

How to create high-performance programs without wasting your time or money

The screenshot displays the Power C compiler interface with four main windows:

- C Source Window:** Shows C code for a structure named 'ADDRESS' with members 'name', 'street', 'city', 'state', and 'zip'. A watchpoint is set on the 'size' member.
- Output Window:** Displays the program's output: "Joe Bob's Texas", "1251 Cow Chip Trail", and "Fort Worth TX 76442".
- Watch Point Window:** Shows a message: "A watchpoint has natured. size > 0. Old value was 8. New value is 122. Space bar will clear notice & watchpoint".
- Variables Window:** Lists memory addresses and values for variables: 'addr' (far ptr), 'addr->name' (array), 'addr->street' (array), 'addr->city' (array), 'addr->state' (array), 'addr->zip' (array), 'size' (int), and 'job' (array of pointers).

Below the screenshot is the **Power Trace Debugger** logo.

The High-Performance C Compiler



Step 1: The \$19.95 Power C compiler

Power C is the new ANSI compatible C compiler that runs faster than Microsoft C[®] and has more functions than Turbo C[®]. Power C combines high-performance software with superb documentation, all for less than the price of most C books alone. It's your fast route to fast programs without the fast bucks.

The quality of the Power C documentation makes it easier to learn C. The manuals that accompany our competitor's products are terse at best. They contain little or no information about C, and very few examples. In contrast, the Power C book includes a step-by-step tutorial and is chock-full of example programs. Most of our customers are saying that it's the best C book they've ever used.

The quantity of functions in the Power C library makes it easier to accomplish your programming tasks. The Power C library contains more than 420 functions, a superset of the functions in Microsoft C[®] 4.0 and Turbo C[®] 1.0. In addition, Power C includes a large number of video and graphics functions. You get super-fast functions for drawing lines, boxes, circles, ellipses, pie charts, and more.

The speed of the Power C compiler makes programming fast. Power C's integrated *Make* utility saves you time and effort by automatically managing your large programming projects. If you modify your program, Power C makes a new version by recompiling only the files that have changed. The compiled programs are equally fast. Just check out the performance chart. See how much time and money you save with Power C.

Performance / PriceChart (execution times in seconds)			
	Power C	Quick C [®]	Turbo C [®]
1) fib	23.8	53.4	26.4
2) sieve	27.6	43.2	25.5
3) tdbl	3.5	9.0	9.6
4) diskio	13.5	14.4	14.3
5) report	11.0	71.7	60.7
6) drystone	36.6	41.6	31.8
Compile/Link	73.9	113.5	81.4
EXE File Size	25120	32092	27184
Compiler Price	\$19.95	\$99.00	\$99.95
Debugger Price	\$19.95	N/C	N/A
Library Source	\$10.00	\$150.00	\$150.00
Total Cost	\$49.90	\$249.00	\$249.95

N/C no charge - N/A not available
Benchmarks compiled using Make utility, command-line compiler, and medium memory model

Step 2: The \$19.95 Power Trace debugger

Power Trace is the new state-of-the-art C debugger that makes Microsoft's Codeview[®] look like old technology. Power Trace reduces the time you spend debugging your C programs by at least a factor of 10. With Power Trace, you work smarter instead of harder. Actually, using Power Trace is so much fun that debugging doesn't even feel like work anymore.

Power Trace shows you 7 windows of program information: 1) C source statements, 2) screen output, 3) variables, 4) watch points, 5) memory, 6) symbols, and 7) assembly instructions. You can view a single window or as many as 4 windows at the same time (as shown on the screen above). Eight predefined window arrangements are available at the press of a key, or you can design your own.

Power Trace has a unique animated trace feature that shows the flow of execution in vivid detail - not just line by line, but statement by statement. It's like watching the bouncing ball as the cursor dances over your C source statements. You can single step your program or let it run continuously at either trace or full speed. You can easily control the execution of the program by setting an unlimited number of break points and up to 32 watch points. An execution profile shows you how many times each C statement has executed.

Power Trace is loaded with many other advanced features. Power Trace automatically displays *all* of your variables (including arrays and structures), saving you from having to remember and type their names. The virtual output window lets you see the screen output from your program while simultaneously viewing any of the other windows. Interruptible input allows you to get control even while your program is reading input from the keyboard. Backwards tracing gives you the ability to trace backwards through the execution path.

With all its advanced features, the single most important feature of Power Trace is simple operation. With Power Trace, you won't waste any time trying to understand or remember cryptic commands. A single keystroke is all it takes. Help screens show you which key to press and pop-up menus list your options. Invest just 10 minutes of your time with Power Trace now, and you'll save hours from now on.



Technical Specifications

Minimum System Requirements: DOS 2.0 or later, 320K memory, 2 floppy drives or hard drive. Runs on IBM PC, XT, AT, PS/2 and compatibles.

Power C

Power C includes the Power C compiler with integrated *Make* utility, the Power C linker, the Power C libraries (420 functions), and the Power C book (680 pages). Power C supports the proposed ANSI standard, IEEE floating point math, 8087/80287 math coprocessor, auto-sensing of the 8087/80287, automatic register variables, unlimited program size, mixed memory model with near & far pointers, interrupt trapping with memory resident capability, graphics for the CGA, EGA, VGA, & Hercules adapters, and the following functions . . .

abort	curscrl	farstrncat	frexp	intdosx	move__to	setbuf	strncat
abs	curslin	farstrncmp	fscanf	ioctl	movedata	setcbkr	strncmp
absread	cursoff	farstrncpy	fseek	isalnum	movmem	setcolor	strncpy
abswrite	curson	farstrnicmp	fsetpos	isalpha	__msize	setdate	strnicmp
access	curswrow	farstrnset	fstat	isascii	__nfree	setdisk	strnset
acos	difftime	farstrpbkr	ftell	isatty	__nmalloc	setdta	strpbkr
alloca	disable	farstrchr	ftime	iscntrl	__nmsize	setftime	strchr
allocmem	div	farstrrev	ftoa	isdigit	onexit	setjmp	strrev
asctime	dosexterr	farstrset	fwrite	isgraph	open	setlocale	strset
asin	dostounix	farstrspn	gcvt	islower	outp	setmem	strspn
asm	dup	farstrstr	geninterrupt	isprint	outport	setmode	strstr
assert	dup2	farstrtok	getc	ispunct	outportb	setpixel	strtod
atan	ecvt	farstrupr	getcbkr	isspace	parsfnm	settime	strtok
atexit	ellipse	fartol	getch	isupper	peek	setvbuf	strtol
atof	enable	fclose	getchar	isxdigit	peekb	setvct	strtol
atoi	eof	fcloseall	getche	itoa	pen__color	setverify	strupr
atol	execl	fcvt	getcseg	j0	perror	setvmode	swab
bdosptr	execle	fdopen	getcurdir	jl	pie	setvpage	system
bioscom	execlp	feof	getcwd	jn	plotch	signal	tan
biosdisk	execlpe	ferror	getdate	kbhit	plots	sin	tanh
biosequip	execv	fflush	getdfree	keep	poke	sinh	tell
bioskey	execve	__ffree	getdisk	labs	pokeb	sleep	tempnam
biosmemory	execvp	fgetc	getdseg	ldexp	poly	sopen	time
biosprint	execvpe	fgetpos	getdta	ldiv	poscurs	sound	tmpfile
box	exit	fgets	getenv	lfind	pow	spawnl	tmpnam
brk	__exit	filelength	getfat	line__by	pow10	spawnle	__tolower
bsearch	exitmsg	fileno	getfatd	line__style	printf	spawnlp	tolower
cabs	exp	fill	getftime	line__to	putc	spawnlpe	__toupper
calloc	__expand	fill__style	getkey	localtime	putch	spawnv	toupper
ceil	fabs	findfirst	getpass	locking	putchar	spawnve	tzset
cgets	farcalloc	findnext	getpid	log	putenv	spawnvp	uitoa
chdir	farcoreleft	flood	getpixel	log10	puts	spawnvpe	umask
chmod	farfree	floor	getpsp	longjmp	putw	sprintf	ungetc
chsize	farmalloc	flush	gets	lsearch	qsort	sqrt	ungetch
circle	farmemccpy	__fmalloc	gettime	lseek	raise	srand	unixtodos
__clear87	farmemchr	fmod	getvconfig	ltofar	rand	sscanf	unlink
clearerr	farmemcmp	__fmsize	getvect	malloc	read	stackavail	utime
clock	farmemcpy	fnmerge	getverify	matherr	readatr	stat	va__arg
close	farmemicmp	fnsplit	getvmode	__memavl	readch	__status87	va__end
clrcrn	farmemmove	fopen	getw	memccpy	readdot	stime	va__start
clrcrn2	farmemset	__fpreset	gmtime	memccpy	realloc	stpcpy	vfprintf
__control87	farrealloc	fprintf	hallo	memchr	remove	strcat	vprintf
coreleft	farsetsiz	fputc	harderr	memcmp	rename	strchr	vsprintf
cos	farstrcat	fputchar	hardresume	memcpy	repmem	strcmp	write
cosh	farstrchr	fputs	hardretn	memicmp	rewind	strcmpi	writetech
country	farstrcmp	FP__OFF	hfree	memmove	rmdir	strcpy	writetchs
cprintf	farstrcmpi	FP__SEG	hypot	memset	rmtemp	strcspn	writedot
cputs	farstrncpy	fread	inp	mkdir	sbrk	strdup	y0
creat	farstrcspn	free	inport	mktemp	scanf	strerror	yl
cscanf	farstrdup	__freect	inportb	mktime	searchpath	strftime	yn
ctime	farstristr	freemem	int86	MK__FP	segread	stristr	
ctrlbrk	farstrlen	freopen	int86x	modf	setapage	strlen	
curtblk	farstrlwr		intdos	move__by	setblock	strlwr	

Optional Products

Power Ctrace

Power Ctrace includes the Power Ctrace debugger, example programs from the Power C tutorial (on disk), and the Power Ctrace book (140 pages). Power Ctrace supports C source level debugging, assembly level debugging, graphics debugging on a single monitor, backwards tracing, virtual screen output, interruptible input, execution profiles, unlimited break points, and up to 32 watch points.

Library Source Code

The Library Source Code includes the Power C assembler, the Power C library manager, and all of the C and assembly language source code for the Power C function libraries. The Library Source Code is useful for examining, changing, or extending the operation of one or more of the library functions. The Power C assembler may be used as an alternative to Microsoft's assembler for writing functions in assembly language.

BCD Business Math

The BCD Business Math library includes binary coded decimal floating point routines and financial functions for calculating the time value of money, depreciation, etc. . . . The IEEE floating point routines supplied with Power C are best suited for scientific calculations, whereas the BCD floating point routines are best suited for financial calculations. The BCD routines eliminate inaccuracies caused by rounding.