

strchr.com

Perfectionistic and minimalistic programming.

Featured pages

[Discussion: the first language](#)

Which programming language to learn first?

[Hash functions:](#)

[An empirical comparison](#)

Benchmark program for hash tables and comparison of 15 popular hash functions.

[Recommended books and sites](#)

The minimal reading list to become a good programmer.

[Software interface design tips](#)

How to design easy-to-use interfaces between modules of your program.

[Implementing strcmp, strlen, and strstr using SSE 4.2 instructions](#)

Using new Intel Core i7 instructions to speed up string manipulation.

[Table-driven approach](#)

How to make you code shorter and easier to maintain by using arrays.

[x86 Machine Code Statistics](#)

Which instructions and addressing modes are used most often. What is the average

Win32 Assembly Cheat Sheet

The cheat sheet is intended for 32-bit Windows programming with [FASM](#). One A4 page contains almost all general-purpose x86 instructions (except FPU, MMX and SSE instructions).

What is included

You will find various kinds of moves (MOV, CMOV, XCHG), arithmetical (ADD, SUB, MUL, DIV) and logical (AND, OR, XOR, NOT) instructions here. Several charts illustrate shifts (SHL/SHR, ROL/ROR, RCL/RCR) and stack frames. Code samples for typical high-level language constructs (if conditions, while and for loops, switches, function calls) are shown. Also included are quick references for RDTSC and CPUID instructions, description of string operations such as REP MOVSB, some code patterns for branchless conditions, a list of registers that should be saved in functions, and a lot of other useful stuff.

The idea is to put all reference information about x86 assembly language on the one page. Some rarely-used instructions such as LDS, BOUNDS or AAA are skipped.

Notation

The cheat sheet use common notation for operands: *reg* means register, *[mem]* means memory location, and *imm* is an immediate operand. Also, *x*, *y*, and *z* denote the first, the second, and the third operand. Instruction mnemonics are written in capital letters to make them easier to find when you are skipping through the cheat sheet.

Example

Multiplication and division	
$ax = al * x$	MUL $\frac{reg}{reg}$
$dx:ax = ax * x$	IMUL $\frac{[mem]}{[mem]}$
$edx:eax = eax * x$	
	remainder:
$al = ax / x$	DIV $\frac{reg}{reg}$
$ax = dx:ax / x$	IDIV $\frac{[mem]}{[mem]}$
$eax = edx:eax / x$	
	ah ax edx
$x = x * y$	IMUL $\frac{reg, reg}{reg, [mem]}$
$x = y * z$	IMUL $\frac{reg, reg, imm}{reg, [mem], imm}$

For example, let's look at multiplication and division section. There are instructions for signed (IMUL) and unsigned (MUL) multiplication. Both instructions take one operand, which may be register (*reg*) or memory

Abstract:

One-page reference for Win32 assembly language programming.

Created 6 years ago by *Peter Kankowski*

Last changed 2 years ago
Filed under *Assembly language and machine code*



instruction length.

(*mem*). There are three possible cases:

Recent comments

westggv@qq.com:

do you know RDTSC of ARM-cortex cpu ? can show some codes for ARM RDTSC ?

[slavne](#):

thank you Peter, nice and very practical article

[Willy Carfield](#):

I have had to maintain servers with some Python scripts for 10+ years...

[ntysdd](#):

when you #define something you should always remember to add parentheses.

[Reini Urban](#):

This is wrong: "it's better to use a switch instead of the jump table, because the switch is portable and shorter..."

- If operand size is one byte, MUL or IMUL multiplies it by *al* and stores the result in *ax*
- If operand size is a word, MUL or IMUL multiplies it by *ax* and stores the high-order word of the result in *dx* and the low-order word in *ax*.
- If operand size is a double word, MUL or IMUL multiplies it by *eax* and stores the high-order dword in *edx* and the low-order dword in *eax*.

There are also two-operand and three-operand forms of IMUL shown on the figure above.

Other features of assembly language are described in a similar way.

Download

The cheat sheet is designed for A4 page size; if you print it on US Letter paper, you will get large margins. You can print the cheat sheet and put it on your table to look for some instructions when you forget them.

[Download Win32 Assembly Cheat Sheet \(PNG picture, 713 Kb\)](#)

[Serbo-Croatian translation](#) of this article by [WHG Team](#).

About the author



Peter Kankowski

Peter lives in Siberia, the land of sleeping sun, beautiful mountains, and infinitely deep snow. He likes to program in C with a bit of C++, also in x86 assembly language, Python, and PHP (on Windows platform). He can be reached at kankowski@narod.ru.

24 comments

Ten recent comments are shown below. [Show all comments](#)

ken, 6 years ago

excellent cheat sheet. thank you very much!

nairam, 6 years ago

Hello. All is A4 format:

x86 registers:

<http://www.nairam.sk/pc01.pdf>

x86 instructions:

<http://www.nairam.sk/pc03.pdf>

<http://www.nairam.sk/pc05.pdf>

Peter Kankowski, 6 years ago

Thank you! I cannot read Slovak, but your charts seem to be good for learning assembly language.

Itamar, 3 years ago

Hi, I'd like to learn "assembly" at home.

I can program in java but I don't know anything about "assembly". I say ANYTHING.

Where do I start?

Peter Kankowski, 3 years ago

Please start by googling for "assembly tutorial" (try also in your native language). [Paul Carter's tutorial](#) looks good at the first glance. You can also download [FASM](#) and read [its docs](#).

Itamar, 3 years ago

Thank You Mr Peter.

It helped me a lot.

Mater Liu, 3 years ago

great Peter, thank you.

regards

mater

Marco, 2 years ago

Thank you, by this very helpful site :)

Kind regards,

Marco

(from Portugal)

hello, 1 year ago

<p>thank you</p>

cube, 10 months ago

student: example count cube in assembler tasm x86

acegimo@hotmail.it

Your name:

Comment:

Enter verification code:



Submit

(cc) strchr.com, 2006—2014. Some rights reserved.