Другие темы раздела

FASM Уроки Iczelion'a на FASM https://www.cyberforum.ru/ fasm/thread1240590.html
Уроки Iczelion'a на FASM Урок первый. MessageBox на FASM format PE GUI include 'win32ax.inc' ; import data in the same section invoke MessageBox,NULL,msgBoxText,msgBoxCaption,MB_OK ...

Побайтовый вывод файла FASM
Пытаюсь ввести в консоль файл в шестнадцатеричном виде, но происходит ошибка при выполнении. format PE console 4.0 include 'win32a.inc' xor ebx, ebx ; invoke CreateFile,\ ...

FASM Вывод адреса на консоль
Пытаюсь на консоль вывести адрес fin: invoke printf, не робит - как правильно надо? format PE console 4.0 entry start include 'win32a.inc' section '.data' data readable fin ...

FASM ГСЧ на макросах Всем привет. Понадобилось заюзать ГСЧ посредством макросов, чтобы каждый раз на стадии компиляции, использовалось уникальное значение. Учитывая семантику препроцессора (там чёрт ногу сломит),... https://www.cyberforum.ru/ fasm/thread1213146.html

FASM Создание окна на fasm https://www.cyberforum.ru/ fasm/thread1209394.html
Всем привет. Только что начал изучать ассемблер fasm. Возник первый вопрос: как создать окно? Прошу не просто дать мне код, а ещё объяснить что значит. Заранее благодарен

FASM Вызываем функции из clib (библиотека Си) в DOS
Вобщем, сбылась мечта идиота. Теперь, нежели писать свой ввод/вывод(особенно всегда напрягал ввод/вывод вещественных чисел на экран), можно воспользоваться стандартными ф-циями из библиотеки языка...

Организовать вычисления по формуле FASM
привет, всем активным участникам этого чудесного форума!!! помогите, пожалуйста, написать программу на Fasm Assembler. задание: Создать программу на языке Ассемблер, что позволяет организовать...

FASM Как сделать выход по ESC org 100h old dw 0 jmp start number dw 0 c dw 0 start: xor ax,ax mov es,ax cli https://www.cyberforum.ru/ fasm/thread1161834.html

FASM Получение CLSID image/png https://www.cyberforum.ru/fasm/ thread1160365.html
Всем ку! int GetEncoderClsid(const WCHAR* format, CLSID* pClsid) { UINT num = 0; // number of image encoders UINT size = 0; // size of the image encoder array in bytes...

FASM Вывод трех строк в один MessageBox Здравствуйте, помогите, пожалуйста, с такой проблемой: не могу вывести 3 строки (Год+Месяц+День) в один MessageBox Вот такой код: format PE GUI 4.0 entry start include 'win32ax.inc' include... https://www.cyberforum.ru/ fasm/thread1142589.html

MikI
Ушел с форума
13987 / 7000 / **813**
Регистрация: 11.11.2010
Сообщений: 12,592

21.08.2014, 09:06 **[TC]** **0**

Мануал по flat assembler

21.08.2014, 09:06. **Просмотров** 99777. **Ответов** 50
Метки (Все метки)

Ответ

2.1.16 SSE2 instructions

SSE2 (англ. Streaming SIMD Extensions 2, потоковое SIMD-расширение процессора) — это SIMD (англ. Single Instruction, Multiple Data, Одна инструкция — множество данных) набор инструкций, разработанный Intel и впервые представленный в процессорах серии Pentium 4. SSE2 расширяет набор инструкций SSE с целью полного вытеснения MMX. Набор SSE2 добавил 144 новые команды к SSE, в котором было только 70 команд.

SSE2 использует восемь 128-битных регистров (xmm0 до xmm7), включённых в архитектуру x86 с вводом расширения SSE, каждый из которых трактуется как 2 последовательных значения с плавающей точкой двойной точности.

SSE2 включает в себя набор инструкций, который производит операции со скалярными и упакованными типами данных.

SSE2 содержит инструкции для потоковой обработки целочисленных данных в тех же 128-битных xmm регистрах, что делает это расширение более предпочтительным для целочисленных вычислений, нежели использование набора инструкций MMX, появившегося гораздо раньше.

SSE2 включает в себя две части – расширение SSE и расширение MMX.

- расширение SSE работает с вещественными числами.
- расширение MMX работает с целыми.

В SSE2 регистры по сравнению с MMX удвоились (64 бита -> 128 битов). Т.к. скорость выполнения инструкций не изменилась, при оптимизации под SSE2 программа получает двукратный прирост производительности. Если программа уже была оптимизирована под MMX, то оптимизация под SSE2 дается сравнительно легко в силу сходности системы команд.

SSE2 включает в себя ряд команд управления кэшем, предназначенных для минимизации загрязнения кэша при обработке объёмных потоков данных.

SSE2 включает в себя сложные дополнения к командам преобразования чисел
Технология SSE2 повышает эффективность

The SSE2 extension introduces the operations on packed double precision floating point values, extends the syntax of MMX instructions, and adds also some new instructions.
movapd and movupd transfer a double quad word operand containing packed double precision values from source operand to destination operand. These instructions are analogous to movaps and movups and have the same rules for

operands.
movlpd moves double precision value between the memory and the low quad word of SSE register. movhpd moved double precision value between the memory and the high quad word of SSE register. These instructions are analogous to movlps and movhps and have the same rules for operands.
movmskpd transfers the most signicant bit of each of the two double precision values in the SSE register into low two bits of a general register. This instruction is analogous to **movmskps** and has the same rules for operands.
movsd transfers a double precision value between source and destination operand (only the low quad word is trasferred). At least one of the operands have to be a SSE register, the second one can be also a SSE register or 64-bit memory location.
Arithmetic operations on double precision values are: addpd, addsd, subpd, subsd, mulpd, mulsd, divpd, divsd, sqrtpd, sqrtsd, maxpd, maxsd, minpd, minsd, and they are analoguous to arithmetic operations on single precision values described in previous section. When the mnemonic ends with pd instead of ps, the operation is performed on packed two double precision values, but rules for operands are the same. When the mnemonic ends with sd instead of ss, the source operand can be a 64-bit memory location or a SSE register, the destination operand must be a SSE register and the operation is performed on double precision values, only low quad words of SSE registers are used in this case.
andpd, andnpd, orpd and xorpd perform the logical operations on packed double precision values. They are analoguous to SSE logical operations on single prevision values and have the same rules for operands.
cmppd compares packed double precision values and returns and returns a mask result into the destination operand. This instruction is analoguous to cmpps and has the same rules for operands. cmpsd performs the same operation on double precision values, only low quad word of destination register is aected, in this case source operand can be a 64-bit memory or SSE register. Variant with only two operands are obtained by attaching the condition mnemonic from table 2.3 to the cmp mnemonic and then attaching the pd or sd at the end.
comisd and **ucomisd** compare the double precision values and set the ZF, PF and CF flags to show the result. The destination operand must be a SSE register, the source operand can be a 128-bit memory location or SSE register.
shufpd moves any of the two double precision values from the destination operand into the low quad word of the destination operand, and any of the two values from the source operand into the high quad word of the destination operand. This instruction is analoguous to shufps and has the same rules for operand. Bit 0 of the third operand selects the value to be moved from the destination operand, bit 1 selects the value to be moved from the source operand, the rest of bits are reserved and must be zeroed.
unpckhpd performs an unpack of the high quad words from the source and destination operands, unpcklpd performs an unpack of the low quad words from the source and destination operands. They are analoguous to unpckhps and unpcklps, and have the same rules for operands.
cvtps2pd converts the packed two single precision floating point values to two packed double precision floating point values, the destination operand must be a SSE register, the source operand can be a 64-bit memory location or SSE register. cvtpd2ps converts the packed two double precision floating point values to packed two single precision floating point values, the destination operand must be a SSE register, the source operand can be a 128-bit memory location or SSE register. cvtss2sd converts the single precision floating point value to double precision floating point value, the destination operand must be a SSE register, the source operand can be a 32-bit memory location or SSE register. cvtsd2ss converts the double precision floating point value to single precision floating point value, the destination operand must be a SSE register, the source operand can be 64-bit memory location or SSE register.
cvtpi2pd converts packed two double word integers into the the packed double precision floating point values, the destination operand must be a SSE register, the source operand can be a 64-bit memory location or MMX register.
cvtsi2sd converts a double word integer into a double precision floating point value, the destination operand must be a SSE register, the source operand can be a 32-bit memory location or 32-bit general register. **cvtpd2pi** converts packed double precision floating point values into packed two double word integers, the destination operand should be a MMX register, the source operand can be a 128-bit memory location or SSE register. **cvttpd2pi** performs the similar operation, except that truncation is used to round a source values to integers, rules for operands are the same.
cvtsd2si converts a double precision floating point value into a double word integer, the destination operand should be a 32-bit general register, the source operand can be a 64-bit memory location or SSE register. **cvttsd2si** performs the similar operation, except that truncation is used to round a source value to integer, rules for operands are the same.

Вернуться к обсуждению:
Мануал по flat assembler

Следующий ответ

1

Programming
Эксперт
94731 / 64177 / **26122**
Регистрация: 12.04.2006
Сообщений: 116,782

21.08.2014, 09:06

Готовые ответы и решения:

Неофициальная разработка Flat assembler версии 2.0.0
Разработчик Flat assembler'a Tomasz Grysztar в одном из блогов сообщил о разработке новой...

Flat assembler ругается на PROC
Доброго времени суток. Есть программа, собственно вот что она делает: "На экране инициализировать...

Как подключить include к flat компилятору
Здравствуйте,как подключить include к flat компилятору?Требуется подключить include 'win32a.inc' к...

Flat Assembler
Со временем задачи стали нерешаемыми из-за ужасно медленной скорости. Уже давно хочу перейти на...

50