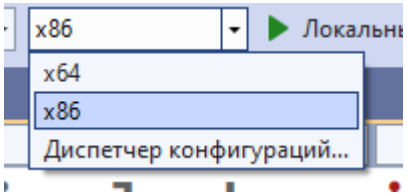
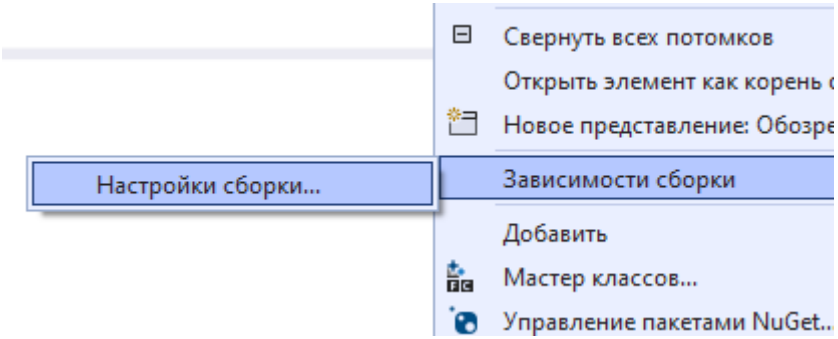
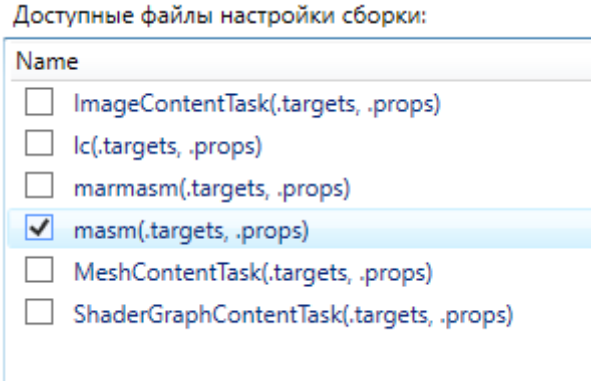
Математический сопроцессор

Запуск кода

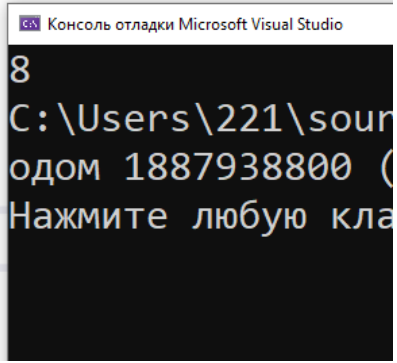

x86
Локальны
x64
x86
Диспетчер конфигураций...


Свернуть всех потомков
Открыть элемент как корень
Новое представление: Обозре
Настройки сборки...
Зависимости сборки
Добавить
Мастер классов...
Управление пакетами NuGet...


Доступные файлы настройки сборки:
Name
ImageContentTask(.targets, .props)
lc(.targets, .props)
marmasm(.targets, .props)
masm(.targets, .props)
MeshContentTask(.targets, .props)
ShaderGraphContentTask(.targets, .props)

Запуск кода

```
#include <iostream>
using namespace std;
int main()
{
    int a = 5, b = 3, sum;
    __asm {
        mov eax, a;
        mov ebx, b;
        add eax, ebx;
        mov sum, eax;
    }
    cout << sum;
}
```



Математический сопроцессор?

Важной частью архитектуры микропроцессоров Intel является наличие устройства для обработки числовых данных в формате с плавающей точкой, называемого математическим сопроцессором. Архитектура компьютеров на базе микропроцессоров вначале опиралась исключительно на целочисленную арифметику. С ростом мощи стали появляться устройства для обработки чисел с плавающей точкой.

Возможности

- полная поддержка стандартов IEEE-754 и 854 на арифметику с плавающей точкой. Эти стандарты описывают как форматы данных, с которыми должен работать сопроцессор, так и набор реализуемых им функций;
- поддержка численных алгоритмов для вычисления значений тригонометрических функций, логарифмов и т. п.;
- обработка десятичных чисел с точностью до 18 разрядов, что позволяет сопроцессору выполнять арифметические операции без округления над целыми десятичными числами со значениями до 1018;
- обработка вещественных чисел из диапазона $\pm 3.37 \times 10^{-4932} \ldots 1.18 \times 10^{+4932}$.

Математический сопроцессор

- использует стек (LIFO – Last In First Out)
- ST(0) – ST(7)
- Вершина стека — ST(0)
- Дно - ST(7)

Команды сопроцессора

- команды передачи данных в вещественном формате;
- команды передачи данных в целочисленном формате;
- команды передачи данных в двоично-десятичном формате.

Математический сопроцессор

Основными командами передачи данных являются

- команда FLD - загрузка данных в вершину стека сопроцессора
- команда FST - сохранение вершины стека сопроцессора в память

Команды передачи данных вещественного типа

Команда	Операнды	Пояснение	Описание
FLD	src	TOP_{SWR}-=1; ST(0)=src;	Загрузка операнда в вершину стека
FST	dst	dst=ST(0);	Сохранение вершины стека в память
FSTP	dst	dst=ST(0); TOP_{SWR}+=1;	Сохранение вершины стека в память с выталкиванием
FXCH	ST(i)	ST(0) ↔ ST(i)	Обмен значений ST(0) и ST(i)

Система команд

- все мнемонические обозначения начинаются с символа f (float);
- вторая буква мнемонического обозначения определяет тип операнда в памяти, с которым работает команда: — i — целое двоичное число; — b — целое десятичное число; — отсутствие буквы — вещественное число;
- последняя буква мнемонического обозначения команды p означает, что последним действием команды обязательно является извлечение операнда из стека;
- последняя или предпоследняя буква r (reversed) означает реверсивное следование операндов при выполнении команд вычитания и деления, так как для них важен порядок следования операндов

Команды загрузки констант

Команда	Пояснение	Описание
FLDZ	TOP_{SWR}-=1; ST(0)=0;	Загрузка 0
FLD1	TOP_{SWR}-=1; ST(0)=1;	Загрузка 1
FLDPI	TOP_{SWR}-=1; ST(0)=3.1415926535;	Загрузка π
FLDL2T	TOP_{SWR}-=1; ST(0)=3.3219280948;	Загрузка $\log_2 10$
FLDL2E	TOP_{SWR}-=1; ST(0)=1.4426950408;	Загрузка $\log_2 e$
FLDLG2	TOP_{SWR}-=1; ST(0)=0.3010299956;	Загрузка lg 2
FLDLN2	TOP_{SWR}-=1; ST(0)=0.6931471805;	Загрузка ln 2

Арифметические команды

Команда	Операнды	Пояснение	Описание
FADD	dst, src	dst = dst + src;	Сложение вещественное
FADDP	ST(i), ST(0)	ST(i) = ST(i) + ST(0); TOP_{SWR}+=1;	Сложение вещественное с выталкиванием
FSUB	dst, src	dst = dst — src;	Вычитание вещественное
FSUBP	ST(i), ST(0)	ST(i) = ST(i) — ST(0); TOP_{SWR}+=1;	Вычитание вещественное с выталкиванием
FSUBR	dst, src	dst = src — dst;	Вычитание вещественное реверсивное
FSUBRP	ST(i), ST(0)	ST(i) = ST(0) — ST(i); TOP_{SWR}+=1;	Вычитание вещественное реверсивное с выталкиванием
FMUL	dst, src	dst = dst * src;	Умножение вещественное
FMULP	ST(i), ST(0)	ST(i) = ST(i) * ST(0); TOP_{SWR}+=1;	Умножение вещественное с выталкиванием
FDIV	dst, src	dst = dst / src;	Деление вещественное
FDIVP	ST(i), ST(0)	ST(i) = ST(i) / ST(0); TOP_{SWR}+=1;	Деление вещественное с выталкиванием
FDIVR	dst, src	dst = src /dst;	Деление вещественное реверсивное
FDIVRP	ST(i), ST(0)	ST(i) = ST(0) / ST(i); TOP_{SWR}+=1;	Деление вещественное реверсивное с выталкиванием

Арифметические команды

Команда	Пояснение	Описание
FSQRT	ST(0) = √ST(0)	Вычисление квадратного корня
FABS	ST(0) = |ST(0)|	Вычисление модуля
FCHS	ST(0) = -ST(0)	Изменение знака
FXTRACT	temp = ST(0); ST(0)=порядок(temp); TOP-=1; ST(0)=мантисса(temp);	Выделение порядка и мантиссы
FSCALE	$ST(0) = ST(0) \cdot 2^{ST(1)}$	Масштабирование по степеням 2
FRNDINT	ST(0)=(ST(0))	Округление ST(0)
FPREM	ST(0)=ST(0)-Q*ST(1)	Частичный остаток от деления
FPREM1		

Арифметические команды

Команда	Пояснение	Описание
FSIN	ST(0) = sin(ST(0))	Вычисление синуса
FCOS	ST(0) = cos(ST(0))	Вычисление косинуса
FSINCOS	temp=ST(0); ST(0)=sin(temp); TOP-=1; ST(0)=cos(temp);	Вычисление синуса и косинуса
FPTAN	ST(0)=tg(ST(0)); TOP-=1; ST(0)=1.0;	Вычисление тангенса
FPATAN	ST(1)=atan(ST(1)/ST(0)); TOP+=1;	Вычисление арктангенса
F2XM1	$ST(0)=2^{ST(0)}-1$;	Вычисление выражения $y=2^x-1$
FYL2X	x=ST(0); y=ST(1); TOP+=1; $ST(0)=y*\log_2 x$;	Вычисление выражения $y*\log_2 x$
FYL2XP1	x=ST(0); y=ST(1); TOP+=1; $ST(0)=y*\log_2(x+1)$;	Вычисление выражения $y*\log_2(x+1)$

Пример

```
int main(){
  float a, b, res;//обязательно тип данных float
  cin >> a >> b;
  _asm{
    fld a;//st(0) = a
    fld b;//st(0) = b, st(1) = a
    fadd st(0), st(1); // st(0) = st(0) + st(1)
    fstp res; //res = st(0)
  }
  cout << " a + b = " << res;
}
```