

Upward Migration

Part 1: Translators

Using translation programs to move CP/M-86 programs to CP/M and MS-DOS

Roger Taylor and Phil Lemmons
c/o BYTE Publications Inc.
POB 372
Hancock, NH 03449

In software migrations as in any other, the first thing you must know is where you are and where you are going. If a Californian decides to move to Australia, he can call an airline, ask for a ticket from San Francisco to Sydney, and the airline clerk will be happy to reserve a seat for him. If the Californian asks for a ticket from Los Angeles-or-San Francisco to Sydney, however, the airline clerk will find the request confusing. If the Californian asks for a ticket from Los Angeles-or-San Francisco to Sydney under Catholic rule, or a ticket from Los Angeles-or-San Francisco to Sydney under Protestant rule, the airline clerk will probably say, "You've already spent too much time in the hot tub, buddy. Stay in California. You're right where you belong."

In this little parable of modern

times, Los Angeles is the 8080, San Francisco is the Z80, Sydney is the 8086, Catholicism is CP/M-86, and Protestantism is MS-DOS. The

**XLT86 takes 8080
source code and
converts it into
8086 source code
in an intelligent
manner using
data-flow-analysis
techniques.**

operating systems are represented by religions because they generate similar passions, controversies, true believers, and skeptics.

Is there really any need to explain

why Los Angeles stands for the 8080 and San Francisco for the Z80? Or that the airlines stand for the software houses that have written translation programs?

You've probably guessed who the guy is who's trying to buy the ticket. He's the experienced 8080 or Z80 programmer, and the hot tub symbolizes his strong preference for staying right where he is—on one familiar processor with one familiar operating system. What could be cozier?

The programmer may not have the urge for going, but he has to go to one unfamiliar processor and to both operating systems. And so do the rest of us.

We're lucky to have software from at least three companies to help us along. In this article, the first of two parts, we will review three CP/M-80-to-8086 translator programs. We'll

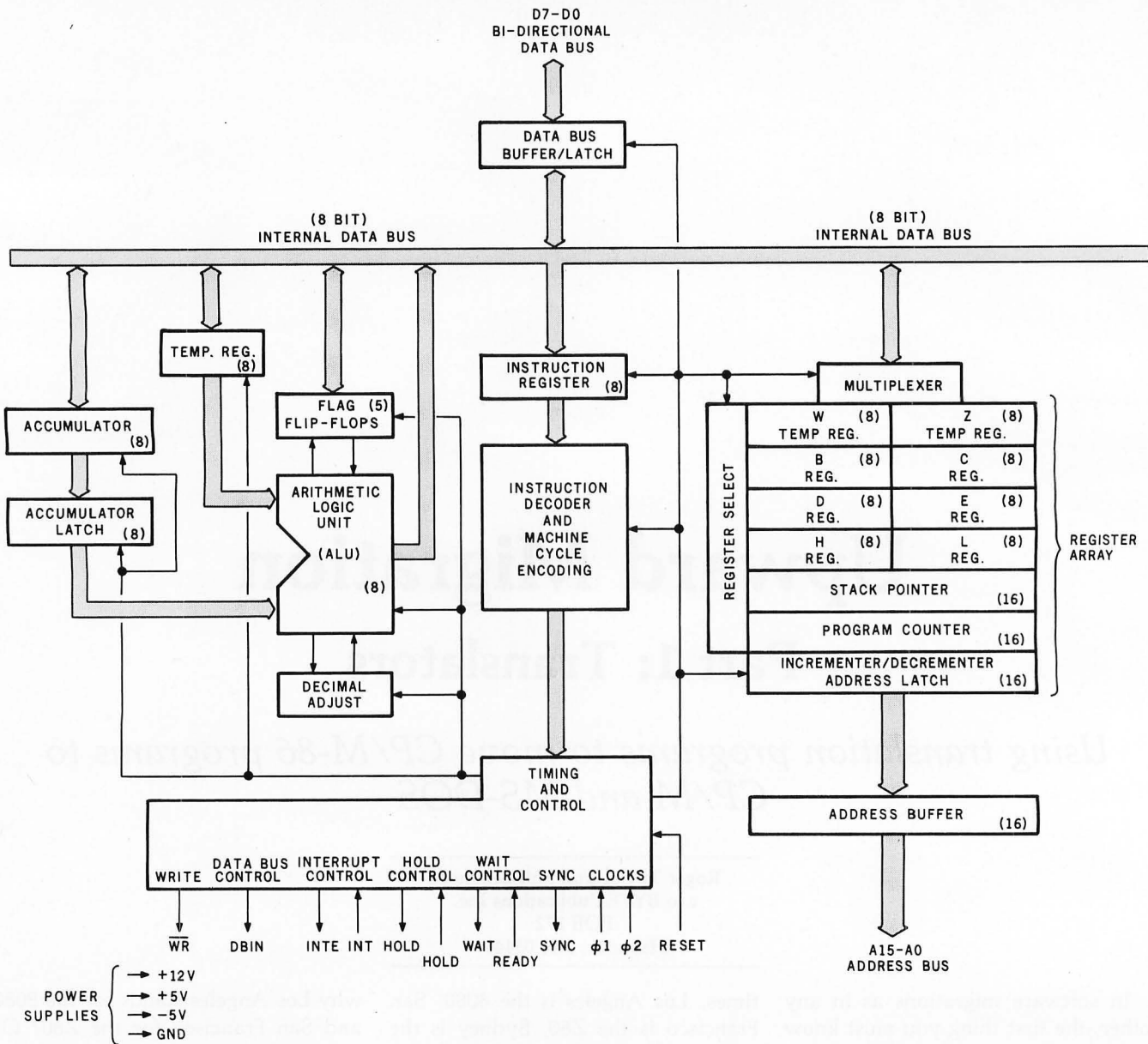


Figure 1: Architecture of the 8080 microprocessor showing the internal registers.

show the output that each of the three translators produced from the same original code. In addition, we'll make some observations about the differences in adapting the translated code to CP/M-86 and MS-DOS. Next month, we'll take a closer (although still not comprehensive) look at CP/M-86 and MS-DOS.

Orientation to the 8086

The first thing we have to do is examine the differences between the familiar 8080 and Z80 microprocessors and the 8086. For reference, figure 1 shows the registers and architecture of the 8080; figure 2 shows

the registers and architecture of the Z80. We'll make few comments about these registers because they are familiar to you if you have 8080 or Z80 source code that you want to translate.

Figure 3 shows the registers and architecture of the 8086. Since the 8086 is less familiar, we'll take a brief look at it for orientation. (For further enlightenment, see *The 8086 Book*, Russell Rector and George Alexy, Osborne/McGraw-Hill, 1980, and *The 8086 Primer*, Stephen P. Morse, Hayden, 1980.) The 8086 is, of course, a 16-bit microprocessor. The 8088 is the same as the 8086 inter-

nally. Externally, however, they appear different due to the 8-bit bus of the 8088 and the 16-bit bus of the 8086. This means that programs that run on the 8088 will also run on the 8086 assuming that the memory resources and peripheral resources are the same. In general, statements in this article that apply to the 8086 apply to the 8088 as well.

The 8086 can access up to 1 megabyte of memory and as many as 65,000 input/output ports. The megabyte of memory is 2^{20} 8-bit bytes; any two consecutive bytes are a 16-bit word. Some 8086 instructions access bytes; others access words.

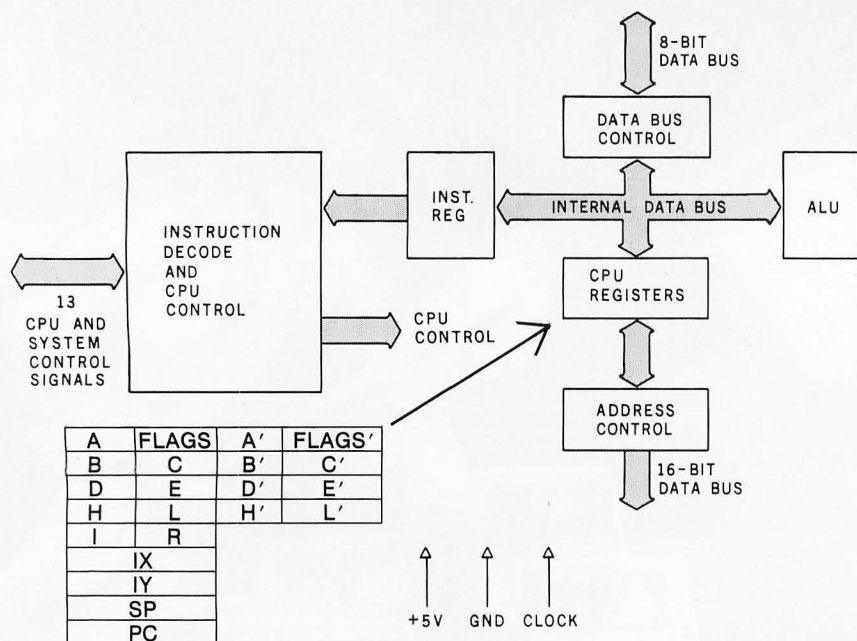


Figure 2: Architecture of the Z80 microprocessor showing the internal registers.

Sixteen bits are not enough to address a megabyte of memory. The 8086 manages to do so, however, by dividing the megabyte of memory into a number of segments of 64K bytes each. Each segment begins at an address that yields an even result when divided by 16.

All calculations of memory addresses in the 8086 involve four special registers called segment registers. The 8080 family has a 16-bit address bus that allows addressing of 65,536 bytes of memory. While the internal registers of the 8088/8086 family also have 16 bits, the external address bus has 20 bits. To get the 20-bit address, the 8086 extends a segment register with 4 low-order bits of 0, and adds the segment register to a 16-bit address from another register, as shown in figure 4.

Each segment register defines what is known as its own "current" segment. Each instruction specifies an offset into a segment. The segment registers, which *cannot* be used interchangeably, are as follows:

CS—The Code Segment register defines the 64K-byte current code segment. When an instruction is fetched, the contents of the program counter are added to the CS register contents to calculate the address of the instruction to be fetched.

DS—The Data Segment register defines the current data segment. With three exceptions, all data memory references are understood in relation to the DS register. (The exceptions are that the stack pointer is used to calculate stack addresses, any data memory addresses calculated using the BP register are taken in relation to the stack segment, and any string operations involving the destination are taken in relation to the extra segment. See SS and ES immediately below.)

SS—The Stack Segment register identifies the current stack segment. References to data memory that use the BP or SP register in calculating the address are understood in relation to the SS register. For example, the PUSH, POP, CALL, INT, and RET instructions use the SS register.

ES—The Extra Segment register plays

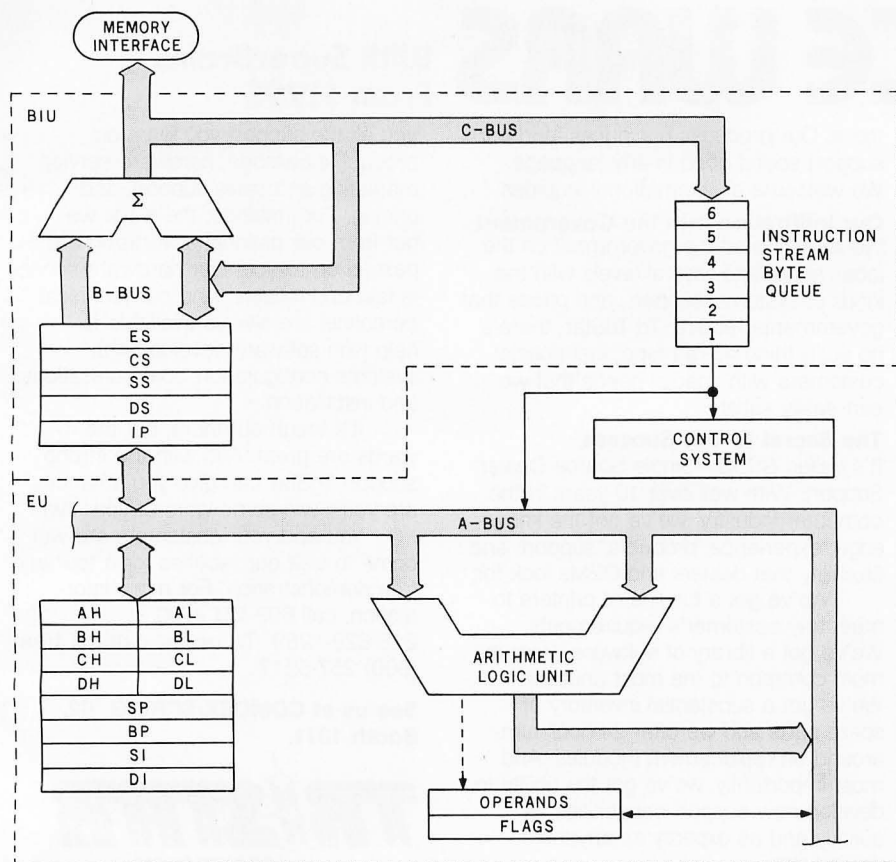


Figure 3: Architecture of the 8086/8088 microprocessor showing the internal registers.

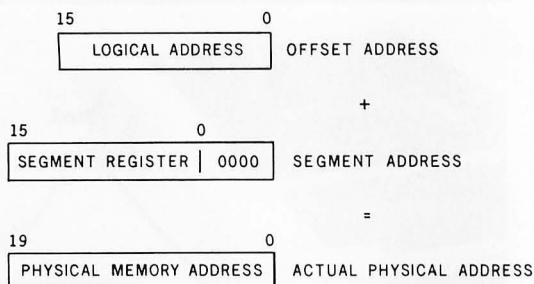


Figure 4: The 8086 extends a segment register with 4 low-order bits of 0. It then adds the segment to a 16-bit address from another register to achieve a 20-bit address.

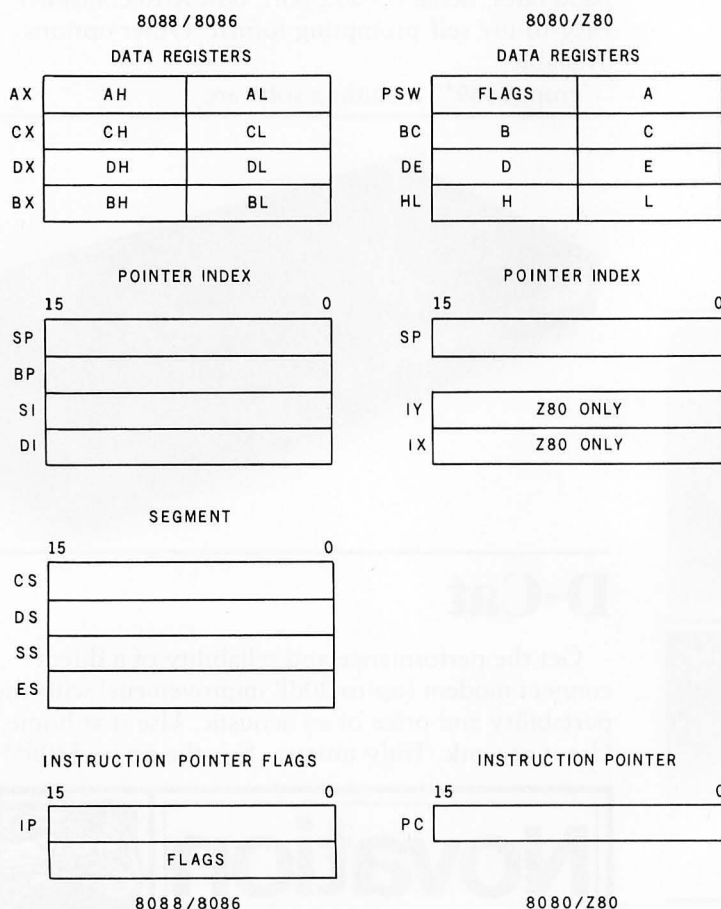


Figure 5: Register-usage mapping between the two processor families is shown. Note that the IX and IY registers of the Z80 do not exist for the 8080/8085. More registers are in the 8088/8086 than in either the 8080 or Z80. Observe that the 8088/8086 segment registers have no parallel in the 8080/Z80 family. The 8080 family has a 16-bit address bus that allows addressing of 65,536 bytes of memory. While the internal registers of the 8088/8086 family also have 16 bits, the external address bus has 20 bits, formed by extending a segment register with 4 low-order bits of 0, and adding it to a 16-bit address from another register.

a role in string operations. All destinations of string operations use the DI register in calculating addresses, and are taken relative to the ES register.

Besides the four segment registers, the 8086 has the following:

- Four general-purpose registers: AX, BX, CX, and DX. Each is addressable as a 16-bit register or as two 8-bit registers. When addressed as 8-bit registers, the pairs are called: AH, AL; BX, BL; CH, CL; and DH, DL. The general-purpose registers hold the intermediate results of operations.
- Four pointer and index registers; these locate data within a specified segment of memory. SP is the stack pointer, BP the base pointer, SI the source index, and DI the destination index.
- One program counter.
- One 16-bit flag register (program status word, or status register) containing nine flags.

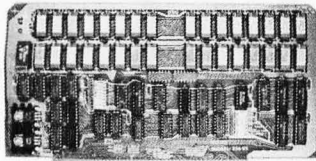
Now that we've looked inside the 8086, we can take a look at how register usage in the 8080/Z80 processor family corresponds generally with that in the 8086. Figure 5 shows the sets of registers alongside each other so that you can see the general correspondence clearly. Note that the IX and IY registers of the Z80 do not exist for the 8080/8085. On the other hand, the alternate register set of the Z80 (not shown) as well as the I and R registers have no parallel in the 8086 register set; operations involving these will require special attention from the programmer after the conversion is done.

Clearly, the 8086 has more registers than either the 8080 or Z80. Since the 8086 also has a more powerful instruction set, translation should be possible with minor restrictions.

Complications

Since all CP/M-80 programs had to exist in a 64K-byte region, there should be little trouble fitting a translated program into a 1-megabyte (1,048,576-byte) region. If you're translating from the Z80, however, things are complicated slightly

S-100 Boards from S. C. Digital



256K DYNAMIC RAM

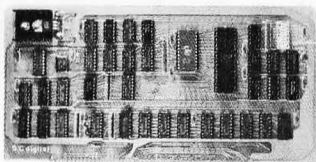
features: **Model 256KE**
 ● 16 or 24 bit address. ● 8/16 bit wide data ● Transparent refresh with unlimited DMA, immune to Wait States, halts, resets. ● Fast access time 180nsec from Smemr or Psync high, will run with Z80, Z8000 to 6mhz, 8080, 8085, 8086 to 8mhz without Wait States. ● Accepts 4116, 4164's.

64K DYNAMIC RAM 'Uniselect: 2'

features: **Model 64KUS**
 ● 16 or 24 bit address. ● 8 bit data. ● Bank select by SW settable port, bits in two blocks. ● Two 32kb (128kb) addressing. ● Transparent refresh - same as M:256KE. ● Fast access time - 220nsec, will run with Z80, Z8000 to 4mhz, 8080, 8085, 8086, 8088 to 5mhz without Wait States. ● Can be configured to various multiusers OS's. ● Expandable to 256KB using 4164's.

32K STATIC RAM 'Uniselect: 3'

features: **Model 32KUS**
 ● Fully Static using 2k by 8 NMOS chips. ● 16 or 24 bit address. ● 8/16 bit wide data. ● Bank Select by port and bit in 32K block. ● Two 16K block addressing with window capability in 2k increments. ● EPROM can be mixed with RAM. ● Fast access - 250nsec from address valid - will run with Z80, Z8000 to 4mhz, 8080, 8085, 8088, 8086 or 68000 to 8mhz without Wait States. ● Provision for Battery Backup.



Z80 CPU Board

features: **Model CPU1 Z80**
 ● 2 or 4mhz clock. ● Jump on Reset. ● 8 levels of prioritized vectored interrupts.

I/O, Memory Interface 'Interface: 1'

features: **Model 3SPC**
 ● 3 serials using UART, RS-232C or 20ma current loop. ● 1 Parallel I/O with hand shakes. ● 4k Ram, 4k EPROM (not supplied). ● Built in Kansas City Audio Cassette interface. ● Baud rate generator from 19.2kbaud to 110 baud.

2K Z80 Monitor Program available for M:3SPC

features: many routines including breaker points, cassette record and play back . . . etc. Comes in 2 EPROMs and 1K RAM.

All boards conform to IEEE696/S100 specifications, fully socketed, screened legends, masks, Gold contacts. Guaranteed One Full year.

Model	Prices	with	
256KE	\$795	256KB	AGT
256KE-128	\$635	128KB	AGT
64KUS-B4	\$395	64KB (4164's)	AGT
64KUS	\$395	64KB (4116's)	AGT
64KUS-16	\$285	16KB	AGT
32KUS	\$399	32KB	AGT
32KUS-1B	\$269	16KB	AGT
32KUS-N	\$149	no memory	AGT
CPU1-Z80	\$219	with interrupt	AGT
CPU1-Z80-K	\$149	no interrupt	Kit
3SPC	\$229	with cassette	AGT
3SPC-KC	\$159	with cassette	Kit
2K Monitor	\$ 55	with 1K Ram	

Delivery is within 3 working days. MC, Visa or COD orders accepted. Illinois residents add 5 1/4% sales tax.

O.E.M. & DEALER PRICING AVAILABLE

S. C. DIGITAL

P.O. Box 906, Aurora, Illinois 60507
 Phone: (312) 897-7749

because its extensive instruction set has to be mapped to the 8086 instructions, as well as the 8080 registers.

The 8086 does, however, have a rich instruction set that covers all the bit instructions and most of the block instructions. The problems encountered in mapping instructions and registers can be formalized and solved by using a variety of software tools.

To convert a source file running under CP/M-80 so as to assemble it with an 8086 assembler requires either the use of a code translator or a considerable effort with a program editor. At least three commercial products come under the translator category:

- XLT86 from Digital Research Inc.
- TRANS86/ACT86 from Sorcim
- The Z80-to-8086 translator from Seattle Computer Products

Intel also has a translator, CONVERT 86, but it is sold only as part of a large software package intended for its OEM customers and doesn't run under CP/M. In addition, some reports indicate that Microsoft may soon bring out a translator.

Two issues must be addressed when converting from a CP/M-80 source:

1. Where is the source coming from?

Z80 instruction mnemonics?

- (a) Zilog
- (b) TDL (Technical Design Labs)
- (c) Intel (Digital Research with macros)
- (d) Sorcim/ACT-80

or

Straight 8080/8085 code with no macros?

2. Where is the resultant code going?

CP/M-86 or MS-DOS?

The three translators under review do not always approach these issues in the same way. Furthermore, they all handle register mapping somewhat differently. We'll look at the

translators now one at a time. Then we'll see how they translated the same CP/M-80 listing. With the listing in hand, we'll see how each of the translators handled register mapping. Finally, we'll turn to the subject of how the two different 16-bit operating systems affect program translation.

XLT86 from Digital Research

Digital Research's XLT86 takes standard 8080 source code in a format compatible with ASM, MAC, or RMAC assemblers and converts the 8080 source code to 8086 source code in a format compatible with ASM86 operating under either CP/M-80 or CP/M-86. Since XLT86 is written in PL/I-80, the translator can run either stand-alone under CP/M-80 or for cross development under VAX/VMS. It produces optimized 8086 code in a five-phase, multipass process, doing global data-flow analysis to determine optimal register usage.

Although macro definitions are not supported, conditional-assembly directives are. The *XLT86 User's Guide* suggests that if you want macro expansion, you can use a pass through MAC or RMAC to produce a PRN file that can be edited (removing the first few columns of generated hexadecimal code) to produce an expanded source file for input acceptable to XLT86. XLT86 does not recognize Z80 instructions. XLT86 passes repeat loops through to the 8086 source code.

XLT86 analyzes the source program in its entirety, determining the block structure and the register/flag usage. Working from this information, it translates the code to 8086 assembler code in an optimized way. The decision algorithm for each instruction type is given in a section of the manual to allow the user to see what happens in each situation.

Register mapping generally follows the correspondence shown in figure 4, with a loose relationship between the 8086 AX and the 8080 PSW; the exact relationship is determined from register usage at translate time.

Many run-time options are available to control the translation pro-

cess, both on the command line and embedded in the 8080 source text. The options control the disks that the work and output files are on, whether the block-analysis information is output to disk, whether code and data segments are to be intermixed or kept separate, and whether the condition flags are active on exiting from subroutines.

XLT86 is a sophisticated program that does a reasonable job of optimizing the translation of 8080 source code to 8086 source code. BDOS calls from CP/M-80 are mapped into BDOS calls that are compatible with CP/M-86.

XLT86 has special features for handling translation of conditional JMP and CALL instructions in 8080 source code. In the 8080 instructions, JMP and CALL instructions are capable of reaching any address within the 64K-byte region. The 8086 conditional JMP instructions can reach only 128 bytes on either side of the IP (Instruction Pointer) register. XLT86 examines the target of the con-

ditional JMP. If the target cannot be reached, XLT86 changes the sense of the conditional JMP and skips over a long JMP to the target address. Since there are no conditional CALL or RET instructions in the 8086, the sense of the condition is changed and a short conditional JMP is performed

The 8086 can access up to 1 megabyte of memory and as many as 65,000 input/output ports.

to skip over an unconditional CALL or RET.

As noted earlier, the segment registers allow for separation of code and data regions. To reference data, you have to tell the 8086 whether data is in the code segment (CS) or the data segment (DS). For the Digital Research ASM86 assembler, the Offset directive handles this chore.

XLT86 examines an expression and determines the proper segment for the particular instruction.

XLT86 does have limits on the size of the 8080 source files that it can translate because the flow-analysis information must be in memory. In a 64K-byte CP/M system, the maximum source file that can be translated is approximately 6K bytes, depending on the structure of the program. Nothing is said in the manual about being able to deal with modular code using RMAC and external references. This implies that the entire source program must be converted at once, limiting the size of the program that can be translated by using XLT86 to 6K bytes.

In summary, if you're starting from 8080/8085 assembly code written for ASM or MAC and you want to go to CP/M-86, and if the source program does not exceed 6K bytes, XLT86 is the most useful translator. Code written for Z80s using MAC requires careful examination after the translation process to make sure that no

For your students' first course in computing... pick the program that's first in delivering actual computer skills

COMPUTER POWER

Michael Moshell, Project Director

A hands-on approach to computer literacy

Students do a lot more than *read* about computers. They also...

- become *capable beginning computer programmers* who can program the computer for business, personal, and school use
- strengthen their *problem-solving skills and attention spans*
- sharpen their *planning skills* by analyzing sub-tasks
- learn about *career opportunities* in the computer field

Classroom-tested materials that are easy to use

The *student text* and *courseware* reduce computer anxiety by offering lively instruction featuring graphics and animation. And the *Teacher's Manual* offers thorough, step-by-step guidance to the new-to-the-computer teacher.

Text, 224 pp. (65773-4) \$7.50*
Teacher's Manual, 372 pp. incl. 7 disks (65774-2) \$385.00*

To learn more, just fill out and mail the coupon. **Or dial toll-free: 800-223-4180. (From New York State, call collect: 212-997-2646.)**

Computer Power
Gregg/McGraw-Hill
1221 Avenue of the Americas
New York, N.Y. 10020



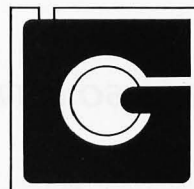
Name _____
Title _____
Institution _____
Street Address _____
City _____
State _____ Zip _____
Best Time to Call _____
Area Code & Phone # _____

BYTE 6/82.

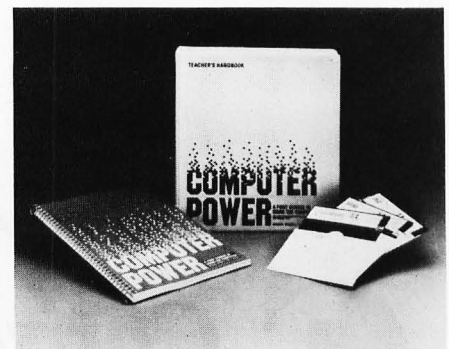
Note: *For better service, Gregg's educational courseware is available under a license agreement, rather than through sale.

To take full advantage of **Computer Power**, you need the following equipment:

APPLE II Plus micro-computer, equipped with a video monitor (color or black and white), UCSD APPLE Pascal-language card, and game paddles.



GREGG SOFTWARE



flags were inadvertently changed. You have to expand macros before using XLT86 unless the number of invocations was small. In that event, expanding the macros by hand with an editor might be just as easy.

Sorcim's TRANS86

TRANS86 is an 8080/ACT80-to-8086 translator. It takes 8080 or ACT80 source code as input and creates a file compatible with the input to ACT86, an 8086 assembler.

The output of TRANS86 is incompatible with any assembler other than ACT86. The ACT86 mnemonics are different enough so that, unless the programmer has a sophisticated text processor and the talent (or patience) to do a great deal of text manipulation, TRANS86 should be used only with ACT86.

Both TRANS86 and ACT86 run on either 8080 or Z80 processors under CP/M-80, MP/M, or CDOS with a minimum of 24K bytes of RAM (random-access read/write memory). TRANS86 consists of an executable file, an overlay file, and a translation table. The input assembly source code must be in a form acceptable to the standard CP/M-80 assemblers (ASM, MAC, or RMAC), or to ACT80, Sorcim's Assembly Code Translator for 8080/Z80 processors.

Translation occurs on an instruction-by-instruction basis with some optimization rules applied to conditional jumps. There appears to be no limit as to the size of the source file that can be translated. A file is produced on the same disk as the source file with the same name and an .ASN extension. If a file by that name already exists, the user is asked if the file should be deleted or if the program should be aborted.

TRANS86 flags the following Z80 instructions as errors:

ACT80 code	Zilog/Mostek equivalent code
Mov A,R	Ld A,R
Mov R,A	Ld R,A
Mov A,I	Ld A,I
Mov I,A	Ld I,A
In,C reg	In reg,(C)
Inir	Inir
Otir	Otir
Rld	Rld
Rrd	Rrd

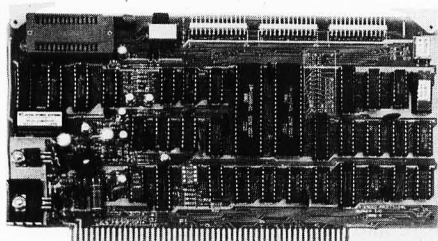
TRANS86 supports macros in the ACT80 format. Although TRANS86 acts on macros in the MAC format, there is no guarantee that the macros will expand correctly. The user is cautioned to examine the result of a

macro expansion to determine if the sense of the macro has been maintained. Examples are given of some macros that work and some that do not. The *TRANS86 User's Reference Manual* includes a section that gives hints on how to hand-optimize the output of TRANS86; specifically, accumulator indirect loads through the DE and BC registers, 8080 conditional jumps, and Z80 block instructions.

Another section describes the differences between ACT80 and Z80 mnemonics. This information allows the programmer to manually convert assembly source code to a form acceptable to TRANS86. The ACT80 instruction set has some ASM-style instructions, some Z80-style instructions, and some instructions that are unique to the ACT80 assembler. If the source code is written in 8080 ASM mnemonics, TRANS86 will process it and output ACT86 assembler code. The 8080 instruction SPHL, however, was translated incorrectly in the current version of TRANS86.

Another section in the manual contains suggested constructions that can be manually entered to deal with Z80 op codes that are flagged as errors. Block input/output instructions and input/output through register C are described in detail.

NEW FROM EXTENDED PROCESSING THE BURNER I/O FOR THE S-100 BUS



The BURNER I/O has a complete EPROM programmer, two serial ports, one parallel I/O port with handshaking and memory management.

Programmer features:

- Most EPROM types from 2704 thru 2764 and 2508, 2516 and TMS2716.
- CPM compatible software supplied in EPROM that can be easily written on a diskette.

- Programming socket is zero insertion force type.
- Programming voltages generated on board.
- Programmer is totally I/O mapped.

I/O features: (serial)

- 2 fully independent RS-232 serial ports.
- RS-232 data ready supported.
- Each serial port has independent baud rate generators that are software programmable from 50 to 19,200 baud.
- Serial ports may be polled and/or interrupt driven.

I/O features: (parallel)

- Independent 8 bit output, input and status flags.
- All I/O including flags are latched.
- In addition, there are 4 direct sense lines.

Memory management features:

- Controls the S-100 address lines from A16-A23.
- Uses output instruction to load the address. Normally 40H.

We are offering this board with all options, or just the portions that are needed may be purchased. Regardless of which version that is purchased, documentation for the entire board is supplied. All combinations are assembled and tested.

Option A: Complete board with programmer,

I/O and memory management. \$324.95

Option B: Programmer only. \$199.95

Option C: I/O only. (2S + P) \$199.95

Option D: Option B and C. \$299.95

Option E: Memory management only \$150.00

Memory management may be added to

options B or C for \$25.00.

AVAILABLE FROM

Microbyte Computer Systems

2798 So. Bascom, San Jose, CA 95124

(408) 377-4685

DEALER INQUIRIES INVITED

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.

NEW FROM NETRONICS AUTO-PATCH HARD DISK

With plug-in multi-user ports
Automatically Installs Itself Into
Your Present CP/M® 2.2 Operating
system & Floppy Disk Hardware.

It's Exclusive!

6 megabytes . . . \$2995.00 12 megabytes . . . \$3495.00



What's the big concern of S100 owners when they consider adding Hard Disks? They worry that it will be difficult to install, that it won't be compatible with their present software and hardware, and that it may cause down-time on their S100 system.

Worry no more — Netronics new AUTOPATCH Hard Disks Systems are here. AUTOPATCH installs in just one-two-three: (1) plug in the hard disk S100 card; (2) run three short programs supplied on disk; (3) disable the boot on your floppy controller and enable the boot on your hard disk controller (this step not required if you wish to continue to boot to your floppy drives).

And that's it: The AUTOPATCH feature automatically finds the end of your existing BIOS and then self relocates and patches itself into the existing BIOS. A virgin copy of CCP and BIOS are loaded into memory, a customized SBOOT is added to the front of CCP and the whole memory image is written to the reserved tracks on your hard disk. You can add up to 4 hard disks to the controller supplied. The new BIOS will automatically rename any old devices as B: and C: and define the hard disk as drive A:. All with the lift of one finger!!! If your BIOS is large you may have to re-assign your system down 1 or 2 k. If this is necessary the AUTOPATCH program will prompt you to do so.

AUTOPATCH Hard Disk Systems are available in 6 and 12 megabyte models. Included in the system: 6 or 12 megabyte Hard Disk Drive . . . Controller for up to 4 Hard Disk drives . . . S100 Hard Disk card with provisions for adding 8 additional I/O ports to be used when adding a multi-user operating system . . . Power Supply . . . Deluxe Steel Cabinet . . . All necessary cables . . . AUTOPATCH Programs supplied on either 8" or 5 1/4" IBM formatted single density diskettes (specify style required) . . . Complete installation instructions . . . Fully wired and tested, ready to go.

SPECIFICATIONS

Unformatted Recording Capacity: 6.4 or 11.6 MB . . .
No. of tracks: 612 or 1380 . . . Data Transfer Rate: 3 ms
. . . Bytes/sector format: 512 . . . Communication Port:
DO (other ports available on special order) . . . Programs
supplied on 5 1/4" or 8" single density IBM formatted
diskettes (North Star CP/M® version available
on special order)

10 DAY MONEY BACK OFFER

Continental U.S.A. Credit Card Buyers Outside Conn.

CALL TOLL FREE 800-243-7428

To Order From Connecticut Or For Tech. Assist.
Call (203) 354-9375

NETRONICS R&D LTD. Dept.

333 Litchfield Road, New Milford, CT 06776

Please send the items checked below:

AUTOPATCH/6 Hard Disk System . . . \$2995.00

AUTOPATCH/12 Hard Disk System . . . \$3495.00

Additional 6-megabyte drive with power supply,
cabinet, cables and necessary software . . . \$1995.00

Additional 12-megabyte drive with power supply,
cabinet, cables and necessary software . . . \$2495.00

All plus \$15.00 P&I (postage & insurance). For Canadian orders, double the postage (\$30.00). Conn. res. add sales tax.

Total Enclosed \$ _____

Personal Check Cashier's Check/M.O.

VISA MasterCard (Bank No. _____)

Acct. No. _____ Exp. Date _____

Signature _____

Print Name _____

Address _____

City _____ State _____ Zip _____

(a)

	8080/8085	TRANS86	8086/8088 Seattle	XLT86
8-bit registers	A	AL	AL	AL
	B	CH	CH	CH
	C	CL	CL	CL
	D	DH	DH	DH
	E	DL	DL	DL
	H	BH	BH	BH
	L	BL	BL	BL
16-bit register pairs	PSW	AX(1)	AX	AX
	BC	CX	CX(2)	CX(2)
	DE	DX	DX(2)	DX(2)
	HL	BX	BX	BX
16-bit register pairs	SP	SP	SP	SP
	PC	IP	IP	IP

(1) TRANS86 does not preserve 8080 byte order on the stack.

(2) The Seattle translator uses SI on loads from memory and DI for stores to memory. TRANS86 and XLT86 do a register exchange between BX and the appropriate register to allow indirect addressing through BX, then a register exchange to fix up BX and the appropriate register.

(b)

	Z80	TRANS86	8086/8088 Seattle	XLT86
8-bit	R	(3)	(3)	(3)
	I	(3)	(3)	(3)
16-bit	IX	DI	(4)	(5)
	IY	SI	(4)	(5)
alternate registers set	BC'	(5)	(4)	(5)
	DE'	(5)	(4)	(5)
	HL'	(5)	(4)	(5)

(3) Since the 8086 does not have equivalent registers, none of the translators support these registers. However, they can be mapped to a memory location by the programmer.

(4) Seattle's TRANS86 handles these registers by generating memory references to storage locations defined by the programmer.

(5) Although these registers are not mapped by the translators, the programmer can define storage locations and deal with them through macro definitions.

Figure 6: 8080/8085/Z80-to-8086/8088 register mapping. Figure 6a shows 8080/8085-to-8086 register mapping by the three translator programs. Figure 6b shows Z80-to-8086 register mapping by the three translator programs.

- In general, when the 8080 does 16-bit arithmetic, only the carry bit is affected; this is definitely not so in the 8086.

- The Z80 and 8086 do string and block operations differently; the 8080 has no primitive block operations at all.

- As noted earlier, the segment registers in the 8086 allow addressing of up to 1 megabyte; no corresponding registers exist in either the 8080 or Z80.

- Registers used for indirect memory references in the 8080/Z80 are different from the corresponding mapped registers in the 8086.

- Conditional jumps in the 8080/Z80

can reach anywhere in its address space; conditional jumps in the 8086 can reach only 128 bytes on either side relative to the IP register.

- No conditional calls in the 8086 correspond to the conditional calls of the 8080/Z80.

Listing 1 makes it apparent that the three translators treat most instructions the same way, allowing for the differences in the target instruction set. The following comments highlight the differences found.

The only incorrect translation is TRANS86's rendering of the SPHL instruction. The transfer is in the wrong direction. The comment field of the instruction was wrong in the

MTI stocks 'em all for faster delivery.

Ask about our "QED" discounts.
VISA and MasterCard orders accepted.

VIDEO TERMINALS

	MTI Price
VT100 DECscope	\$ 1595
VT18X Computing option	2395
VT101 DECscope	1215
VT131 DECscope	1785
VT132 DECscope	1995
ADM 3A (dumb terminal)	595
ADM 5 (dumb with visual attributes) .	645
ADM 31 (two page buffer)	1095
ADM 21, 24, 32, 36, 42	*
Hazeltine Esprit	645
Hazeltine Executive 80 Model 20	1495
Hazeltine Executive 80 Model 30	1715
1410 (Hazeltine dumb terminal)	575
1421 (Consul 580 & ADM 3A comp.) .	595
1500 (dumb terminal)	825
1520 (buffered, printer port)	1105

RETRO-GRAPHICS TERMINALS

VT100 with graphics pkg.	3250
VT125 (DEC graphics)	3280
ADM 3A with graphics pkg.	1795
ADM 5 with graphics pkg.	1845

300 BAUD TELEPRINTERS

LA 34-AA DECwriter IV	1095
LA 36 DECwriter II	1095
Diablo 630 RO	2295
Diablo 630 KSR	2695
Diablo 1650 KSR	2635
TI 743 (portable)	1190
TI 745 (port/built-in coupler)	1485
TI 765 (port/bubble/b.i. coupler)	2595

600 BAUD TELEPRINTERS

Epson MX-80	645
TI 825 KSR impact	1570
TI 825 KSR pkg.	1795

1200 BAUD TELEPRINTERS

Epson MX-100	995
LA 120 RA (receive only)	2095
LA 120 AA DECwriter III	2295
TI 783 (portable)	1645
TI 785 (port/built-in coupler)	2270
TI 787 (port/internal modem)	2595
TI 810 RO impact	1545
TI 810 RO pkg.	1795
TI 820 RO impact	1850
TI 820 RO pkg.	2025
TI 820 KSR impact	2025
TI 820 KSR pkg.	2195
Lear Siegler 310 ballistic	1945

2400 BAUD

Dataproducs M200 (2400 baud)	2910
------------------------------------	------

DATAPRODUCTS LINE PRINTERS

B300 (300 LPM band)	5455
B600 (600 LPM band)	6930
B1000 (1000 LPM band)	11330
BP1500 (1500 LPM band)	19700

ACOUSTIC COUPLERS

A/J A242-A (300 baud orig.)	242
A/J 247 (300 baud orig.)	315
Vadic VA 3413 (1000/1200 orig.)	845
Vadic VA 3434 (1200 baud orig.)	845

MODEMS

GDC 103A3 (300 baud Bell)	395
GDC 202S/T (1200 baud Bell)	565
VA 3212 (Bell 212A comp.)	825
VA 103 (300 baud modemphone)	235
VA 3451 (orig/ans triple modem)	885
VA 3455 (1200 baud orig/ans.)	770
VA 2450 (Bell 201 comp.)	725

CASSETTE STORAGE SYSTEMS

Techtran 816 (store/forward)	735
Techtran 817 (store/for/speed up)	915
Techtran 818 (editing)	1225
Techtran 822 (dual)	1640

FLOPPY DISK SYSTEMS

Techtran 950 (store/forward)	1395
Techtran 951 (editing)	1995

* Please call for quote.



Applications Specialists & Distributors
Computer Terminals, Peripherals & Systems

New York:

516/621-6200, 212/767-0677, 518/449-5959

Outside N.Y.S.: 800/645-6530

New Jersey: 201/227-5552

Ohio: 216/464-6688

Listing 1: The original 8080 code, the Sorcim TRANS86 translation, the Seattle translation, and the XLT86 translation.

Original 8080 code	TRANS86	Seattle	XLT86
Stax B	Xchg BX,CX Sto AL,[BX] Xchg BX,CX	Mov DI,CX Stob	Mov SI,CX Mov [SI],AL
Inx B	Inc CX	Lahf Inc CX Sahf	Lahf Inc CX Sahf
Dad B	Add BX,CX	Lahf Add BX,CX Rcr SI Sahf Rcl SI	Lahf Add BX,CX Sahf
Ldax B	Xchg BX,CX Ld AL,[BX] Xchg BX,CX	Mov SI,CX Lodsb	Mov SI,CX Mov AL,[SI]
Stax D	Xchg BX,DX Sto AL,[BX] Xchg BX,DX	Mov DI,DX Stob	Mov SI,DX Mov [SI],AL
Ldax D	Xchg BX,DX Ld AL,[BX] Xchg BX,DX	Mov SI,DX Lodsb	Mov SI,DX Mov AL,[SI]
Dad H	Add BX,BX	Lahf Add BX,BX Rcr SI Sahf Rcl SI	SHL BX,1
Dec H	Dec BX	Lahf Dec BX Sahf	Dec BX
Inr M	Incb [BX]	Inc B,[BX]	Inc Byte Ptr 0[BX]
Mvi M,3	Stob #3,[BX]	Mov B,[BX],3	Mov Byte Ptr 0[BX],3
Dad SP	Add BX,SP	Lahf Add BX,SP Rcr SI Sahf Rcl SI	Add BX,SP
DCx SP	Dec SP	Lahf Dec SP Sahf	Dec SP
Mov M,A	Sto AL,[BX]	Mov [BX],AL	Mov Byte Ptr 0[BX],AL
Rnz	Jz :G1 Ret :G1:	Jnz Ret	Jz L 1 Ret L 1:
Jnz Help	Jz :G2 Jmp Help :G2:	Jnz Help	Jz L 2 Jmp Help L 2:

Listing 1 continued on page 340

Listing 1 continued:

Cnz Help	Jz :G3 Call Help :G3:	Jz X000 Call Help X000:	Jz L 3 Call Help L 3:
Xthl	Mov BP,SP Xchg BX,[BP]	Pop SI Xchg SI,BX Push SI	Mov BP,SP Xchg BX,[BP]
Pop PSW	Pop AX Sahf	Pop AX Xchg AH,AL Sahf	Pop AX Xchg AL,AH Sahf
Push PSW	Lahf Push AX	Lahf Xchg AH,AL Push AX Xchg AH,AL	Lahf Xchg AL,AH Push AX Xchg AL,AH
Sphl	Mov BX,SP	Mov SP,BX	Mov SP,BX
(Z80 opcodes)			
Djnz Label	Dec CX Jnz Label	Dec CH Jnz Label	-----
Cpir	Xchg BX,DI Cld Repnz Scasb Xchg BX,DI	UP Mov DI,BX Repnz Scab Mov BX,DI	-----
Ldir	Xchg BX,SI Xchg DX,DI Cld Repnz Movsb Xchg BX,SI Xchg DX,DI	Up Mov SI,BX Mov DI,DX Rep Movb Mov DX,DI Mov BX,SI	-----
Push Ix	Push DI	Push [Ix]	-----
Pop Iy	Pop SI	Pop [Iy]	-----

translated text, also. We therefore suspect that the program has a minor bug with regard to this particular instruction.

Since this test program was nonsense as far as logical program flow is concerned, XLT86 was at a loss to determine the active registers, and sometimes chose simpler instructions than it would have in a real program. The results for XLT86 could be different in different situations.

The conditional jumps for TRANS86 and XLT86 can vary, depending on the distance of the target label from where the jump is.

The conditional return in Seattle's translator references a label called RET. This refers to any RET within 128 bytes on either side of the statement. This is one reason why Seattle's

translator should be used with ASM86; no other assembler will take advantage of this feature.

Note that TRANS86 and the Seattle translator treat the DJNZ instruction differently. TRANS86 uses a 16-bit register, CX, and the Seattle translator uses CH, an 8-bit register. A warning message comes out of the Seattle translator reminding the programmer that DJNZ does not affect the flags in the Z80 but that this sequence of instructions will affect the 8086 flags.

Register Mapping

Figure 6 shows a detailed, side-by-side comparison of the differences in register mapping performed by the three translators. Figure 6a deals with the 8080/8085-to-8086 mapping;

figure 6b, with Z80-to-8086 mapping.

As the notes there state, TRANS86 does not preserve 8080 byte order on the stack.

The Seattle translator uses SI on loads from memory and DI for stores to memory.

TRANS86 and XLT86 do a register exchange between BX and the appropriate register to allow indirect addressing through BX, then a register exchange to fix up BX and the appropriate register.

Since the 8086 does not have some of the registers of the Z80, the translators can't support them. The programmer can, however, map those registers to a memory location.

TRANS86 generates memory references to storage locations supplied by the programmer to take care of the Z80's IX, IY, BC', DE', and HL' registers.

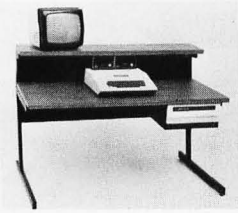
Summing Up the Translators

A general view is that Sorcim's TRANS86 is a useful product if the original source is in 8080 or ACT80 form and the user has ACT86 as a target 8086 assembler. The register and flag usage appear to be a little looser than for the other two programs. This requires more knowledge and more involvement from the programmer to make sure that the sense of the translated code is maintained. No limitations exist as to the size of the source file and macros are supported if the input is in ACT80 format. Sorcim's TRANS86 is sold separately from ACT86, but they should be used together.

The Seattle Computer Products' Z80-to-8086 translator is a straightforward code translator that uses Zilog mnemonics and runs only on Z80-based processors. There appear to be no limitations as to the size of the source program that may be translated since the program translates one instruction at a time. Register and flag usage are very conservative, protecting the source architecture as much as possible and providing warnings when potential problems could arise. The converted program has more of a chance of working the first time than a less conservative translation would have.

CHOOSE...

Choose an Apple Desk



A compact Bi-Level desk ideal for the Apple computer system. This 42" x 29½" desk comes with a shelf to hold two Apple disk drives. The top shelf for your TV or monitor and manuals can also have an optional paper slot to accommodate a printer. It is shown here with the optional Corvis shelf which will hold one Corvis disk drive. The Corvis shelf is available on the 52" x 29½" version of the Apple desk.

Choose a Micro Desk



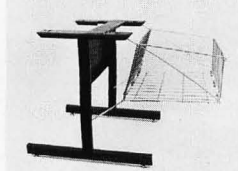
The Universal Micro desk accommodates the S-100 type microcomputers. The desk is available in four sizes: 17.75 inch, 19.06 inch, and 20.75 inch wide openings with 24 inch front-to-rear mounting space. The fourth size is a 20.75 inch wide opening with a 26.50 inch front-to-rear mounting space.

Choose a Mini Rack



Mini racks and mini micro racks have standard venting, cable cut outs and adjustable RETMA rails. Choose a stand alone bay or a 48", 60", or 72" desk model in a variety of colors and wood tones. A custom rack is available for the Cromemco.

Choose a Printer Stand



The Universal printer stand fits the:

Centronics 700's	Diablo 1600's & 2300's
Dec LA 34	T.I. 810 & 820
NEC Spinwriter	Okidata Slimline
Lear Siegler 300's	Anadex 9500's

Delivery in days on over 200 styles and colors in stock. Dealer inquiries invited.

ELECTRONIC SYSTEMS FURNITURE COMPANY

17129 S. Kingsview Avenue
Carson, California 90746
Telephone: (213)538-9601

Once the program is working correctly, the programmer can reedit the 8086 source to trim out the unnecessary register saves and flag manipulations to speed up the code and make it smaller. If the source code is in Zilog mnemonics, or if a filter program is available to convert to Zilog mnemonics, we recommend the pair of programs from Seattle Computer Products. The Seattle translator and its ASM86 assembler should be used together for optimum benefit.

XLT86 from Digital Research is a program that takes 8080 source code and converts it into 8086 source code in an intelligent manner using data-flow-analysis techniques. It will produce better code than either of the other two translators if two conditions are met: (1) no Z80 instructions are used, and (2) the source code is not bigger than 6K bytes (assuming the CP/M system is a 64K-byte system). XLT86 should be used with ASM86 under either CP/M-80 or CP/M-86.

MS-DOS versus CP/M-86

In the trade press and various advertisements, we see claims that conversion of CP/M-80 programs to MS-DOS (IBM PC-DOS, Lifeboat Associates' SB-86, Seattle Computer Products' 86-DOS) is as easy, or even easier, than from CP/M-80 to CP/M-86. In fact, some differences are seen in interfacing to the two 16-bit operating systems. With the assumption that the program had to operate in a 64K-byte region under CP/M-80, we will bypass memory-allocation questions. The remaining issues that have to be addressed are:

- How does a program gain access to operating system resources?
- How are file control blocks used to manipulate files?

Gaining access to the operating system under CP/M-80 requires placing the function code in the C register, placing the information address in the DE registers, and calling location 05 hexadecimal, the CP/M-80 entry point. If the system

call returns a value, it returns the value to the A register. A so-called warm boot under CP/M-80 is accessed by jumping to location 00 hexadecimal, which reads in the operating system and resets the disk system.

Now let's look at similar functions under CP/M-86 and MS-DOS.

Gaining Access to CP/M-86

Gaining access to CP/M-86 requires placing the function code in the CL register, placing the byte parameter in the DL register or placing the word parameter in the DX register, placing the data segment in the DS register (the data segment is usually not changed for a converted program), and executing a software interrupt, INT #224. The result is returned in the AL register if it is a byte value; if the result is a word value, it is returned in both the AX and BX registers. Double-word values are returned with the offset in the BX registers and the segment in the ES register. Conversion of programs from CP/M-80 to CP/M-86, then, requires replacing the call to location 5 with the software interrupt INT #224.

Another necessary change involves the warm boot. Under CP/M-80, the warm boot may be accessed by a system call with a function code of 0 for a jump to location 0. CP/M-86, however, does not support the jump to location 0. As a result, you must change this program exit in the translated program if the program is to run correctly.

Provided that the call to location 5 is replaced with INT #224, that the warm boot change is made, and that the registers are mapped correctly, there should be little problem in getting the translated program to access the CP/M-86 system functions.

Gaining Access to MS-DOS

Although MS-DOS has a "preferred" mechanism through a software interrupt, INT #33, for accessing the system, an additional mechanism is provided for "pre-existing" programs that is compatible with CP/M-80 calling conventions, at least for functions in the range of

0-36. As far as system calls within the allowed function range are concerned, the programmer doesn't have to do anything to translated programs to get them to run under MS-DOS other than to correctly map the registers.

MS-DOS also supports the warm boot function of CP/M-80. A jump to location 0 under MS-DOS executes a software interrupt, INT #32, which is functionally a program end and the normal way to exit from a program.

Manipulating File Control Blocks

The file control block used in CP/M-80 consists of a 36-byte block, which describes the disk drive on which to find or create the file, the file name, and information relating to which record of the file is desired.

At least so far as normal file-access requests are concerned, both MS-DOS and CP/M-86 treat this block of information the same.

System-level information is quite different in the two cases, and pro-

grams that look at system bytes within the file control block need to be changed for MS-DOS to function correctly. The MS-DOS file control block has many more features, including the date the file was created

The problems encountered in mapping instructions and registers can be formalized and solved by using a variety of software tools.

or last updated, the logical record size, and the file size. These system-information bytes are in areas within the file control block that application programs normally do not access. Nevertheless, converting programs to make use of MS-DOS file control blocks should take little effort.

Conclusion

There is, in fact, little if any difference in the difficulty of translating sound CP/M-80 programs to CP/M-86 or MS-DOS. With CP/M-86, the programmer will have to make minor changes to gain access to the operating system. With MS-DOS, the programmer will have to make minor changes to handle the extra features of the MS-DOS file control blocks.

Next month, we will make further comparisons between MS-DOS and CP/M-86. We will include some benchmarks made with the Compu-pro 8085/8088 dual-processor S-100 system. We will report not only the results of running programs under both CP/M-86 and MS-DOS on the same 8088 in the same machine, but also the results of running the same programs under MS-DOS running Emulator-86 on the same 8088 in the same machine. Although that may sound more like a cat chasing its own tail than a test of operating systems, we will try to keep it all straight. ■

Bare Bones APPLE II

w/o Keyboard
w/o Pwr. Supply

\$450. 48k RAM

Microswitch: Power Supply: APPLE
Keyboard / Pad: 5amp.: Reference Manl.

\$95.00 : \$124.00 : \$18.00

SPECIALS

8255 - \$5.95
8748-8 - \$31.00
3341PC - \$2.00
MM5060 - 35c
MC6800 - \$7.75
MC6802 - \$14.95
MC6850 - \$4.50
MC6821 - \$4.95

REAL-TIME CLOCK CALENDAR (MSM 5832)

Features: Mono Metal Gate CMOS IC
Time: Month, Date, Year, & Day of Week
Bus Oriented
1 Bit Data Bus
1 Bit Address
R/W Hold Selec.
Inter Signal
32 768KHz xtal Control
5v Pow Sup
Low Power Dissipation

\$7.45
XTAL \$2.85

'82' I.C. MASTER \$59.95

GLOBAL LPK-1:
Logic Probe Kit - complete nothing extra to buy. Min. pulse width 300nsec.
\$18.95

DISKETTE SALE!! 'WABASH'

5 1/4 8inch
SS/SD \$25.00 \$25.00
SS/DD 27.40 30.40
DS/DD 32.40 37.40

Box of 10 pcs

QTY. PRICE AVAIL

COMPUTERS ATARI 800™ COMPUTER SYSTEM

400 w/16K \$350.00
800 w/16K \$699.00
800 Computer w/48K \$825.00

ATARI PERIPHERALS:

Printer 825 - 650.00 Asteroids
Disk Dr. 810 - 485.00 Missile Com. -> 32.50
Record. 410 - 820.00 Sup. Brk Out
Paddle (pr) > 16.95 Assem. Edit. - 490.00
Joystick (pr) > 16.95 Star Raiders - 450.00
32k RAM - 179.95 Basketball - 280.00
Basic Cart - 490.00 Chess - 320.00

*** MONITORS ***

ZENITH 1 12in. 15MHz. 1. Green Phos. Hi-Res. -> \$118.50
J.C.S. 2 12in. 18MHz. 2. Green Phos. Hi-Res. -> \$155.50
AMDEK 3 12in. 12MHz. 3. Green Phos. Hi-Res. -> \$165.50
13in. Color 3a 3a Lo-Res. -> \$375.50

CONCORD COMPUTER PRODUCTS

1971 SO STATE COLLEGE ANAHEIM, CALIF 92806
(714) 937-0637

10 MIN ORDER. CA RES ADD 6% FR

10 45 12.00 250 499 99.00
50 95 12.00 200 399 110.00
100 240 8.00 1000 UP CALL

COMPONENTS

74LS SERIES		7400 SERIES	
74LS00 24	74LS83 70	74LS1665 85	7400 .18
74LS02 24	74LS86 38	74LS1666 85	7454 .19
74LS03 24	74LS90 60	74LS1667 85	7455 .19
74LS04 24	74LS91 85	74LS1668 85	7456 .19
74LS05 24	74LS92 65	74LS1669 85	7457 .19
74LS06 30	74LS93 60	74LS1670 85	7458 .19
74LS08 28	74LS95 80	74LS1671 85	7459 .19
74LS10 24	74LS107 38	74LS1672 85	7460 .19
74LS11 33	74LS109 38	74LS1673 85	7461 .19
74LS12 33	74LS112 40	74LS1674 85	7462 .19
74LS13 45	74LS113 40	74LS1675 85	7463 .19
74LS14 89	74LS114 45	74LS1676 85	7464 .19
74LS20 24	74LS122 40	74LS1677 85	7465 .19
74LS21 24	74LS123 85	74LS1678 85	7466 .19
74LS26 30	74LS125 90	74LS1679 85	7467 .19
74LS27 28	74LS126 85	74LS1680 85	7468 .19
74LS28 32	74LS132 70	74LS1681 85	7469 .19
74LS30 24	74LS136 50	74LS1682 85	7470 .19
74LS32 32	74LS138 65	74LS1683 85	7471 .19
74LS37 50	74LS139 65	74LS1684 85	7472 .19
74LS38 32	74LS145 110	74LS1685 85	7473 .19
74LS48 50	74LS151 70	74LS1686 85	7474 .19
74LS49 70	74LS153 70	74LS1687 85	7475 .19
74LS51 24	74LS157 70	74LS1688 85	7476 .19
74LS55 35	74LS158 70	74LS1689 85	7477 .19
74LS74 35	74LS161 85	74LS1690 85	7478 .19
74LS75 48	74LS162 85	74LS1691 85	7479 .19
74LS76 38	74LS163 85	74LS1692 85	7480 .19
74LS78 50	74LS164 85	74LS1693 85	7481 .19
		74LS1694 85	7482 .19
		74LS1695 85	7483 .19
		74LS1696 85	7484 .19
		74LS1697 85	7485 .19
		74LS1698 85	7486 .19
		74LS1699 85	7487 .19
		74LS1700 85	7488 .19
		74LS1701 85	7489 .19
		74LS1702 85	7490 .19
		74LS1703 85	7491 .19
		74LS1704 85	7492 .19
		74LS1705 85	7493 .19
		74LS1706 85	7494 .19
		74LS1707 85	7495 .19
		74LS1708 85	7496 .19
		74LS1709 85	7497 .19
		74LS1710 85	7498 .19
		74LS1711 85	7499 .19
		74LS1712 85	7500 .19
		74LS1713 85	7501 .19
		74LS1714 85	7502 .19
		74LS1715 85	7503 .19
		74LS1716 85	7504 .19
		74LS1717 85	7505 .19
		74LS1718 85	7506 .19
		74LS1719 85	7507 .19
		74LS1720 85	7508 .19
		74LS1721 85	7509 .19
		74LS1722 85	7510 .19
		74LS1723 85	7511 .19
		74LS1724 85	7512 .19
		74LS1725 85	7513 .19
		74LS1726 85	7514 .19
		74LS1727 85	7515 .19
		74LS1728 85	7516 .19
		74LS1729 85	7517 .19
		74LS1730 85	7518 .19
		74LS1731 85	7519 .19
		74LS1732 85	7520 .19
		74LS1733 85	7521 .19
		74LS1734 85	7522 .19
		74LS1735 85	7523 .19
		74LS1736 85	7524 .19
		74LS1737 85	7525 .19
		74LS1738 85	7526 .19
		74LS1739 85	7527 .19
		74LS1740 85	7528 .19
		74LS1741 85	7529 .19
		74LS1742 85	7530 .19
		74LS1743 85	7531 .19
		74LS1744 85	7532 .19
		74LS1745 85	7533 .19
		74LS1746 85	7534 .19
		74LS1747 85	7535 .19
		74LS1748 85	7536 .19
		74LS1749 85	7537 .19
		74LS1750 85	7538 .19
		74LS1751 85	7539 .19
		74LS1752 85	7540 .19
		74LS1753 85	7541 .19
		74LS1754 85	7542 .19
		74LS1755 85	7543 .19
		74LS1756 85	7544 .19
		74LS1757 85	7545 .19
		74LS1758 85	7546 .19
		74LS1759 85	7547 .19
		74LS1760 85	7548 .19
		74LS1761 85	7549 .19
		74LS1762 85	7550 .19
		74LS1763 85	7551 .19
		74LS1764 85	7552 .19
		74LS1765 85	7553 .19
		74LS1766 85	7554 .19
		74LS1767 85	7555 .19
		74LS1768 85	7556 .19
		74LS1769 85	7557 .19
		74LS1770 85	7558 .19
		74LS1771 85	7559 .19
		74LS1772 85	7560 .19
		74LS1773 85	7561 .19
		74LS1774 85	7562 .19
		74LS1775 85	7563 .19
		74LS1776 85	7564 .19
		74LS1777 85	7565 .19
		74LS1778 85	7566 .19
		74LS1779 85	7567 .19
		74LS1780 85	7568 .19
		74LS1781 85	7569 .19
		74LS1782 85	7570 .19
		74LS1783 85	7571 .19
		74LS1784 85	7572 .19
		74LS1785 85	7573 .19
		74LS1786 85	7574 .19
		74LS1787 85	7575 .19
		74LS1788 85	7576 .19
		74LS1789 85	7577 .19
		74LS1790 85	7578 .19
		74LS1791 85	7579 .19
		74LS1792 85	7580 .19
		74LS1793 85	7581 .19
		74LS1794 85	7582 .19
		74LS1795 85	7583 .19
		74LS1796 85	7584 .19
		74LS1797 85	7585 .19
		74LS1798 85	7586 .19
		74LS1799 85	7587 .19
		74LS1800 85	7588 .19
		74LS1801 85	7589 .19
		74LS1802 85	7590 .19
		74LS1803 85	7591 .19
		74LS1804 85	7592 .19
		74LS1805 85	7593 .19
		74LS1806 85	7594 .19
		74LS1807 85	7595 .19
		74LS1808 85	7596 .19
		74LS1809 85	7597 .19
		74LS1810 85	7598 .19
		74LS1811 85	7599 .19
		74LS1812 85	7600 .19
		74LS1813 85	7601 .19
		74LS1814 85	7602 .19
		74LS1815 85	7603 .19
		74LS1816 85	7604 .19
		74LS1817 85	7605 .19
		74LS1818 85	7606 .19
		74LS1819 85	7607 .19
		74LS1820 85	7608 .19
		74LS1821 85	7609 .19
		74LS1822 85	7610 .19
		74LS1823 85	7611 .19
		74LS1824 85	7612 .19
		74LS1825 85	7613 .19
		74LS1826 85	7614 .19
		74LS1827 85	7615 .19
		74LS1828 85	7616 .19
		74LS1829 85	7617 .19
		74LS1830 85	7618 .19
		74LS1831 85	7619 .19
		74LS1832 85	7620 .19
		74LS1833 85	7621 .19
		74LS1834 85	7622 .19
		74LS1835 85	7623 .19
		74LS1836 85	7624 .19
		74LS1837 85	7625 .19
		74LS1838 85	7626 .19
		74LS1839 85	7627 .19
		74LS1840 85	7628 .19
		74LS1841 85	7629 .19
		74LS1842 85	7630 .19
		74LS1843 85	7631 .19
		74LS1844 85	7632 .19
		74LS1845 85	7633 .19
		74LS1846 85	7634 .19
		74LS1847 85	7635 .19
		74LS1848 85	7636 .19
		74LS1849 85	7637 .19
		74LS1850 85	7638 .19
		74LS1851 85	7639 .19
		74LS1852 85	7640 .19
		74LS1853 85	7641 .19
		74LS1854 85	7642 .19
		74LS1855 85	7643 .19
		74LS1856 85	7644 .19
		74LS1857 85	7645 .19
		74LS1858 85	7646 .19
		74LS1859 85	7647 .19
		74LS1860 85	7648 .19
		74LS1861 85	7649 .19
		74LS1862 85	7650 .19
		74LS1863 85	7651 .19
		74LS1864 85	7652 .19
		74LS1865 85	7653 .19
		74LS1866 85	7654 .19
		74LS1867 85	7655 .19
		74LS1868 85	7656 .19