SOUTHERN SOFTWARE Machine-Language Programs.

All Southern Software machine-language programs are self-relocating. This means that you can load them at any chosen address in memory, in conjunction with other Southern Software programs, or with programs from other suppliers. If you upgrade your memory size, you can choose a new loading location. When you have relocated a program you can dump its core image on disk (with the DUMP command) or on tape, using Southern Software's TSAVE utility, It will then load directly into its final location, using LOAD (from disk) or the SYSTEM command (from tape). All programs will run in 4K (or more) unless otherwise stated.

Name	Price £	Price $	Size (bytes)	Description
DLOAD	4.95	9.95	160	Dynamic load of program segments, from tape.
RENUM	3.95	7.95	512	Renumber BASIC programs.
FGRAF	4.95	9.95	864	Fast Graphics subroutines, callable from BASIC.
XREF	5.95	11.95	480	Cross-Reference display.
SDUMP	2.95	5.95	304	Symbolic Dump.
ZBUG	11.95	23.95	1152	Debug Z80 programs.
TSAVE	4.95	9.95	512	Prepare System tapes (core image dump).
LIFE1	2.00	4.00	1408	Game of LIFE (16 x 64).
LIFE2(16K)	3.00	6.00	6656	Game of LIFE (48 x 64).
SRCH	2.95	6.95	128	Search a BASIC program for a string.
USRN	2.95	6.95	128	Multiple USR calls, for level 2.
EXEC(Disk)	9.50	22.00	1000	Execute Command Lists.
EDIT(16K)	17.50	40.00	3072	Full-Screen Editor for BASIC.
ACCEL(16K)	19.95	44.95	2816	Compiler for Level 2 BASIC.
ACCEL2(16K)	39.95	88.95	5632	Compiler for Disk BASIC.

DLOAD will load a BASIC program segment from tape, linking it to an existing BASIC program, already in memory.
Build new programs from existing subroutines.
Load program overlay segments dynamically, from tape.
Speed up DATA input, by dynamically loading DATA statements.

RENUM renumbers BASIC programs, in situ, in memory.
The new starting line number is specifiable by the user. The new numbering interval is 10. RENUM recomputes line number references after GOTO, GOSUB, RESUME, THEN, ELSE, LIST, DELETE, and RUN. If the recomputation makes the line numbers longer, then RENUM expands the program text to make room.

FGRAF is a set of machine-language subroutines for fast screen graphics.
It is callable from any BASIC program, and is controlled by four BASIC variables (X1,Y1) and (X2,Y2). Six functions are provided:- Draw a line, Erase a line, Moving "ball"
Moving "ball", stopping at any obstacle, Rectangular frame, Solid white rectangle.

Use these routines in ball-game simulations, or to improve data presentation by underlining or framing important items, or by display of histograms, etc.

XREF displays a where-used cross-reference for variables in a BASIC program.
This is a great programming productivity aid. You can quickly identify the usage of all variables in a BASIC program, prior to making changes, XREF scans the BASIC program in conjunction with its associated internal symbol table (built by the BASIC interpreter). It then displays every active variable name, complete with a list of every line-number in which that variable is used. The variable types (INTEGER, SINGLE, DOUBLE, and STRING) are correctly distinguished and displayed.

southern software

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

SDUMP displays the current values of all BASIC scalar variables.

Using the BASIC internal symbol table, SDUMP displays the names and values of all active scalar variables. So if your program stops with an error condition, you can call SDUMP from the keyboard to get an immediate "snapshot" of the situation when the error occured. Alternatively you can imbed calls to SDUMP in your program at strategic points. A must for debugging.

ZBUG is a software interpreter for Z80 machine-code.

It is used for debugging machine-language programs for the TRS-80. You can step through the program, under the control of ZBUG, displaying and/or modifying the contents of registers or memory. Functions provided are:

1) Instruction trace. At each trace point ZBUG displays the last executed instruction, the program counter, the stack pointer, the main registers HL, DE, BC and AF, and the index registers IX and IY.
2) Selective trace. You can limit trace display, e.g. CALL and RETURN only.
3) Mnemonic address stop. You can set up a number of addresses, each associated with a mnemonic label, at which you want execution to trace and pause.
4) Trace suspension. You can suspend trace altogether below a chosen address, or depth of calling.
5) Hex display of merry.
6) Memory modification (including register values).

TSAVE prepares system tapes.

A tape produced by TSAVE will load as a core-image file directly into memory under the SYSTEM command. In particular you can take backup copies of machine-language programs, including automatic start address, and/or preprimed USR invocation address. TSAVE has the following important advantages:

1) A single tape file can be produced from multiple disjoint segments of memory.
2) You can produce duplicate copies of the file, without rekeying the address ranges.
3) You can verify the saved file to check for a good tape, without rekeying the ranges.
4) You can opt to work with Decimal or Hexadecimal addresses.

LIFE(1 or 2) plays the game of LIFE invented by Prof Conway.

LIFE1 plays the game on a 16 x 64 grid (the TRS-80 screen buffer) in which each cell is alive (an asterisk) or dead (a blank). LIFE2 uses a 48 x 64 grid and utilises the square box graphic, giving a symmetrical representation of the LIFE patterns, without gaps between the cells. Both versions come with a "full-screen" editor, written in BASIC, which enables you to initialise the pattern to any shape on a 16 x 64 grid. LIFE1 runs at better than 5 generations a second, LIFE2 at better than 2. LIFE2 permits you to experiment with patterns that grow too large for LIFE1, for instance the famous Glider Gun.

SRCH lets you scan a BASIC program for any string, up to 16 characters long.

Keep SRCH available in high memory, and invoke it whenever you change a BASIC program, For instance if you delete a line, or remove the use of a variable, search the program to make sure there are no dangling references to the variable or line.

USRN allows you to set up multiple simultaneous USR calls.

It provides, under Level2, a similar function to DEFUSRn in Disk BASIC. You can establish a table of entry points to machine-language or vendor programs, and invoke e.g. the third such routine by USR(3). If you want to pass an argument to that routine, then you use USR(3)(X), where X is any valid BASIC expression.

Dollar cheques accepted, but U.S. customers please add $2.00 for airmail postage.

State if Video-Genie, when ordering.

southern software

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

EXEC - A DOS Command Processor.

Price £9.50 ($22.00). Size 1024 bytes (but usually overlaid).

EXEC gives users of TRSDOS, NEWDOS, and other DOS systems the capability to prepare and execute programs consisting of DOS COMMANDS. In particular you can EXECute a command list automatically on power-up (using TRSDOS AUTO) which can:

1)LOAD one or more core-image files
2)Enter DISK BASIC, setting the BASIC start-up parameters
3)Execute a number of BASIC statements, for instance to set DEFUSR values, or to activate machine-language programs
4)Finally invoke a BASIC application for yourself, or some end-user.

A COMMAND LIST itself is prepared using the BASIC editor, and then saved on disk like any other program, but with the ASCII option, Up to 9 parameters can be passed to the COMMAND LIST at invocation, and these can be substituted anywhere in the commands, or BASIC statements. Default values may be supplied automatically, where the parameters are omitted on the invocation.

Example invocation	COMMAND LIST(called CLIST1)	EXECUTED COMMANDS	
EXEC CLIST1,REPORT	10 LOAD INITPROG/CIM	LOAD IN ITPROG/CIM	
	20 BASIC	BASIC	
	30 @2<3> (param2,default 3)	3	(default used)
	40 49152	49152	(protect memory)
	50 DEFUSR=&F800	DEFUSR=&F8000	
	60 SYSTEM	SYSTEM	
	70 /49152	/49152	
	80 RUN "@1<SALES>"	RUN "REPORT"	(param I used)

A typical use for a command list is for file organisation, for instance to COPY or RENAME selected sets of files. Another particular use is in the automatic activation of the Southern Software programs EDIT and ACCEL2, and, in the case of ACCEL2, in the preparation of a user-defined command to bring up the run-time environment for a compiled program for sale.

southern software

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

EDIT: A Major new product from Southern Software.

This is the way the professionals create programs! Southern Software's new Full-Screen Editor gives you the function and luxury of a large-system editor. Compared with the TRS-80 built-in editor you get faster program correction, with less keystrokes, much greater program visibility, and as a result, far, far, fewer editing errors.

Instead of editing just one program line you can see and modify fifteen at a time. Using typamatic (i.e. automatically repeating) cursor-control keys, you can navigate to any character you want to change, and once there, you can modify, delete or insert characters simply and visibly. The cursor keys also give you controlled automatic scrolling up or down through your program, so that you can "window" on to any block of fifteen lines.

How often have you deleted the wrong line by mistake? How often have you got so tangled up with Change and Insert that in frustration you retyped the whole line? These errors cant happen with the full-screen editor because every change you make is immediately visible. At last you can enjoy programming!

You can navigate round your program in a number of ways - by cursor-controlled scrolling, by line number, or by program content. This ability to "browse" allows you to program creatively and accurately at your keyboard, without pencil and paper. You can find and/or modify one or many occurrences of a program string, and check line number or program variable references. To save keystrokes, you can replicate, copy or move program lines, or blocks of lines.

Whether you program yourself, or whether you rely on someone else to program your system, you cannot afford to pass up the productivity improvements that the Southern Software Full-Screen Editor gives you!

Specifications:

Available on tape (for tape or disk systems), price £17.50 ($40.00).
Size: 3072 bytes (Hex 0C00).
Over 30 commands and functions.
Executes under TRS-80 Level2 BASIC, or DISK BASIC under TRSDOS, NEWDOS, NEWDOS80 and other operating systems.

Highlights:

o Cursor-controlled scrolling giving maximum program visibility.

o All keys automatically repeating (typamatic), including function keys.

o Change program content by overtyping, by deletion, or by direct insertion.

o Insert new lines, or Copy, Move, Replicate, or Delete existing lines.

o Copy, Move, Replicate, or Delete blocks of lines.

o Search program for strings of characters.

o Replace some or all occurrences of one string by another, with only a few keystrokes.

southern software

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

SOUTHERN SOFTWARE's ACCEL and ACCEL2 Compilers for TRS BASIC.
---

o Have you ever wished your BASIC programs would run faster?

o Could you sell your programs, if only you had the time and knowledge to write machine-code?

o Have you ever wished you'd best a Micro with a built-in PASCAL compiler? Is BASIC too slow?

o ft is it that your ore-megacycle CPU seems incapable of getting through more than 500 additions a second?

o Are your thumbs sore, from sitting there, twiddling?

The remedy is simple: Get yourself a BASIC oiler from SOUTHERN SOFTIE.

ACCEL	£19.95	($44.95)	2816 Bytes	Compiler for Level 2 BASIC
ACCEL2	£39.95	($88.95)	5632 Bytes	Compiler for full Disk BASIC

ACCEL2	£44.00	($99.00)	on 35-track diskette. (Not self-relocating).
ACCEL2	£42.00	($94.00)	on EXATRON wafer. (Not compatible with ACULAB).

ACCEL and ACCEL2 are versions of the same product. They will compile a BASIC "source" program into an "object" program which is compatible with the original, except that it runs faster. Performance improvements which can be achieved vary from spectacular (20 to 30 times) to modest (a few percent). Measured examples are given later. Both ACCEL and ACCEL2 will give outstanding improvements on programs of logic, such as games, music synthesis, screen graphics, searching algorithms, etc., while ACCEL2 will give valuable gains, 4 to 5 times, for string handling programs. Neither will help programs that are entirely limited by I/O (disk, printer, tape, or keyboard).

ACCEL2 is a direct extension of ACCEL. It handles the full Disk BASIC language, (under TRSDOS or NEWDOS), whereas ACCEL is limited to Level2. ACCEL2 will also give performance improvements that ACCEL will not, notably in string-handling, in SINGLE and DOUBLE arithmetic, and in manipulation of one-dimensional, fixed-bound arrays. You'll need 16K of memory (or more) to run ACCEL satisfactorily, and 32K of memory for ACCEL2 with Disk BASIC. If you want to use ACCEL2 on Level2 (non-disk) then 16K is viable. Both will produce saleable object programs that run in 16K.

Why you should buy from Southern Software
---

1) Coexistence.

Southern Software programs distributed on cassette (or wafer) are SELF-RELOCATING. When you load the original tape you can choose to locate the compiler anywhere in memory. You can run Southern Software programs concurrently with other Southern Software programs, or with programs from other vendors, and you can upgrade your memory without problems. After relocation, the compiler can be saved on disk using TRSDOS DUMP, or on tape using TRS TBUG or Southern Software TSAVE, for subsequent direct loading.

2) Improvements.

Over the next few years program piracy is going to become common-place, and its prevention will be uneconomic. But only if you buy from the originators can you be sure of improvements that are added. For example purchasers of ACCEL (which was released first) can return their original Southern Software cassette and receive ACCEL2 for £20.00 ($44.00), the difference in price between the two versions. All copies of ACCEL2 shipped after February 81 give faster compilation, lower memory utilisation, and easier operating characteristics than the original ACCEL2 and early purchasers have been able to upgrade with the payment of a £5.00 service charge.

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

The Mechanics of Compilation.

Using ACCEL or ACCEL2, you get the advantages of both interpretation and compilation. Programs are built, modified and debugged in the normal way, using the BASIC editor/interpreter built in to your TRS80 or Video-Genie. When correct, the program is compiled to get improved execution speed. The source form of the program (in BASIC) can be saved in the normal way, using CSAVE and CLOAD, or SAVE and LOAD. But the compiled program no longer has the format of a normal source program, and it cannot be edited or modified in any way (except by recompilation). Nor can it be saved and reloaded, except as a core-image. To save a compiled program on TAPE you will need to purchase also the Southern Software utility TSAVE (£4.95 or $9.95), which produces tapes that can ix reloaded with the SYSTEM command. With ACCEL2, under TRSDOS or NEWDOS, you can save and load compiled programs to or from disk, using routines that are built in to the compiler. ACCEL2 also supports saving and loading of compiled programs on EXATRON Stringy Floppy.

Selling Compiled Programs.

ACCEL and ACCEL2 can significantly enhance programs written in BASIC that you may wish to sell. In a booming but overcrowded market-place, compilation can give your programs an enormous competitive advantage. The core-image tapes produced by TSAVE are self-contained, that is they can be reloaded on any TRS80 using the Level2 SYSTEM command without any other routines being necessary. The disk files produced by ACCEL2 are not self-contained, but a complete product can be achieved by the addition of a second file, the run-time component of ACCEL2. Sale of this component (or of the run-time component of ACCEL on tape) involves the resale of part of a Southern Software product. However it is too small a part to justify collection of royalties, and so the implicit resale will be ignored by Southern Software provided:

- o No part of the compile-time routine is copied or resold.
- o An acknowledgement is given to Southern Software in the program documentation.

Capabilities of the Compilers.

The result of compilation is a program which is a mixture of BASIC statements and directly executing Z80 machine instructions. The run-time routines in ACCEL and ACCEL2 give control to the interpreter when an unoptimised statement is encountered, and then also ensure that the variable values accessed by the interpreter and the compiled code are consistent. The rule is that if a statement contains any operation that the compiler cannot convert to machine-code, then the whole statement is left in interpretive form. So if you are considering the sale of sour compiled programs you should allocate some time to the tuning of the program to the capabilities of the compiler, which are of course directly related to the capabilities of the Z80 CPU chip. Any item not included in the following list, e.g. SIN(X) or A(5,5), will inhibit the optimisation as machine-code of the statement in which it appears, but will not affect correct execution. Translation to machine-code is summarised as follows:

Function	ACCEL	ACCEL2
GOTO,GOSUB,RETURN,RESTORE,IF,THEN,ELSE,CLEAR,ON	Always	Always
LET(assignment),POKE,SET,RESET,POINT,PEEK,USR,VARPTR + - AND OR NOT = (and all comparison operators) Constants, e.g. 123, 12.3, "ABC"	Integers only	All data types
FOR, NEXT	Integers only	Integers only
* / (multiple and divide)	No	All data types
LEN, MID$, LEFT$, RIGHT$, CHR$, ASC, CVI	No	All data types
One-dimensional, fixed-bound arrays	No	All data types

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

Preresolution of Names and Line-numbers.

The BASIC interpreter finds the location of each variable by a sequential search through its dictionary at execution time. By contrast the compiler allocates storage for each variable at compile-time, and replaces each reference to that variable (in an optimised statement) by a direct machine address. Similarly each line reference in GOTO or GOSUB is translated to a branch address, whereas the BASIC interpreter searches sequentially through the program to resolve each target line at run-time. The longer the program, and the more variables it contains, then the greater the performance improvements that result from compilation.

Program Size.

The compiled machine instructions normally occupy more space than the BASIC statements they replace. To counteract this the compiler removes REMARKs from the program, so its final size may be larger or smaller. However, the compiler itself occupies space, and the program length during compilation may expand beyond its final length. So you will need memory in excess of the size of the BASIC program. Space required by the compiler itself:

	Compilation	Execution
ACCEL	2816	256
ACCEL2	5632	1536

ACCEL2 supports control options which enable you to limit compilation to only those parts of the program that are performance critical. This helps contain code expansion.

Examples.

Out of a sample of 72 independently-written programs, 2 failed to run because of improper use of FOR-NEXT (see restrictions). The same sample showed a net reduction in size, though this is untypical.

The following programs were compared for both speed and size, before and after compilation. For consistency of measurement the programs had no REMARKs (which is pessimistic) and no keyboard I/O. The SORT example is instructive because it is possible to run the same program, with equivalent data values, against all four data types. ACCEL shows up badly an this example which consists largely of shuffling array elements, which ACCEL does not optimise. However, it is possible to recode the same example using PEEK and POKE, rather than arrays, to optimise its performance under ACCEL, and this is given for comparison.

Sizes are in bytes, times in seconds. "Gain" is the ratio of speed when compiled to original speed.

	Uncompiled		ACCEL2				ACCEL			
Program	Size	Time	Size	Time	Gain	Compile Time	Size	Time	Gain	Compile Time
Screen Graphics	323	496	519	23	21.6	1	487	23	21.6	1
Disk Dump	691	30.1	1316	10.3	2.9	4				
Income Tax	1184	39	2154	21	1.9	10	1381	37	1.1	5
Game of LIFE	503	30	942	.8	39	3	939	.8	39	2
Blackjack	3173	91	7380	32	2.8	115	5524	57	1.6	86
Mann-Whitney (statistics)	1914	15.5	3212	3.1	5.0	24	1960	15.5	1.0	14
Sort(INTEGER)	714	43.2	1230	1.8	24	4	937	34.4	1.3	3
Sort(SINGLE)	714	43.2	1509	8.2	5.3	5	932	35.4	1.2	3
Sort(DOUBLE)	714	46.8	1923	11.4	4.1	7	932	38.9	1.2	3
Sort(STRING)	716	39.2	1391	4.3	9.1	5	932	32.4	1.2	3
Sort(PEEK,POKE)	913	216					1276	5.7	7.6	7

Restrictions.
---

1) No redefinition of meaning of names. E.g. DEFSNG I : I=1 : DEFINT I : I=1 is disallowed.

2) Programs must be properly structured. Each FOR-NEXT loop must be properly nested and uniquely terminated.

```
10 FOR I= 1 TO 10
20 IF I=5 THEN NEXT                Invalid.
30 PRINT I : NEXT                  Multiple NEXTs
```

3) Behaviour of error conditions is not necessarily compatible. Data-dependent errors, such as OVERFLOW, or function argument out of range, are not necessarily diagnosed. The current line number (used in diagnosis, error handling and in trace) is not accurately maintained.

4) Editing is not possible on the compiled program. The commands AUTO, CLOAD?, CSAVE, DELETE, EDIT, SAVE, and MERGE are not meaningful. NEW, CLOAD, or LOAD must be used to reset the machine to its normal state.

******** Feb 1st 1981 NEWS FLASH! ******** USER EXPERIENCE WITH ACCEL2. *******

Southern Software's ACCEL compiler for Level 2 BASIC was first released at the end of 1979. The upgrade to ACCEL2, giving compilation of the full TRS-80 Disk BASIC language, was released in August 1980. Here's what customers said about ACCEL2:

"Excellent product"
"Very impressed by your BASIC compiler"
"ACCEL2 is amazing!"
"I can see I'll be doing a lot less assembly-language programming in future"
"It's a great compiler!"
"The program is very good, and claims on speed are not exaggerated"
"Very happy with ACCEL2"
"On the average, improved run time by a factor of 7"
"Congratulations on an excellent software package"
"Very pleased with its operation, and the improved speed of my BASIC programs"

Now, since February 1st 1981, a further improvement to ACCEL2 has been available, but still at the same price as the original ACCEL2, and marketed under the same name, Highlights of this latest version are#'

- o Greatly improved compile time for large programs.
- o Less program expansion during compilation.
- o Better ease-of-use, Built-in commands for compiling, and for saving and loading of a compiled program.
- o Better chaining of compiled and non-compiled programs.
- o Compatible support for TRSDOS, NEWDOS, and other operating systems.
- o Comprehensive instructions for selling compiled programs on tape or on disk.

If you're still undecided about purchasing ACCEL2, then send £1.00 for a copy of our "Three Blind Mice" game. The tape contains the same BASIC program in source form, and compiled by ACCEL2. This shows you directly what sort of improvement you can expect from compilation.

All programs are distributed on an "as is" basis, without warranty. Southern Software shall have no liability or responsibility for loss of business caused, or alleged to be caused by the use of these programs.

U.S. customers: Personal dollar cheques accepted, but please add $2.00 per order for airmail postage.

PO Box 39, Eastleigh, Hants, England, SO5 5WQ