

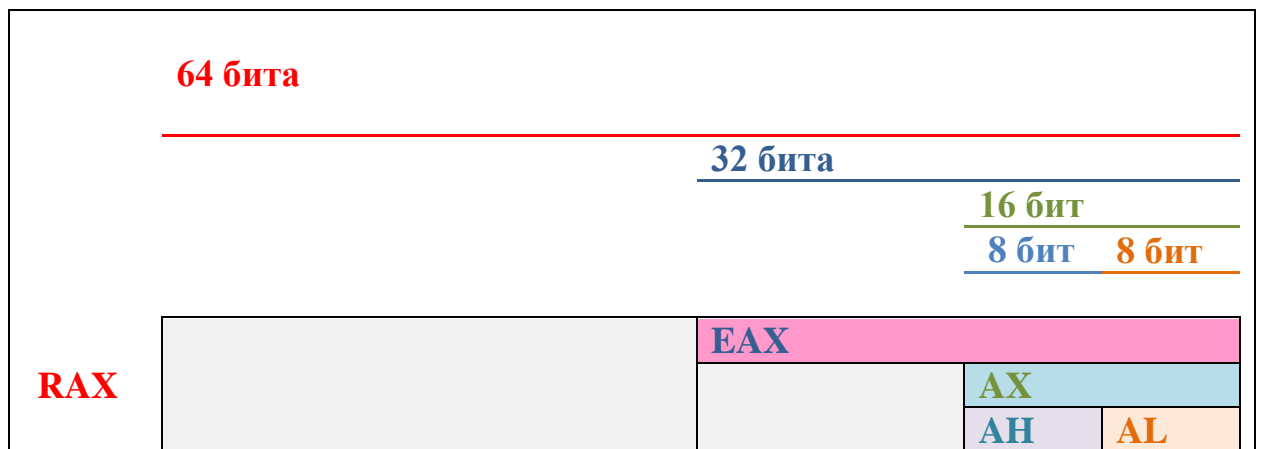
## Особенности программирования на MASM для x64.

Основные возможности архитектуры x86-64:

- 64-битное адресное пространство;
- расширенный набор регистров;
- привычный для разработчиков набор команд;
- возможность запуска старых 32-битных приложений в 64-битной операционной системе;
- возможность использования 32-битных операционных систем.

Список основных 16 целочисленных 64-разрядных регистров общего назначения:

- RAX, RBX, RCX, RDX, RSI, RDI, RSP, RBP
- R8, R9, R10, R11, R12, R13, R14, R15 ← (new)



Имена регистров общего назначения в процессорах x64 начинаются с буквы R, например RAX, RBX и др. Это развитие старой схемы именования на основе префикса E для 32-битных регистров x86.

Новые регистры просто пронумерованы от R8 до R15.

Для обращения к младшим 8-, 16- и 32-битам новых регистров используются суффиксы b, w и d соответственно.

Например,

R9 – 64-разрядный регистр;

R9b – его младший байт (по аналогии с AL);

R9w – младшее слово (по аналогии с AX в EAX);

R9d – младшее двойное слово (по аналогии с EAX).

SSE 128 бит		64 бита
XMM0	RAX	EAX
XMM1	RBX	EBX
XMM2	RCX	ECX
XMM3	RDX	EDX
XMM4	RSI	ESI
XMM5	RDI	EDI
XMM6	RBP	EBP
XMM7	RSP	ESP
XMM8	R8	
XMM9	R9	
XMM10	R10	
XMM11	R11	
XMM12	R12	
XMM13	R13	
XMM14	R14	
XMM15	R15	

#### Новые возможности:

- использовать адресацию относительно RIP регистра, например «MOV AL, [RIP]»;
- операции с плавающей запятой выполняются с помощью 16 регистров XMM;
- для **относительной** адресации используется регистр RIP (всегда указывает на инструкцию, следующую по порядку);
- можно обращаться к младшим байтам **индексных** регистров (bpl, spl, dil, sil).

#### Ограничения.

Инструкция не может ссылаться одновременно на младший байт старых регистров (ah, bh, dh, ch) и младший байт новых регистров.

"mov ah, dl" — допустимо, а

"mov ah, r8b" — не допустимо.

Операции с 32 битными операндами **обнуляют старшие 4 байта результата**. То есть "add eax, ebx" обнулит старшую часть регистра eax.

**x86-64 fast calling conversion** (соглашение о быстрой передаче параметров для x86-64).

- Первые четыре целочисленных аргумента (слева направо) передаются в 64-битных регистрах RCX, RDX, R8 и R9.
- Остальные целочисленные аргументы передаются через стек (справа налево). Указатель `this` всегда хранится в регистре RCX.
- Для каждого аргумента, даже переданного через регистр, вызывающая функция **обязана** резервировать для него место в стеке, уменьшая значение регистра RSP (регистр указателя стека).
- Параметры с плавающей точкой передаются через регистры XMM0-XMM3, если их более четырех, последующие — через стек потока.
- Если параметр занимает больше 8 байт, он передаётся через указатель.
- Стек освобождается вызывающей стороной.

**Выравнивание стека:**

Вызывающая функция отвечает за выделение пространства для параметров вызываемой функции и всегда должна выделять **достаточное** для 4 параметров, даже если вызываемая функция не содержит такого количества параметров.

Команда «`call`» помещает в стек 8-байтовое возвращаемое значение.

Перед вызовом функции значение в RSP должно быть кратно 16.

Пример вызова функции с пятью аргументами `func(1,2,3,4,5)`:

```
mov    dword ptr [rsp+20h], 5      ; пятый слева аргумент в стек
mov    r9d, 4                     ; четвертый слева аргумент в r9d
mov    r8d, 3                     ; третий слева аргумент в r8d
mov    edx, 2                     ; второй слева аргумент в edx
mov    ecx, 1                     ; первый слева аргумент в ecx

call   func                       ; вызов функции
```

Смещение пятого аргумента относительно верхушки стека равно 20h. Эти байты «резервируются» для первых четырех аргументов, переданных через регистры. Эти ячейки содержат неинициализированный мусор и называются «spill».

```
sub    rsp,8*5
```

В этом примере резервируется место в стеке для четырех параметров и происходит выравнивание стека на границу 16 байт (32 байта для параметров + 8 байт для выравнивания = 40 байт).

В приведенных ниже примерах используется 64-битная версия MASM ML64.EXE, свободно доступная в Windows Platform SDK.

c:\masm32\bin64\ml64.exe \HELP

```
D:\Ade1\LPLab\Asm64\Asm64>c:\masm32\bin64\ml64.exe /HELP
Microsoft (R) Macro Assembler (x64) Version 14.00.24210.0
Copyright (C) Microsoft Corporation. All rights reserved.

    ML64 [ /options ] filelist [ /link linkoptions ]

/B1<linker> Use alternate linker                /Sf Generate first pass listing
/c Assemble without linking                    /Sl<width> Set line width
/Cp Preserve case of user identifiers           /Sn Suppress symbol-table listing
/Cx Preserve case in publics, externs         /Sp<length> Set page length
/D<name>[=text] Define text macro              /Ss<string> Set subtitle
/EP Output preprocessed listing to stdout      /St<string> Set title
/F <hex> Set stack size (bytes)                /Sx List false conditionals
/Fe<file> Name executable                     /Ta<file> Assemble non-.ASM file
/Fl[file] Generate listing                    /w Same as /W0 /WX
/Fm[file] Generate map                       /WX Treat warnings as errors
/Fo<file> Name object file                   /W<number> Set warning level
/Fr[file] Generate limited browser info       /X Ignore INCLUDE environment path
/FR[file] Generate full browser info          /Zd Add line number debug info
/I<name> Add include path                    /Zf Make all symbols public
/link <linker options and libraries>          /Zi Add symbolic debug info
/nologo Suppress copyright message            /Zp[n] Set structure alignment
/Sa Maximize source listing                  /Zs Perform syntax check only
/ZH:SHA_256 Use SHA256 for checksum
        in debug info (experimental)
/Gy[-] separate functions for linker
/errorReport:<option> Report internal assembler errors to Microsoft
        none - do not send report
        prompt - prompt to immediately send report
        queue - at next admin logon, prompt to send report
        send - send report automatically
```

```

;Conf.asm
includelib c:/masm64/lib/kernel32.lib
includelib c:/masm64/lib/user32.lib
extrn MessageBoxA : PROC ; внешняя API - функция
extrn ExitProcess : PROC ; внешняя API - функция

; секция данных с атрибутами по умолчанию (чтение и запись)
.data
mytit db 'Hi 64-bit!', 0
mymsg db 'Hello World!', 0

; секция кода с атрибутами по умолчанию (чтение и исполнение)
.code
Main PROC
sub rsp, 28h ; выравнивание стека: 8*4+8
mov r9d, 0 ; uType = MB_OK
lea r8, mytit ; заголовок окна
lea rdx, mymsg ; текст
mov rcx, 0 ; hWnd = HWND_DESKTOP
call MessageBoxA
mov ecx, eax ; uExitCode = MessageBox(...)
call ExitProcess
Main ENDP
End

```

Ассемблируем:

```
ml64.exe Conf.asm /link /subsystem:windows /entry:Main
```

```

D:\Ade1\LPLab\Conf\Conf>c:\masm32\bin64\ml64.exe Conf.asm /link /subsystem:windows /entry:Main
Microsoft (R) Macro Assembler (x64) Version 14.00.24210.0
Copyright (C) Microsoft Corporation. All rights reserved.

Assembling: Conf.asm
Microsoft (R) Incremental Linker Version 14.00.24215.1
Copyright (C) Microsoft Corporation. All rights reserved.

/OUT:Conf.exe
Conf.obj
/subsystem:windows
/entry:Main

```

Выполняем:

```
D:\Ade1\LPLab\Conf\Conf>Conf.exe
```

```
D:\Ade1\LPLab\Conf\Conf>
```

