Содержание

1 Введение в NASM **2**
1.1 Что такое NASM ? 2
1.2 История 2

2 Запуск NASM **2**
2.1 Компиляция 2
2.2 Линковка 2

3 Введение в синтаксис **3**
3.1 Регистры 3
3.2 Резервирования данных 3
3.3 Резервирования инициализированных данных 4
3.4 Арифметические операции 4

1 Введение в NASM

1.1 Что такое NASM ?

NASM (Netwide Assembler) — свободный (LGPL и лицензия BSD) ассемблер для архитектуры Intel x86. Используется для написания 16-, 32- и 64-разрядных программ.

1.2 История

NASM был создан Саймоном Тэтхемом совместно с Юлианом Холлом и в настоящее время развивается небольшой командой разработчиков на SourceForge.net. Первоначально он был выпущен согласно его собственной лицензии, но позже эта лицензия была заменена на GNU LGPL после множества проблем, вызванных выбором лицензии. Начиная с версии 2.07 лицензия заменена на «упрощённую BSD» (BSD из 2 пунктов).

NASM может работать на платформах, отличных от x86, таких как SPARC и PowerPC, однако код он генерирует только для x86 и x86-64.

NASM успешно конкурирует со стандартным в Linux- и многих других UNIX-системах ассемблером gas. Считается, что качество документации у NASM выше, чем у gas. Кроме того, ассемблер gas по умолчанию использует AT&T-синтаксис, ориентированный на процессоры не от Intel, в то время как NASM использует вариант традиционного для x86-ассемблеров Intel-синтаксиса; Intel-синтаксис используется всеми ассемблерами для DOS/Windows, например, MASM, TASM, fasm.

2 Запуск NASM

2.1 Компиляция

Для ассемблирования файла вы должны ввести следующую команду:

nasm -f <format> <filename> [-o <output>]

Например,

nasm -f elf main.asm

2.2 Линковка

Для линковки объектников вы должны ввести следующую команду:

ld <filename> [-o <output>]

Например,

ld main.o -o nasm

3 Введение в синтаксис

3.1 Регистры

Регистр (цифровая техника) — последовательное или параллельное логическое устройство, используемое для хранения n-разрядных двоичных чисел и выполнения преобразований над ними.

Типы регистров

1. Общего назначения
2. Специальные регистр
3. Сегментные регистры

Примеры:
1. EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP - 32 bit (extended)
AX, BX, CX, DX, SI, DI, BP, SP - 16 bit
EAX - 32 bit
AX - 16 bit
AH - 8 bit and AL - 8 bit

EAX (acummulator) - хранит результат арифметической операци или является операндом
ECX (counter) - счетчик
EDX (data) - данные (дополняет EAX при арифметических операциях)
ESI and EDI (source index and destination index)
SP (stack pointer) - хранить указатель на вершину стека

2. EIP - счетчик команд (регистр в котором хранится адрес следующей команды в памяти) FLAGS - регистр флагов ZF (zero flag) - флаг нулевого результата при арифм. операции
CF (carry flag) - флаг переноса (результат арифм. операции не поместился в регистр)
SF (sign flag) - флаг знака
OF (overflow) - флаг переполнения

3.2 Резервирования данных

Резервирования неинициализированных данных происходит в секции .BSS.

resb - 1 byte
resw - 2 byte
resd - 4 byte

Например,
string resb 20 ; массив 1 байтовых ячеек
count resw 256 ; массив 2 байтовых ячеек
x resd 1 ; одно 4 байтовое целое число

3.3 Резервирования инициализированных данных

Резервирования нициализированных данных происходит в секции .DATA.

db - 1 byte
dw - 2 byte
dd - 4 byte

3.4 Арифметические операции

Взять значение из регистра eax и добавить к нему значение ebx
Пример ассемблерного кода,

add eax, ebx

Взять 4 байтное число и отнять от него значение регистра ecx
Пример ассемблерного кода,

sub dword [x], ecx