**Білет №8**

1. ***Команди для обчислення часткових математичних функцій.***

Процессор 8087 может вычислить любую элементарную трансцен­дентную функцию с аргументом, точность представления которого оп­ределяется содержимым регистра управления, и последующей выдачей результата такой же точности. В названиях команд и их описаниях принято обозначать аргументы, содержащиеся в st[0] как X, а в st[1] - как Y. Результаты, формирующиеся в st[0], будем обозначать как x, а в st[1] - как y.

Используются такие команды: FPTAN – частичный тангенс (st/x = tg(X), 0<= X <=1 pi/4); FPATAN – частичный арктангенс (++st, arctg(Y/X), Y > X >0); FYL2X - Вычисление логарифма (++st, Y\*log2(X), X > 0); FYL2XP1 - Вычисление логарифма (++st, Y\*log2(X+1), abs(X) < 1-sqrt(2)/2 ); F2XM1 - Вычисление (X=2^X-1). Вычисление тригонометрических функций основано на выполнении команды FPTAN - нахождения частичного тангенса, которая в качестве результата дает два таких числа x и y, что y/x = tg(X). Число y заменяет старое содержимое st[0], а число включается в стек до полнительно. Диапазон изменения аргумента можно свести к допустимому коман­дой FPREM или проверить с помощью команды FXAM, так как он должен быть нормализован и находиться в диапазоне 0 < st[0] < п/4. Если аргумент Xлежит вне этого диапазона, то в начале программы необходимо выполнить преобразования по сведению аргумента к требуемому диапазону и запомнить данные для обратного пребразования. Для функции tg(X) можно использовать 4 преобразования:

X< 0; S = sign(X); = abs(X); tg(X) = S\*tg(Y); (1)

в этом перечне сначала указано условие сведения для отрицательныхаргументов, затем две формулы для получения сведенных аргументов и, наконец, формула для восстановления итогового результата.

Y> п; Z= Y - n\*п; tg(Y) = tg(Z); (2)

Здесь устраняется многократное повторение периода и дополнительных действий по восстановлению результата не требуется. Следующие два преобразования могут быть проведены в безусловной форме, так как модифицированный аргумент лежит в диапазоне 0 <= Y <= п.

U= Z/2; tg(Z) = 2\*tg(U)/(1-tg(U)\*tg(U)); (3)

V= U/2; tg(U) = 2\*tg(V)/(1-tg(V)\*tg(V)); (4)

Эти формулы, основанные на двухкратном применении известной в ма­тематике универсальной тригонометрической подстановки через тан­генс половинного угла, легко реализуются, так как деление на 2 быстро осуществляется командой FSCALE, и могут быть преобразованы с учетом того, что результат вычисления функции FPTAN(V) сформиро­ван в виде чисел и . Тогда

tg(V) = 2\*x\*y/(x\*x-y\*y) и u = 2\*x\*y; v=x\*x-y\*y, (5)

где u/v = tg(V) Эти формулы можно рассматривать как базовые для расчета тангенса и других тригонометрических функций с помощью команд FPTAN и FPREM, используя таблицу формул приведения и следую­щие формулы, выраженные через и .

sin(Y) = 2\*(u/v)/(1+(u/v)^2);

cos(Y) = (1-(u/v)^2) / (1+(u/v)^2);

cosec(Y) = (1+(u/v)^2) / 2(u/v);

sec(Y) = (1+(u/v)^2) /(1-(u/v)^2).

Кроме рассмотренного способа тригонометрические функции могут вычисляться либо через тангенс половинного угла по формуле (3), либо через тангенс полного угла по формулам:

sin(x) = tg(x) / sqrt(1+(tg(x)^2);

cos(x) = 1 / sqrt(1+(tg(x)^2);

ctg(x) = 1 / tg(x).

ПРИМЕР:

*\_tg PROC*

*PUSH BP ; Стандартное сохранение базового указателя стека.*

*MOV BP,SP ; Установка нового значения базового указателя. FLDPI ; Загрузка числа п*

*FLD QWORD PTR[BP+4] ; Загрузка начального значения суммы . FTST*

*FSTSW stsw*

*PUSH stsw*

*rm: FPREM; Исключение периода*

*fj p,rm ; Циклическое исключение остатка*

*FLD1*

*FCHS*

*FADD st,st ; Формирование - 2*

*FXCH st[1]*

*FSCALE ; Деление на 4*

*FPTAN ;Вычисление составляющих tg*

*FLD st[1] ; Дублирование числителя*

*FMUL st,st ; Квадрат числителя \*y*

*FXCH st[1] ; Обмен на знаменатель*

*FMUL st[2],st ; Вычисление \*y*

*FMUL st,st ; Вычисление \*x*

*FSUBP st[1],st ; Получение -v*

*FMUL st,st[2] ; Получение -u*

*FLD st ; Дублирование числителя u*

*FMUL st,st ; Квадрат числителя \*u*

*FXCH st[2] ; Обмен на числитель*

*FMUL st[1],st ; Вычисление \*v*

*FMUL st,st ; Вычисление \*v*

*FSUBP st[2],st ; Получение - знаменателя tg*

*FMULP st[2],st ; Получение - числителя tg*

*; Для этой команды нужно предусмотреть защиту от особых ситуаций.*

*FDIVP st[1],st ; Получение значения tg*

*FSTP result ; Сохранение результата*

*MOV AX,offset DGROUP:result*

*POP BP ; Стандартное восстановление базового указателя стека. RET ; Выход из подпрограммы.*

*\_tg ENDP*

**2:::** *Грамматики классифицируются по виду их правил вывода (по Хомскому)*

ТИП 0: Грамматика G = (VT, VN, P, S) называется *грамматикой типа 0*, если на правила вывода не накладывается никаких ограничений (кроме тех, которые указаны в определении грамматики).

ТИП 1: Грамматика G = (VT, VN, P, S) называется *неукорачивающей* *грамматикой*, если каждое правило из P имеет вид   , где   (VT  VN)+,   (VT  VN)+ и |  | ≤ |  | ≈ *контекстно-зависимой (КЗ)*, если каждое правило из P имеет вид   , где  = ξ1Aξ2;  = ξ1ξ2; A  VN;   (VT  VN)+; ξ1,ξ2  (VT  VN)\*.

*Множество языков*, *порождаемых неукорачивающими* грамматиками, *совпадает с множеством языков*, порождаемых *КЗ*-грамматиками.

ТИП 2: Грамматика G = (VT, VN, P, S) называется *контекстно-свободной (КС)*, если каждое правило из Р имеет вид A  , где A  VN,   (VT  VN)+ ≈ *укорачивающей контекстно-свободной (УКС)*, если каждое правило из Р имеет вид A  , где A  VN,   (VT  VN)\*.

Для каждой УКС-грамматики существует почти эквивалентная КС-грамматика.

ТИП 3: Грамматика G = (VT, VN, P, S) называется *праволинейной*, если каждое правило из Р имеет вид A  tB либо A  t, где A  VN, B  VN, t  VT.

Грамматика G = (VT, VN, P, S) называется *леволинейной*, если каждое правило из Р имеет вид A  Bt либо A  t, где A  VN, B  VN, t  VT.

Соотношения между типами грамматик:

(1) любая регулярная грамматика является КС-грамматикой;

(2) любая регулярная грамматика является УКС-грамматикой;

(3) любая КС-грамматика является КЗ-грамматикой;

??? [ (4) любая КС-грамматика является неукорачивающей грамматикой;

(5) любая КЗ-грамматика является грамматикой типа 0.

(6)любая неукорачивающая грамматика является грамматикой типа 0. ]

Замечание: УКС-грамматика, содержащая правила вида A → ε, не является КЗ-грамматикой и не является неукорачивающей грамматикой.

*Грамматику типа 3 (регулярную, Р-грамматику)* можно определить как праволинейную либо как леволинейную.

Множество языков, порождаемых праволинейными грамматиками, совпадает с множеством языков, порождаемых леволинейными грамматиками

**3) Збереження стану задач в реальному режимі**

В реальном режиме имеются программные и аппаратные прерывания. ПП инициируются командой INT, АП - внешними событиями, по отношению к выполняемой программе. Обычно АП инициируются аппаратурой ввода/вывода после завершения выполнения текущей операции.

Кроме того, некоторые прерывания зарезервированы для использования самим процессором - прерывания по ошибке деления, прерывания для пошаговой работы, немаскируемое прерывание и т.д.

Для обработки прерываний в реальном режиме процессор использует Таблицу Векторов Прерываний. Эта таблица располагается в самом начале оперативной памяти, т.е. её физический адрес - 00000. ТВП реального режима состоит из 256 элементов по 4 байта, таким образом её размер составляет 1 килобайт. Элементы таблицы - дальние указатели на процедуры обработки прерываний. Указатели состоят из 16-битового сегментного адреса процедуры обработки прерывания и 16-битового смещения. Причём смещение хранится по младшему адресу, а сегментный адрес - по старшему.

Когда происходит программное или аппаратное прерывание, текущее содержимое регистров CS, IP а также регистра флагов FLAGS записывается в стек программы (который, в свою очередь, адресуется регистровой парой SS:SP). Далее из таблицы векторов прерываний выбираются новые значения для CS и IP, при этом управление передаётся на процедуру обработки прерывания.

Перед входом в процедуру обработки прерывания принудительно сбрасываются флажки трассировки TF и разрешения прерываний IF. Поэтому если процедура прерывания сама должна быть прерываемой, то необходимо разрешить прерывания командой STI. В противном случае, до завершения процедуры обработки прерывания все прерывания будут запрещены.

Завершив обработку прерывания, процедура должна выдать команду IRET, по которой из стека будут извлечены значения для CS, IP, FLAGS и загружены в соответствующие регистры. Далее выполнение прерванной программы будет продолжено.

Что же касается аппаратных маскируемых прерываний, то в компьютере IBM AT и совместимых с ним существует всего шестнадцать таких прерываний, обозначаемых IRQ0-IRQ15. В реальном режиме для обработки прерываний IRQ0-IRQ7 используются вектора прерываний от 08h до 0Fh, а для IRQ8-IRQ15 - от 70h до 77h.

**Сохранения состояния в реальном режиме**

По идее (поскольку все организовано на прерываниях) сохраняются только регист указателя инструкции (на x86 это ip) и дальше уже обработчик прерываний сохраняет все регистрый которые собирается использовать