

LEKSYKON
KIESZONKOWY

ASEMBLER

Poznaj Asemblera i dołącz
do elity programistów

Dowiedz się,
jak jest zbudowany
i jak działa procesor

Poznaj język Asembler
i narzędzia umożliwiające
tworzenie w nim programów

Nauucz się pisać
wydajny kod działający
w systemach DOS i Windows

DAWID FARBANIEC

Helion



Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Michał Mrowiec

Projekt okładki: Maciek Pasek

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie?asemlk>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody źródłowe wybranych przykładów dostępne są pod adresem:
<ftp://ftp.helion.pl/przyklady/asemlk.zip>

ISBN: 978-83-246-4347-9

Copyright © Helion 2012

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

1. Podstawowe informacje	7
1.1. Słowem wstępu	7
1.2. Architektura x86 — podstawowe informacje	10
1.3. Budowa programu dla podsystemu DOS (16-bitowego)	13
1.4. Budowa programu dla systemu Windows (32-bitowego)	15
2. Podstawowe instrukcje procesora	18
2.1. Instrukcja kopiowania i instrukcje arytmetyczne	18
2.2. Instrukcje logiczne i przesunięć bitowych	20
2.3. Instrukcje wywołania procedury i powrotu	26
2.4. Instrukcja porównania i instrukcje skoku	29
2.5. Instrukcje do operacji na łańcuchach znaków	30
3. Makroinstrukcje	35
3.1. Makroinstrukcje kontroli przepływu	35
3.2. Makroinstrukcje do tworzenia pętli	36
4. Praca z danymi	39
4.1. Adresowanie i wskaźniki	39
4.2. Zmienne i stałe	41
4.3. Praca ze strukturami	42
5. Programowanie w systemie Windows	45
5.1. Konsola w systemie Windows	45
5.2. Proste okno dialogowe	47
5.3. Odczyt myszki i klawiatury	51
5.4. Operacje na plikach i alokacja bloków pamięci	52
5.5. Tworzenie menu	55
5.6. Kontrolka przycisku (Button)	56
5.7. Kontrolka pola tekstowego (Edit)	57
5.8. Kontrolka wielokrotnego wyboru (CheckBox)	60

5.9. Kontrolka pojedynczego wyboru (RadioButton)	62
5.10. Kontrolka listy rozwijanej (ComboBox)	62
5.11. Kontrolka listy (ListBox)	64
5.12. Kontrolka paska postępu (ProgressBar)	65
5.13. Kontrolka widoku drzewa (TreeView)	66
5.14. Kontrolka widoku listy (ListView)	68
5.15. Kontrolka suwaka (TrackBar)	70
5.16. Kontrolka podpowiedzi (Tooltip)	71
5.17. Kontrolka do wprowadzania adresu IP (IPAddress)	78
5.18. Kontrolka tekstu o bogatym formatowaniu (RichEdit)	83
5.19. Kontrolki w nowym stylu (pliki .manifest)	85
5.20. Podpięcia do systemu Windows (Hooks)	87
5.21. Ekran powitalny (Splashscreen)	89
5.22. Ikona w zasobniku systemowym (Tray)	91
5.23. Podmiana procedury obsługi okna	93
5.24. Programowanie aplikacji wielowątkowych	95
5.25. Tworzenie bibliotek DLL	97
5.26. Pliki odwzorowane w pamięci	99
5.27. Pobieranie adresu IP lokalnego komputera	99
6. Dodatkowe zestawy instrukcji	102
6.1. Korzystanie z instrukcji koprocatora (FPU)	102
6.2. Korzystanie z rozszerzeń MMX i SSE	110
6.3. Nowe rozszerzenie — Advanced Vector Extensions (AVX)	118
7. Asembler 64-bitowy (x86-64)	120
7.1. Wstęp do Asemblera dla architektury 64-bitowej	120
8. Opis rozkazów procesorów z rodziny 80x86	125
8.1. Instrukcje na literę A	125
8.2. Instrukcje na literę B	128
8.3. Instrukcje na literę C	132
8.4. Instrukcje na literę D	137
8.5. Instrukcje na literę E	139
8.6. Instrukcje na literę H	139
8.7. Instrukcje na literę I	140
8.8. Instrukcje na literę J	143
8.9. Instrukcje na literę L	144
8.10. Instrukcje na literę M	148
8.11. Instrukcje na literę N	151

8.12. Instrukcje na literę O	152
8.13. Instrukcje na literę P	154
8.14. Instrukcje na literę R	156
8.15. Instrukcje na literę S	160
8.16. Instrukcje na literę T	168
8.17. Instrukcje na literę V	169
8.18. Instrukcje na literę W	169
8.19. Instrukcje na literę X	170
9. Dyrektywy asemblera MASM	172
9.1. Etykiety kodu	172
9.2. Warunkowa kontrola przepływu	173
9.3. Alokaacja danych	175
9.4. Przyrównania	177
9.5. Makra	177
9.6. Procedury	178
9.7. Rodzaj zestawu instrukcji procesora	179
9.8. Bloki powtórzeń	182
9.9. Zakres	183
9.10. Segmenty	184
9.11. Uproszczone segmenty	185
9.12. Łańcuchy znaków	187
9.13. Struktury i rekordy	188
9.14. Różne	189
Dodatki	192
A. Tablica kodów ASCII	192
B. Potęgi liczby dwa	193
C. Lista opkodów procesora 80x86	196
Skorowidz	202

Rozdział 4. Praca z danymi

W tym rozdziale nauczysz się, jak pracować z danymi, takimi jak na przykład zmienne czy struktury. Dowiesz się też, czym jest adres danych w pamięci i jak go pobierać.

4.1. Adresowanie i wskaźniki

W Asemblerze bardzo ważne jest to, aby wiedzieć, czym różni się adres zmiennej (OFFSET) od jej wartości. Adres to miejsce, gdzie znajduje się zmienna, natomiast wartością są dane umieszczone pod tym adresem.

Operator OFFSET

Operator OFFSET służy do pobierania adresu zmiennej.

Oto przykład jego użycia:

```
.386
.model    flat, stdcall
option    casemap: none

include    windows.inc
include    kernel32.inc
include    user32.inc

includelib user32.lib
includelib kernel32.lib

.data
zmienna db "Witaj!", 0

.code
start:
    mov eax, offset zmienna ; do rejestru EAX skopiuj adres zmiennej
    invoke MessageBox, NULL, eax, NULL, NULL ; wyświetl komunikat
    invoke ExitProcess, NULL ; koniec programu
end start
```

Operatorem OFFSET używamy również, gdy pobieramy adres zmiennej, odkładając ją na stos przed wywołaniem procedury, na przykład:

```
push NULL
push NULL
push offset zmienna ; pobieramy adres zmiennej i odkładamy na stos
push NULL
call MessageBox
```

Warto zaznaczyć, że operator OFFSET nie działa na zmiennych lokalnych.

Zmienne lokalne

Zmienne lokalne możemy zadeklarować na przykład wewnątrz jakiejś procedury. Aby to zrobić, podajemy najpierw słowo LOCAL, dalej nazwę zmiennej, a po dwukropku jej rodzaj.

Na przykład:

```
myProc proc
    LOCAL zmienna1: DWORD ; deklaracja zmiennej lokalnej o rozmiarze podwójnego słowa
                           ; jakiś kod
    Ret ; powrót
myProc EndP
```

Operator ADDR

Operatorem ADDR używamy najczęściej przy przekazywaniu argumentów do funkcji wywoływanej za pomocą składni INVOKE, na przykład:

```
invoke MessageBox, NULL, addr MsgText, addr MsgCaption, NULL
```

Dereferencja (operator [])

Posłużymy się tutaj prostym przykładem:

```
mov eax, lpvar    ; skopiuj adres do EAX
mov eax, [eax]    ; dereferencja
mov nuvar, eax    ; skopiuj EAX do nowej zmiennej
```

Nawiasy kwadratowe służą do wydobywania wartości spod adresu w EAX. Operacja ta jest nazywana dereferencją.

Instrukcja LEA

Instrukcja ta jest skrótem angielskiego wyrażenia: *Load Effective Address*. Służy ona do pobierania adresu i kopiowania go do pierwszego operandu, na przykład:

```
lea eax, zmienna
invoke MessageBox, NULL, eax, NULL, NULL
```

4.2. Zmienne i stałe

W języku Asembler prócz operacji na rejestrach możemy definiować zmienne oraz stałe, by wykonywać na nich różne operacje.

Stałe

Stałe umieszczamy w sekcji `.CONST`.

Posłużę się tutaj prostym przykładem:

```
.const ; stałe
jeden equ 1 ; definicja stałej
osiem equ 2*4 ; wartość wyliczona
```

Zmienne

Zmienne możemy podzielić na zainicjowane jakąś wartością oraz niezainicjowane. Zmienne zainicjowane deklarujemy z wartością początkową z prawej strony, niezainicjowane natomiast mają z prawej strony znak zapytania. Zmienne zainicjowane deklarujemy w sekcji `.DATA`, a niezainicjowane w — `.DATA?`. Obie te sekcje występują przed sekcją kodu (`.CODE`).

Zmienne o rozmiarze jednego bajtu/ciągu bajtów

Zmienne te deklarujemy dyrektywą `BYTE` (lub krótko: `DB`; ang. *Define Byte*).

```
.data ; dane zainicjowane wartością podaną z prawej strony
b1 BYTE 0 ; jeden bajt
b2 db 0 ; również jeden bajt
b3 db 100 dup(0) ; sto bajtów
b4 db "asd", 0 ; cztery bajty (ciąg znaków ASCII zakończony zerem)
b5 SBYTE 0 ; jeden bajt ze znakiem
t1 dt 0 ; dziesięć bajtów
```

Zmienne o rozmiarze jednego słowa

Zmienne te deklarujemy przy użyciu dyrektywy `WORD` (lub krótko: `DW`; ang. *Define Word*).

```
w1 word 0 ; słowo
w2 dw 0 ; również słowo
```

Zmienne o rozmiarze podwójnego słowa

Zmienne te deklarujemy za pomocą dyrektywy `DWORD` (lub krótko: `DD`; ang. *Define Double Word*).

```
d1 dword 0 ; podwójne słowo
d2 dd 0 ; również podwójne słowo
d3 sdword 0 ; podwójne słowo ze znakiem
```

Zmienne o rozmiarze poczwórnego słowa

Zmienne te deklarujemy za pomocą dyrektywy QWORD (lub krótko: DQ; ang. *Define Quad Word*).

```
q1 qword 0 ; poczwórne słowo
q2 dq 0 ; również poczwórne słowo
q3 sqword 0 ; poczwórne słowo ze znakiem
```

Zmienne o rozmiarze 6 bajtów

Zmienne te deklarujemy przy użyciu dyrektywy FWORD (lub krótko: DF; ang. *Define Fword*).

```
f1 fword 0 ; zmienna 6-bajtowa
f2 df 0 ; również zmienna 6-bajtowa
```

Zmienne używane w instrukcjach MMX i SSE

Zmienne te deklarujemy za pomocą dyrektyw: MMWORD i XMMWORD. Pierwsza ma 64 bity, a druga — 128.

```
m1 mmword 0.0 ; 64-bitowa zmienna używana w instrukcjach MMX i SSE
x1 xmmword 0.0 ; 128-bitowa zmienna używana w instrukcjach MMX i SSE
```

Zmienne do przechowywania liczb rzeczywistych

Zmienne te deklarujemy przy użyciu dyrektyw: REAL4, REAL8 i REAL10.

```
n1 real4 0.0 ; liczba zmiennoprzecinkowa pojedynczej precyzji (4 bajty)
n2 real8 0.0 ; liczba zmiennoprzecinkowa podwójnej precyzji (8 bajtów)
n3 real10 0.0 ; liczba zmiennoprzecinkowa 10-bajtowa
```

4.3. Praca ze strukturami

W poprzednim rozdziale była mowa o zmiennych i stałych. Teraz opiszę typ danych, jakim jest struktura.

Deklaracja przykładowej struktury dla assemblera MASM wygląda następująco (deklarację tę umieszczamy PRZED SEKCJĄ .DATA):

```
Rect STRUCT
    left    dword ?
    top     dword ?
    right   dword ?
    bottom  dword ?
Rect ENDS
```

Najpierw pojawia się nazwa struktury (Rect), po niej słowo STRUCT, a niżej są umieszczone pola struktury. Każde z nich jest podwójnym słowem (DWORD), niezainicjowanym żadną wartością (znak ?). Na końcu mamy znów nazwę struktury i słowo kończące deklarację ENDS.

Teraz utwórzmy zmienną typu Rect. Robimy to w ten sposób:

```
.data  
Rct Rect <>
```

Do pól struktury odwołujemy się, pisząc nazwę struktury i po kropce jej pole, na przykład:

```
mov Rct.left, 12  
mov Rct.top, 24  
mov Rct.right, 31  
mov Rct.bottom, 10
```

Jeżeli jakaś funkcja wymaga od nas adresu struktury jako argumentu, używamy operatora ADDR, na przykład:

```
invoke Func1, param1, param2, addr Rct
```

Możemy również tworzyć struktury zawierające w sobie inne struktury, na przykład:

```
Rect STRUCT  
    left    dword ?  
    top     dword ?  
    right   dword ?  
    bottom  dword ?  
Rect ENDS  
  
Point STRUCT  
    x dword ?  
    y dword ?  
Point ENDS  
  
MyStruct STRUCT  
    item1 Rect <> ; struktura Rect  
    item2 Point <> ; struktura Point  
MyStruct ENDS  
  
.data  
Struct1 MyStruct <>
```

Wtedy do pól odwołujemy się w następujący sposób:

```
mov Struct1.item1.left, 10
```

Do pól struktury możemy się także odnosić poprzez adres w rejestrze z dodanym przesunięciem, na przykład:

```
mov eax, lpRct ; adres struktury do EAX
mov [eax],     DWORD PTR 10
mov [eax+4],   DWORD PTR 12
mov [eax+8],   DWORD PTR 14
mov [eax+12],  DWORD PTR 16
```

Można to również zrobić przy użyciu dyrektywy ASSUME:

```
ASSUME eax:PTR RECT
    mov eax, lpRct
    mov [eax].left, 10
    mov [eax].top, 12
    mov [eax].right, 14
    mov [eax].bottom, 16
ASSUME eax: nothing
```

Skorowidz

386, 179
386P, 180
387, 180
486, 180
486P, 180
586, 180
586P, 181
686, 181
686P, 181

A

AAA, 125
AAD, 125
AAM, 126
AAS, 126
AC, 12
ADC, 126
ADD, 19, 127
ADDR, 40
adres IP, *Patrz* IPAddress
 lokalnego komputera, 99
adres zmiennej, 39
adresowanie x64, 124
Advanced Vector
 Extensions, *Patrz* AVX
AF, 11
AH, 10
akumulator, 10
AL, 10
ALIAS, 189
ALIGN, 172
Alignment Check, 12
alokacja
 bloków pamięci, 52
 danych, 175
ALPHA, 184
alternatywa, 21
alternatywa wykluczająca,
 21
AMD64 Assembly
 Debugger, 124
AND, 20, 127
aplikacje wielowątkowe, 95
architektura
 64-bitowa, *Patrz* x64
 x86, 10
ARPL, 128

ASCII, 192
assembler, 7
Assembler, 7
 64-bitowy, 120
ASSUME, 44, 184
Auxillary Flag, 11
AVX, 118
 instrukcje, 119
AX, 10

B

bajt, 9
biblioteki DLL, 97
bit, 9
bloki
 pamięci, 52
 powtórzeń, 182
BOUND, 128
BREAK, 36, 173
BS_3STATE, 60
BS_AUTO3STATE, 60
BS_FLAT, 57
BSF, 129
BSR, 129
BST_CHECKED, 60, 61
BST_INDETERMINATE, 60,
 61
BST_UNCHECKED, 60, 61
BSWAP, 130
BT, 130
BTC, 130
BTR, 131
BTS, 131
budowa programu
 DOS, 13, 14
 Windows, 15
Button, 56
BYTE, 41, 175

C

CALL, 17, 26, 132
CallNextHookEx, 88
 hhk, 89
 lParam, 89
 nCode, 89
 wParam, 89

CallWindowProc, 76, 95
Carry Flag, 11, 22
CARRY?, 36
CATSTR, 187
CB_ADDSTRING, 63
CB_DELETESTRING, 63
CB_GETCOUNT, 63
CB_GETCURRESEL, 63
CB_SETCURRESEL, 63
CBW, 133
CDQ, 133
CF, 11, 22
CHARFORMAT, 84
CHARRANGE, 83
CheckBox, 60
 przykładowy program,
 61
 sprawdzanie stanu, 60
 ustawianie stanu, 61
CheckDlgButton, 61
 hDlg, 61
 nIDButton, 61
 uCheck, 61
CLC, 134
CLD, 134
CLI, 134
CloseHandle, 99
CLTS, 135
CMC, 135
CMP, 29, 135
CMPS, 31, 136
CODE, 41, 50, 185
Code Segment, 11
ComboBox, 62
 komunikaty, 63
 przykładowy program,
 63
COMM, 183
COMMENT, 189
CONST, 41, 186
CONTINUE, 173
CPL, 12
CreateFile, 99
CreateFileMapping, 99
CreateThread, 95
 dwCreationFlags, 96
 dwStackSize, 96
 lpParameter, 96

lpStartAddress, 96
lpThreadAttributes, 95
lpThreadId, 96
CreateWindowEx, 72
CS, 11
Current Priority Level, 12
CW_USEDEFAULT, 72
CWD, 136
CWDE, 137

D

DAA, 137
DAS, 137
DATA, 41, 186
Data Segment, 11
DATA?, 41, 186
DB, 41, 175
DD, 41
debuggery, 7
 aplikacji x64, 124
Debugging Tools for
 Windows 64-bit Version,
 124
DEC, 20, 138
dekrementacja, 20
dereferencja, 40
DF, 12, 30, 42
DI, 10
DialogBoxParam, 50
Direction Flag, 12, 30
DIV, 19, 138
DLL, 97
dodawanie, 19
DOS
 budowa programu, 13,
 14
 szablony aplikacji, 14
DOSSEG, 184
double word, 9
DQ, 42
DS, 11
DW, 41
DWORD, 41, 175
dyrektywy asemblera
 MASM, 172
dzielenie
 bez znaku, 19
 ze znakiem, 20

E

EAX, 10
EBP, 10
EBX, 10

ECHO, 189
ECX, 10
EDI, 10
Edit, 57
 pobieranie liczby, 58
 pobieranie tekstu, 58
 przykładowy program,
 60
 ustawianie liczby, 59
 ustawianie tekstu, 59

EDX, 10
EFLAGS, 11, 13
EIP, 13
ekran powitalny,
 Patrz SplashScreen
ELSE, 35, 173
ELSEIF, 35, 173
EM_EXSETSEL, 83, 84
EM_FINDTEXT, 83, 84
EM_SETCHARFORMAT, 84
EM_SETSEL, 84
EM_SETTEXT, 84
END, 184
EndDialog, 50
ENDIF, 35, 173
ENDM, 177
ENDP, 27, 178
ENDS, 43, 185
ENDW, 36, 174
ENTER, 139
EQU, 177
ES, 11
ESI, 10
ESP, 10
etykiety, 29
 kodu, 172
EVEN, 172
exe, 7
EXIT, 186
EXITM, 177
ExitProcess, 16, 50
EXTERN, 183
EXTERNDEF, 183
Extra Segment, 11

F

FARDATA, 186
FARDATA?, 186
FASM, 124
FASTCALL, 123
FDBG, 124
FINDTEXT, 83
FIRST_IPADDRESS, 78
flagi, 11

FloatToStr, 103
FloatToStr2, 103
FOR, 182
FORC, 182
FOURTH_IPADDRESS, 78
FPU, 102
FS, 11
funkcje
 powrót, 27
 tworzenie, 27
 wywołanie, 26
FWORD, 42, 175

G

GetDlgItem, 72
GetDlgItemInt, 58
 bSigned, 58
 hDlg, 58
 lpTranslated, 58
 nIDDlgItem, 58
GetDlgItemText, 58
 hDlg, 58
 lpString, 58
 nIDDlgItem, 58
 nMaxCount, 58
gethostbyname, 100
gethostname, 100
GetModuleHandle, 50
GetProcAddress, 98
 hModule, 99
 lpProcName, 99
GetStdHandle, 45
GoAsm, 124
GOTO, 178
GROUP, 185
GroupBox, 62
GS, 11
GWL_EXSTYLE, 94
GWL_HINSTANCE, 94
GWL_ID, 94
GWL_STYLE, 94
GWL_USERDATA, 94
GWL_WNDPROC, 94

H

HIGH-WORD, 10
HLT, 139
Hooks, 87
 instalowanie, 88
 klawiaturowa, 89
 rodzaje, 87
hostent, 100

I/O Privilege Level, 12
 ID, 12
 Identification, 12
 IDIV, 20, 140
 IF, 12, 35, 174
 IMUL, 19, 140
 IN, 141
 INC, 20, 141
 INCLUDE, 189
 INCLUDELIB, 183
 inet_ntoa, 100
 InitCommonControls, 50
 inkrementacja, 20
 INS, 142
 INSTR, 187
 instrukcje
 arytmetyczne, 18
 AVX, 119
 kopiowania, 18
 koprocatora, 105
 logiczne, 20
 MMX, 110
 operujące na
 łańcuchach, 30
 porównania, 29
 przesunąć bitowych, 21
 rodzaje zestawów, 179
 skoku, 29
 SSE, 111
 SSE2, 113
 SSE3, 115
 SSE4, 117
 SSSE3, 116
 zmiany w x64, 122
 związane
 z procedurami, 26
 INT, 14, 142
 Interrupt Flag, 12
 INVOKE, 17, 179
 IOPL, 12
 IPAddress, 78
 czyszczenie zawartości, 81
 pobieranie adresu, 78
 powiadomienia, 81
 przykładowy program, 82
 ustawianie adresu, 80
 ustawianie focusu, 81
 IPM_CLEARADDRESS, 81
 IPM_GETADDRESS, 78
 IPM_SETADDRESS, 80, 81
 IPM_SETFOCUS, 81
 IPN_FIELDCHANGED, 81

IRET, 143
 IRETD, 143
 IsDlgButtonChecked, 60
 hDlg, 60
 nIDButton, 60

J

JA, 30
 JAE, 30
 JB, 30
 JBE, 30
 Jcc, 143
 JE, 30
 JG, 30
 JGE, 29, 30
 JL, 30
 JLE, 30
 JNE, 30
 JNZ, 30
 jWasm, 124
 JZ, 30

K

K3D, 181
 kernel32.dll, 16
 kilobajt, 10
 klawiatura, 52
 kody ASCII, 192
 kompilatory Asemblera
 x64, 124
 komunikacja między
 wątkami, 96
 koniunkcja, 20
 konsola w Windows, 45
 kolory, 46
 pobieranie danych, 45
 przykładowy program, 47
 tworzenie aplikacji, 45
 wyświetlanie danych, 45
 konsolidator, 7
 kontrola przepływu, 35, 173
 kontrolki
 listy, *Patrz* ListBox
 listy rozwijanej,
 Patrz ComboBox
 nowoczesne, 85
 paska postępu,
 Patrz ProgressBar
 podpowiedzi,
 Patrz Tooltip
 pola tekstowego,
 Patrz Edit
 przycisku, *Patrz* Button

suwaka, *Patrz* TrackBar
 tekstu sformatowanego,
 Patrz RichEdit
 widoku
 drzewa,
 Patrz TreeView
 listy, *Patrz* ListView
 wprowadzania adresu
 IP, *Patrz* IPAddress
 wyboru
 pojedynczego,
 Patrz RadioButton
 wielokrotnego,
 Patrz CheckBox
 konwencja wywołania x64,
 123
 kopiowanie, 18
 koprocesor, 102
 instrukcje, 105
 przykładowy program,
 109
 rejstry, 102

L

LABEL, 172
 LAHF, 144
 LAR, 145
 LB_ADDSTRING, 64
 LB_GETCOUNT, 64
 LB_GETCURSEL, 64
 LB_SETCURSEL, 64
 LEA, 40, 145
 LEAVE, 145
 LGDT, 146
 LGS, 146
 liczby
 zmiennoprzecinkowe, 103
 licznik, 10
 LIDT, 146
 linker, 7
 lista, *Patrz* ListBox
 lista rozwijana,
 Patrz ComboBox
 ListBox, 64
 komunikaty, 64
 przykładowy program,
 64
 ListView, 68
 elementy, 69
 kolumny, 68
 przykładowy program, 69
 LLDT, 147
 LMSW, 147
 LoadIcon, 50

LoadLibrary, 98
 lpLibFileName, 98
 LOCAL, 40, 178
 LOCK, 147
 LODS, 32, 148
 lokalny adres IP, 99
 LOOP, 148
 LOW-WORD, 10
 LVCF_FMT, 68
 LVCF_SUBITEM, 68
 LVCF_TEXT, 68
 LVCF_WIDTH, 68
 LVCFMT_CENTER, 68
 LVCFMT_LEFT, 68
 LVCFMT_RIGHT, 68
 LVCOLUMN, 68
 cchTextMax, 69
 fmt, 68
 imask, 68
 iSubItem, 69
 lx, 69
 pszText, 69
 LVIF_IMAGE, 69
 LVIF_PARAM, 69
 LVIF_STATE, 69
 LVIF_TEXT, 69
 LVITEM, 69
 cchTextMax, 69
 iItem, 69
 imask, 69
 iSubItem, 69
 pszText, 69
 LVM_INSERTCOLUMN, 68
 LVM_INSERTITEM, 69
 LVM_SETITEM, 69
 LVS_REPORT, 68

L

łańcuchy znaków, 30, 187

M

M_BUTTONDBCLK, 51
 MACRO, 178
 makroinstrukcje, 35, 177
 do tworzenia pętli, 36
 kontrolni przepływu, 35
 manifest, 85
 MapViewOfFile, 99
 MASM, 8
 MASM64, 124
 megabajt, 10
 menu, 55
 przykładowy program,
 56

MessageBox, 16
 MF_BITMAP, 56
 MF_CHECKED, 56
 MF_DISABLED, 56
 MF_GRAYED, 56
 MF_MENUBREAK, 56
 MF_OWNERDRAW, 56
 MF_STRING, 56
 młodsze słowo, 10
 MMWORD, 42, 190
 MMX, 13, 42, 110, 181
 instrukcje, 110
 mnożenie
 bez znaku, 19
 ze znakiem, 19
 MODEL, 187
 MOV, 18, 148
 MOVS, 30, 149
 MOVSB, 150
 MOVZX, 150
 MSDN, 16
 MUL, 19, 151
 mysz, 51

N

NASM, 124
 NEG, 151
 Nested Task, 12
 nibble, 9
 NIM_DELETE, 93
 NMHDR, 82
 code, 82
 hwndFrom, 82
 idFrom, 82
 NMIPADDRESS, 81
 hdr, 81
 iField, 81
 iValue, 81
 NOP, 152
 NOT, 21, 152
 NOTIFYICONDATA, 92
 nowoczesne kontrolki, 85
 NT, 12

O

obj, 7
 obsługa okna, 93
 odczyt
 klawiatury, 52
 myszy, 51
 odejmowanie, 19
 odpluskwacz, 7
 odwzorowanie plików
 w pamięci, 99

OF, 12
 OFFSET, 39
 okno dialogowe, 47
 właściwości, 48
 Olly Debugger, 8
 opcode, 196
 operacje na plikach, 52
 przykładowy program,
 54
 operatory porównania, 36
 opkody procesora 80x86,
 196
 OPTION, 190
 OR, 21, 152
 ORG, 172
 OUT, 153
 OUTS, 153
 Overflow Flag, 12
 OVERFLOW?, 36

P

Parity Flag, 11
 PARITY?, 36
 pasek postępu,
 Patrz ProgressBar
 PBM_GETPOS, 65
 PBM_GETRANGE, 65
 PBM_GETSTEP, 65
 PBM_SETPOS, 65
 PBM_SETRANGE, 65
 PBM_SETSTEP, 66
 PBM_STEPIT, 66
 PBRANGE, 65
 pętla, 36
 PF, 11
 pliki
 .manifest, 86
 odwzorowane
 w pamięci, 99
 operacje, 52
 pobieranie lokalnego
 adresu IP, 99
 poczwórne słowo, 10
 podmiana procedury
 obsługi okna, 93
 przykładowy program,
 95
 podpięcia do Windows,
 Patrz Hooks
 odpowiedź, *Patrz* Tooltip
 podwójne słowo, 9
 pole tekstowe, *Patrz* Edit
 POP, 13, 18, 154

POPA, 154
 POPAD, 154
 POPCONTEXT, 190
 POPF, 155
 POPFD, 155
 porównanie, 29
 operatorzy, 36
 potęgi liczby dwa, 193
 powrót
 z funkcji/procedury, 27
 powtórzenia, 182
 półbajt, 9
 PROC, 27, 179
 procedury, 178
 powrót, 27
 tworzenie, 27
 wywołanie, 26
 programowanie
 w Windows, 45
 ProgressBar, 65
 komunikaty, 65
 przykładowy program, 66
 PROTO, 179
 przerwania, 14
 przesunięcia
 arytmetyczne, 21, 22
 bitowe, 21
 logiczne, 21, 22
 logiczne podwójnego rejestru, 24
 przycisk, *Patrz* Button
 przyrównania, 177
 PUBLIC, 184
 PURGE, 178
 PUSH, 13, 17, 18, 155
 PUSHA, 155
 PUSHAD, 155
 PUSHCONTEXT, 190
 PUSHF, 156
 PUSHFD, 156

Q

quad word, 10
 QWORD, 42, 175

R

RadioButton, 62
 przykładowy program, 62
 RADIX, 190
 RCL, 23, 156
 RCR, 24, 157
 ReadFile, 45

REAL10, 42, 176
 REAL4, 42, 176
 REAL8, 42, 176
 RECORD, 188
 rejestry, 10
 bazowy, 10
 danych, 10
 flag, 11
 indeksowe, 10
 koprocatora, 13, 102
 MMX, 13
 ogólnego przeznaczenia, 10
 procesora x64, 121
 segmentowe, 11
 SSE, 13
 wskaźnikowe, 10
 rekordy, 188
 REP, 157
 REPE, 158
 REPEAT, 37, 174, 182
 REPNE, 158
 REPNZ, 158
 REPZ, 158
 Resume Flag, 12
 RET, 27, 158
 RF, 12
 RichEdit, 83
 formatowanie, 84
 przejście do określonego wiersza, 84
 przykładowy program, 85
 wyszukiwanie tekstu, 83
 zamienianie wybranego tekstu, 84
 zaznaczanie całego tekstu, 83
 rodzaje zestawów instrukcji, 179
 ROL, 23, 159
 ROR, 24, 159
 rotacja
 bez użycia flagi CF, 23, 24
 z użyciem flagi CF, 23, 24

S

SAFESSEH, 191
 SAHF, 160
 SAL, 21, 160
 SAR, 22, 161

SBB, 162
 SBYTE, 175
 SCAS, 33, 162
 SCF_ALL, 84
 SDWORD, 175
 SECOND_IPADDRESS, 78
 SEGMENT, 185
 segmenty, 184
 uproszczone, 185
 SendMessage, 62
 hWnd, 63
 lParam, 63
 Msg, 63
 wParam, 63
 SEQ, 185
 SETcc, 163
 SetConsoleTextAttribute, 46
 SetDlgItemInt, 59
 bSigned, 59
 hDlg, 59
 nIDDlgItem, 59
 uValue, 59
 SetDlgItemText, 59
 hDlg, 59
 lpString, 59
 nIDDlgItem, 59
 SetWindowLong, 93
 dwNewLong, 94
 hWnd, 94
 nIndex, 94
 SetWindowsHookEx, 88, 89
 dwThreadId, 88
 hMod, 88
 idHook, 88
 lpfn, 88
 SF, 12
 SGDT, 164
 Shell_NotifyIcon, 92
 SHL, 21, 160
 SHLD, 24, 164
 SHR, 22, 161
 SHRD, 24, 165
 SI, 10
 SIDT, 164
 Sign Flag, 12
 SIGN?, 36
 SIZESTR, 188
 skok, 29
 SLDT, 165
 słowo, 9
 młodsze, 10
 poczwórne, 10
 podwójne, 9
 starsze, 10

SMSW, 166
 Splashscreen, 89
 przykładowy program, 91
 wykonanie, 90
 SS, 11
 SSE, 13, 42, 110
 instrukcje, 111
 SSE2, 113
 SSE3, 115
 SSE4, 117
 SSSE3, 116
 STACK, 187
 Stack Segment, 11
 stałe, 41
 starsze słowo, 10
 STARTUP, 187
 STC, 166
 STD, 166
 STL, 167
 stos, 13
 STOS, 33, 167
 STR, 167
 StrToFloat, 104
 STRUCT, 43, 188
 struktury, 42, 188
 SUB, 19, 168
 SUBSTR, 188
 suwak, *Patrz* TrackBar
 SWORD, 176
 systemy liczbowe, 9
 binarny, 9
 heksadecymalny, 9
 szablony aplikacji DOS, 14

T

tablica kodów ASCII, 192
 TBM_GETPOS, 71
 TBM_SETRANGEMAX, 70
 TBM_SETRANGEMIN, 70
 TBYTE, 176
 tekst sformatowany,
 Patrz RichEdit
 TEST, 168
 TEXTEQU, 177
 TF, 12
 THIRD_IPADDRESS, 78
 TOOLINFO, 73, 75, 76
 Tooltip, 71
 dostosowywanie
 wyglądu, 73
 hiperłącza, 76

 podmiana procedury
 obsługi okna, 74
 przykładowe
 programy, 77
 tworzenie, 72
 TrackBar, 70
 pobieranie pozycji
 suwaka, 71
 przykładowy program,
 71
 ustawianie zakresu, 70
 TranslateMessage, 52
 Trap Flag, 12
 Tray, 91
 przykładowy program,
 93
 TreeView, 66
 dodawanie elementów,
 66
 przykładowy program,
 67
 sprawdzanie
 klikniętego elementu,
 67
 TTF_PARSELINKS, 76
 TTI_ERROR, 74
 TTI_INFO, 74
 TTI_NONE, 74
 TTI_WARNING, 74
 TTM_ADDTOOL, 73
 TTM_POP, 76
 TTM_SETMAXTIPWIDTH,
 74
 TTM_SETTITLE, 74
 TTS_BALLOON, 74
 TTS_CLOSE, 74
 TV_INSERTSTRUCT, 66
 hInsertAfter, 66
 hParent, 66
 item, 67
 TVI_FIRST, 66
 TVI_LAST, 66
 TVI_ROOT, 67
 TVI_SORT, 67
 TVITEM, 67
 TVM_INSERTITEM, 66
 TVN_SELCHANGED, 67
 tworzenie
 bibliotek DLL, 97
 funkcji, 27
 menu, 55
 procedur, 27
 TYPEDEF, 188

U

UnhookWindowsHookEx,
 89
 UNION, 189
 UnmapViewOfFile, 99
 UNTIL, 37, 174
 UNTILCXZ, 174
 uproszczone segmenty, 185

V

VERR, 169
 VERW, 169
 VIF, 12
 VIP, 12
 Virtual 8086 Mode, 12
 Virtual Interrupt Flag, 12
 Virtual Interrupt Pending,
 12
 VM, 12

W

WAIT, 169
 wartość zmiennej, 39
 warunkowa kontrola
 przepływu, 173
 WH_CALLWNDPROC, 87
 WH_CALLWNDPROCRET,
 87
 WH_CBT, 87
 WH_DEBUG, 87
 WH_FOREGROUNDIDLE,
 87
 WH_GETMESSAGE, 87
 WH_JOURNALPLAYBACK,
 87
 WH_JOURNALRECORD,
 87
 WH_KEYBOARD, 87
 WH_KEYBOARD_LL, 87, 89
 WH_MOUSE, 87
 WH_MOUSE_LL, 88
 WH_MSGFILTER, 88
 WH_SHELL, 88
 WH_SYSMSGFILTER, 88
 WHILE, 36, 174, 183
 widok
 drzewa, *Patrz* TreeView
 listy, *Patrz* ListView
 wielowątkowość, 95
 komunikacja, 96
 przykładowy program,
 97

Win32 Programmer's Reference, 16
 WIN32.HLP, 16
 WinAPI, 15
 WinAsm, 8
 ścieżki do katalogów, 8
 Windows
 adres IP, *Patrz* IPAddress
 lokalnego komputera, 99
 alokacja bloków pamięci, 52
 aplikacje wielowątkowe, 95
 biblioteki DLL, 97
 budowa programu, 15
 ekran powitalny, *Patrz* SplashScreen
 konsola, 45
 lista, *Patrz* ListBox
 lista rozwijana, *Patrz* ComboBox
 nowoczesne kontrolki, 85
 Windows
 odczyt
 klawiatury, 52
 myszy, 51
 okno dialogowe, 47
 operacje na plikach, 52
 pasek postępu, *Patrz* ProgressBar
 pliki odwzorowane w pamięci, 99
 podmiana procedury obsługi okna, 93
 podpięcia do systemu, *Patrz* Hooks
 odpowiedź, *Patrz* ToolTip
 pole tekstowe, *Patrz* Edit
 program Witaj, świecie!, 16
 programowanie, 45
 przycisk, *Patrz* Button
 suwak, *Patrz* TrackBar
 tekst sformatowany, *Patrz* RichEdit
 tworzenie menu, 55
 widok
 drzewa, *Patrz* TreeView
 listy, *Patrz* ListView
 wybór
 pojedynczy, *Patrz* RadioButton

wielokrotny, *Patrz* CheckBox
 zasobnik systemowy, *Patrz* Tray
 Windows on Windows 64, 120
 WM_CHAR, 52, 93
 WM_CLOSE, 50
 WM_COMMAND, 50, 57, 92
 WM_INITDIALOG, 50, 53, 90
 WM_KEYDOWN, 52
 WM_KEYUP, 52
 WM_LBUTTONDOWN, 51, 76
 WM_LBUTTONUP, 51
 WM_MBUTTONDOWN, 51
 WM_MBUTTONUP, 51
 WM_MOUSEMOVE, 51, 76
 WM_NOTIFY, 67
 WM_RBUTTONDOWNCLK, 51
 WM_RBUTTONDOWN, 51, 76
 WM_RBUTTONUP, 51
 WM_SHELLNOTIFY, 92
 word, 9
 double, 9
 high, 10
 low, 10
 quad, 10
 WORD, 41, 176
 WOW64, 120
 WriteFile, 45
 WS_EX_CLIENTEDGE, 57
 WS_EX_DLGMOD
 ↪ ALFRAME, 57
 WS_EX_STATICEDGE, 57
 WS_GROUP, 62
 WSStartup, 100
 wskaźnik
 bazowy, 10
 stosu, 10
 wybór
 pojedynczy, *Patrz* RadioButton
 wielokrotny, *Patrz* CheckBox

wyświetlanie liczb
 zmiennooprzecinkowych, 103
 wywołanie
 funkcji, 26
 konwencja dla x64, 123
 procedury, 26

X

x64, 120
 adresowanie, 124
 debuggery, 124
 format plików wykonywalnych, 120
 kompilatory
 Asemblera, 124
 konwencja wywołania, 123
 rejestry procesora, 121
 zmiany w instrukcjach, 122
 x86, 10
 XCHG, 170
 XLAT, 170
 XLATB, 170
 XMM, 182
 XMMWORD, 42, 191
 XOR, 21, 171

Y

YMMWORD, 191

Z

zakres, 183
 zaokrąglenie, 104
 zaprzeczenie, 21
 zasobnik systemowy, *Patrz* Tray
 Zero Flag, 12
 ZERO?, 36
 ZF, 12
 zmienne, 41
 adres, 39
 lokalne, 40
 rozmiar, 41, 42
 wartość, 39
 znaczniki, 11

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

LEKSYKON KIESZONKOWY

ASEMBLER

„Assembler” to słowo wywołujące przerażenie u laików i dreszcz emocji u specjalistów. Osoby znające ten język są uważane za elitę wśród profesjonalnych programistów. Nic dziwnego – w końcu nikt lepiej niż oni nie zna podstaw działania współczesnych procesorów i technik umożliwiających tworzenie bardzo wydajnych programów komputerowych. Jednak czy wiedza o Asemblerze naprawdę jest tajemna i dostępna tylko nielicznym? Czy język ten jest aż tak trudny do opanowania? Wcale nie!

Przekonaj się o tym, sięgając po książkę *Asembler. Leksykon kieszonkowy*. Przystępnie i rzeczowo wyjaśni Ci ona sposób działania procesorów zgodnych z architekturą x86, przedstawi strukturę 16-bitowych programów dla środowiska DOS i 32-bitowych programów dla systemów operacyjnych Windows, a także poszerzy Twoją wiedzę o informacje dotyczące Asemblera dla architektury 64-bitowej. Dzięki lekturze poznasz instrukcje języka, najważniejsze techniki programistyczne, sposoby używania elementów interfejsu użytkownika oraz narzędzia niezbędne w pracy programisty.

- Podstawowe pojęcia związane z Asemblerem i przydatne narzędzia
- Wiadomości na temat architektury x86 i korzystania z rejestrów procesora
- Budowa 16-bitowych programów DOS i 32-bitowych programów Windows
- Korzystanie z makroinstrukcji oraz różnych typów danych
- Tworzenie programów w systemie Windows i używanie kontrolki UI

Nauucz się wydajnie programować w Asemblerze!

helion.pl
księgarnia internetowa



Helion

Nr katalogowy: **7847**

Sprawdź najnowsze promocje:

● <http://helion.pl/promocje>

Książki najchętniej czytane:

● <http://helion.pl/bestsellery>

Zamów informacje o nowościach:

● <http://helion.pl/nowości>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

Księgarnia internetowa:
<http://helion.pl>

Zamówienia telefoniczne:
0 801 339900
0 601 339900

sięgnij po **WIECEJ**



kod korespondencyjny

ISBN 978-83-246-4347-9



Cena 29,00 zł

Informatyka w najlepszym wydaniu