

LIGHTNING FAST SAFE AND SOUND

Rapidly write programs and save your files with **Audio Light** utilities for CP/M:

BACKFIELD

MPL—MENU PROGRAMMING LANGUAGE

Our Quality Software will improve your productivity.

THE BACKFIELD

Our backfield software saves and protects your hard disk data with these fine features:

- Backs up a hard disk to floppies
- Backs up selected files or all files
- Automatically selects files that have changed since last backup

The **BACKFIELD** consists of three programs:

FULLBACK backs up an entire disk to another set of disks file-by-file, optimizing the use of all available memory. It writes a special checksum directory file for later use by **QUARTERBACK**.

HALFBACK backs up a large file to multiple disks.

QUARTERBACK automatically determines which files have been changed since the last backup, and backs up these files only.

MPL—MENU PROGRAMMING LANGUAGE

MPL simplifies application programming by using menus to structure programs:

- Data is displayed as items in a menu
- Data is organized by connecting menus together
- Selecting an item in a menu may call an application module
- Message area at top of each menu for module communication
- Selection causes data to be placed in the message area
- Interfaces with other programming languages

MPL is a revolutionary new programming language for the system developer or end user to structure applications in a natural easy-to-use manner. The design of MPL allows application modules to be written in any compiled language supported by CP/M.

PRICE—ORDER INFORMATION

THE BACKFIELD \$150.00

MENU PROGRAMMING LANGUAGE \$175.00

Backfield available third quarter 1982

MPL requires a 24x80 CRT. All software is supplied on 8" single density diskette for CP/M 2.2.

Call (408) 395-0838, or send check to:

AUDIO LIGHT, INC.
146 Town Terrace, Suite 4
Los Gatos, CA 95030

*California residents add 6% for sales tax

Dealer Inquiries Welcome

MPL is a trademark of Audio Light, Inc.
CP/M is a trademark of Digital Research.



Audio Light

Seattle Computer System

Seattle Computer Products' (SCP) translation system consists of two programs: a Z80 translator (called **TRANS86** on the disk, though not the same as Sorcim's) and a compatible 8086 cross assembler (**ASM86**). Both programs run on Z80 processors under CP/M-80 with a minimum of 24K bytes of RAM. The programs do not run on 8080/8085 processors.

The translator accepts source files in Zilog/Mostek mnemonics and produces an 8086 assembler source file in a form acceptable to the SCP 8086 cross assembler. Since the 8086 source format required by the SCP 8086 cross assembler is different from any other 8086 assembler that we know of, you must use these two programs together. The translator places its output on the same disk as the Z80 source code and gives it an .A86 file extension.

The translation is on an instruction-by-instruction basis with no optimization. There appears to be no limit on the file size that may be translated. Not all Z80 instructions are translated, however. Those in the following list will produce an op code error:

| | |
|------|------|
| Cpd | Ldi |
| Cpi | Otdr |
| Ind | Otir |
| Indr | Outd |
| Ini | Outi |
| Inir | Rld |
| Ldd | Rrd |

These op codes are mostly in the block-manipulation set of instructions. Although programmers do use these instructions, they must be manually coded when converting to the 8086. The SCP translator does not support macros and permits use of the following pseudo-ops only:

| |
|----------|
| Db |
| Dm |
| Ds |
| Dw |
| Equ |
| If/Endif |
| Org |

If the Z80 index registers IX and IY are used, they are mapped into memory locations with the labels IX: and IY:. The programmer has to define these locations; otherwise, they will show up on the assembly listing as undefined labels. The Z80 alternate register set (BC',DE',HL') is treated the same way, as memory locations that the programmer must define.

Either the DI or SI register can be used as a temporary IX register by loading one of them from the location IX when required to do indexed instructions. The programmer has to take care of this substitution; the translator does not.

When using the DI register, you must always keep in mind that the only 8086 segment base that can be used with the DI register is the ES segment; the SI register, on the other hand, can reference all the segment bases, defaulting to the DS segment. If this 8086 source code is going to be run under CP/M-86, you have to be careful about using the ES segment register. The CP/M-86 documentation specifically states that ES is not saved through a BDOS call.

For the SCP Z80-to-8086 translator's register usage, see figure 6b.

Translating the Test File

To determine how the three programs actually translate source text, we prepared a file acceptable as input to an assembler and containing all the op codes of the 8080 and Z80. Since the SCP translator could accept only Zilog/Mostek mnemonics, the test text was run through an 8080-to-Z80 filter program before the translation.

Listing 1 presents, side by side and line by line, the original 8080 code, the Sorcim **TRANS86** translation, the Seattle translation, and the **XLT86** translation. Here are reminders of some things to consider when you examine the translations:

- Because of the differences in the architectures of the 8086 and the 8080/Z80, some choices must be made when translating from one architecture to the other. Therefore, some difference in translation is to be expected.