня из значения, находящегося в вершине стека сопроцессора ST(0) |
| frndint | команда округления до целого значения числа, которое находится в вершине стека сопроцессора |

1. **Команды трансцендентных функций**

Такие команды предназначены для вычисления тригонометрических функций – синус, косинус, тангенс, арктангенс, а также знач логарифмический и показательных функций.

|  |  |
| --- | --- |
| fcos | вычисляет косинус угла в вершине ST(0) |
| fsin | вычисляет синус угла в вершине ST(0) |
| fsincos | вычисляет синус и косинус угла в вершине ST(0) |
| fptan | вычисляет тангенс угла в радианах (в диапазоне –263…263) в ST(0) |
| fpatan | вычисляет арктангенс частного x/y (x - в ST(0), y - в ST(1)) |
| fprem | получение частичного остатка от деления |
| f2xm1 | вычисление значения функции y = 2x–1 |
| fyl2x | вычисление значения функции z = ylog2(x) |

1. **Команды управления сопроцессором**

Такие команды предназначены для общего управления работой сопроцессора.

|  |  |
| --- | --- |
| wait/fwait | для синхронизации работы процессора и сопроцессора |
| finit | для инициализации сопроцессора |
| ffree st(i) | для освобождения регистра стека st(i) |
| fstsw | для сохранения содержимого регистра состояния SWR в ячейке памяти или регистре AX |