microware®

# PIPELINES

Covering the Full Spectrum of 6809 and 68000 Applications of OS-9

Volume 2 Issue 1 Spring 1986


In This Issue:

**PIPELINES**
Spring 1986
Volume 2 Number 1

**Pipelines is published quarterly by:**
Microware Systems Corporation
1866 N.W. 114th Street
Des Moines, IA 50322
(515)224-1929

**Contributing Editors:**
Andy Ball
Phyllis Casel
Ken Kaplan
Kim Kempf
Dave West

**Photography:** George Edwards

Do you have news for Pipelines? Articles submitted for publication should include authors name, address, telephone number and date. Comments, suggestions and letters of "gripes" as well as "praise" are solicited. Please send to Microware, c/o Phyllis Casel. If there is someone you know who would like to be on our mailing list, please call or mail their name and address.

## CHANGE OF ADDRESS

Please note that two Microware Distributors have new addresses.

Elsoft AG
Rudolf Hug
Zelweg 12
CH 5405 Baden-Dattwil
Switzerland
phone: (41)056-833377
telex: 828275

Microprocessor Consultants Propriety
Jeff Skebe
92 Bynya Road
Palm Beach 2108
NSW Australia
phone: 02-919-4917

## COVER PHOTO

The OS-9 billboard shown on the front cover sits on top of Microware Japan's office building. It overlooks the entire downtown area in Funabashi City, Japan.

## MICROWARE PLAYS KEY ROLE IN CD-ROM STANDARD

Two electronics industry giants have announced a revolutionary new CD-ROM standard incorporating technology developed by Microware Systems Corporation. Philips N.V. of The Netherlands and Sony Corporation of Japan have unveiled a new type of compact disc capable of holding not only hi-fi sound but video pictures and computer data as well.

The new product, called Compact Disc Interactive Media ("CD-I"), heralds a new era in entertainment, education and electronic publishing. Based on the small, sturdy, super high fidelity compact disc which is currently sweeping the audio world, CD-I adds natural pictures and interactive capabilities. This will open the way for an important new method of distributing information. Some possibilities are talking encyclopedias, dictionaries, textbooks, electronic magazines and catalogs, as well as exciting new entertainment titles such as "music video" style presentations, songs with words and computer games with life-like realism. This new media may also offer other possibilities as yet unimagined.

The interactive aspect of the media is especially significant as it allows the user to rapidly locate information and communicate with the highly intelligent CD-I player. The basic player is as easy to hook up and operate as a video cassette recorder, plus it can be optionally expanded to a full-feature personal computer. The new CD-I players are completely compatible with all current CD digital audio discs and the standardized format assures that any disc can be used with any player.

CD-I discs are also distinguished by their enormous storage capacity. Each inexpensive disc is capable of holding as many as 600 Mbytes of digital data, compressed audio and graphics. This translates to over 150,000 printed pages of text or up to 20 hours playing time of speech quality sound.

Philips and Sony originally codeveloped the CD digital audio format which was first marketed in 1982. Because the new CD-I media relies heavily on advanced software technology, Sony and Philips invited Microware to participate in the CD-I development program. Microware contributed its special expertise in the creation of the control software for the micro computer-based CD-I player and the information storage format for the discs.

CD-I players are based on the 68000 family microprocessor plus new specialized LSI components developed by Sony and Philips, including audio and video coprocessors. The software is based on the kernel of Microware's OS-9/68000 operating system plus a newly developed CD-I file manager which is tailored to the special characteristics of the read-only media. For example, it allows any file to be opened with only one access even though the file system can have many levels of directories.

The Microware-developed system software package also fully supports the multiple audio and video formats including mixed data types within a single file. Video display capabilities include both natural (delta YUV) and bit-plane graphics (RGB and color look-up table) in normal (384H by 280V) and high resolution (768H by 560V) modes. Four digital audio quality levels range from ultra high-fidelity standard CD to high grade speech. This allows software designers to select audio and video quality levels which optimize storage requirements and playing time for the material to be presented.

It is expected that many major consumer electronics and personal computer manufacturers will obtain licenses to manufacture CD-I players. The basic player software will be included in the master licenses issued by Philips and Sony. Manufacturers may obtain licenses directly from Microware for the optional extensions which upgrade a basic player to a full feature personal computer configuration.

## UPDATE VERSION OF THE C-COMPILER

Microware Systems Corporation is proud to announce the release of the new Version 2.0, OS-9/68000 C compiler. The new release includes both a maintenance update and many additional improvements. The new C compiler manual set is now available and has been updated to reflect the many modifications and enhancements.

The following is a synopsis of the differences between Version 1.3 and 2.0 of the OS-9/68000 C compiler.

* The cc command specifies the default preprocessor include and linker search library directories. A new option (-w=<dir>), specifies the linker library default search directory. The -R=<rels> option now places the output file relative to the current directory rather than the source file directory.

The C preprocessor supports longer macro names, the maximum length is 256 bytes. Two new preprocessor-defined names "__LINE__" and "__FILE__" are available to access the current source line number and the current source file name respectively.

* The compiler will allow non-unique structure and union member names. Extensive type checking is performed by the compiler. Warning messages are issued when operand types require implicit type conversation. This feature is commonly found on VAX-style C compilers. The first 256 characters of all identifier names are significant, excess characters are ignored.

* This release also includes new versions of the assembler (R68), the linker (L68) and the debugger program (debug) to control longer identifier names. The utilities will appear in the next release of the OS-9/68000 Utility Commands. These programs are completely upwards compatible with earlier versions and should replace existing versions in the commands directory.

* The previous versions of these programs truncated identifier names of only eight characters. Existing relocatable files should be reassembled, or the linker may not match the longer names with the shorter names. The new version of L68 will issue a warning when processing a relocatable file produced by an older version of the assembler.

* The debugger can handle programs linked with either the new or the old linker. A warning message is issued when processing an old-style symbol table module as a reminder to re-link the program with the new linker.

These versions of the assembler, linker and debugger are not backwards compatable. Output from these programs cannot be processed by any earlier version of the same.

The C library contains the following changes and additions:

- The __prgname() function is now available in the cio trap handler.
- The <sgtbuf.h> header file has been changed to allow compilation with the non-unique struct members.
- Nineteen new functions have been added for accessing OS-9 system calls.
- An include file <setsys.h> is available for use with the __setsys() function.
- An include file <events.h> is available for use with the event functions.

Updates are available directly through Microware Systems, Microware distributors and vendors of Microware software products. An update coupon/order form has been included for your convenience. (See back page).

## 6809 FORTRAN COMPILER

Microware announces the release of the OS-9/6809 Level Two FORTRAN Compiler, a subset of the FORTRAN 77 ANSI standard. The differences between the OS-9 implementation and the ANSI Standard are primarily the result of memory limitations imposed by 6809's 64K address space. The compiler includes a comprehensive user's manual that fully describes the OS-9 implementation.

Highlights include:

* The ability to generate code for two or four byte integers. The 2-byte integer mode gives a range of -32,768 to 32,767; the code generated is compact and fast. The 4-byte integer allows an extended range from -2,147,483,648 to 2,147,483,647.

* The compiler uses the same floating point representation as the 6809 C Compiler. Single precision is accurate to seven digits. Double precision is accurate to 15 digits. The compiler does not support complex numbers.

* A full math library containing functions such as sine, cosine, tangent, arc sine, arc cosine and arc tangent for both single and double precision is included.

Microware recommends that beginning FORTRAN programmers obtain a copy of "FORTRAN 77: Principles of Programming," written by Jerrold Wagener, John Wiley and Sons. This book is an easy to read tutorial and covers the entire range of programming in FORTRAN.

FORTRAN is available immediately through Microware Systems or any Microware Distributor. A coupon has been enclosed for your ordering convenience. (See back page).

## POWERFUL NEW OS-9 NETWORKING

Microware announces the release of the OS-9 Network File Manager (NFM), a powerful software based networking system for 68000 family processors. A version of NFM for 6809 is scheduled for release later this year.

The Network File Manager combines the file and input/output systems of all connected computers into one logical file system. Any network user can directly access files and I/O devices on any other system on the network as if they were local files. This makes

(continued on the next page)

(OS-9 Networking continued)
the network system extremely easy to use. Also, existing utility application software can be used as is.

Because Network File Manager has a software-based architecture, it is completely hardware independent. It can be used with a wide variety of standard local area network or long-haul data communications hardware. The network can easily be customized for specific communications controllers using simple standard or user-written driver modules. It is compatible with virtually all popular standards such as ETHERNET, OMNINET, ARCNET, IEEE-488, etc. Multiple networks can be connected to a single system. This configuration allows direct inter-network access. Therefore "gateways" between different networks can be established.

Special security features built into the Network File Manager allow full control of local file access from other systems. Combined with the operating system's standard file security mechanism, this provides complete file protection from unauthorized access.

The Network File Manager has undergone extensive testing at Beta test sites world wide, and is immediately available under OEM licenses or by single copy "NetPaks". A "NetPak" is a special package that allows a user to customize and install the network on their own systems. It includes the Network File Manager software, sample source code for typical device driver modules and special installation documentation. Contact Microware Systems or any Microware distributor for additional ordering information.

## TRAINING/EDUCATION DEPARTMENT

The continuing emergence of Microware's OS-9 as a major factor in the 68xxx operating systems world has resulted in the dramatic growth of the number and diversity of OS-9 users. This, in turn, has resulted in an increasing number of requests for cus-

tomer training programs at all levels, ranging from the novice end-user to system engineers and software developers. In order to provide better service for customers, Microware has established a new department for education and training, headed by Dr. James W. Moore, Jr., who joined Microware in February.

Bill, as he prefers to be known, is a native Texan, living there until 1979. He holds B.A. and B.S. degrees in English and Electrical Engineering from Texas A&I, and a PhD. in Behavioral Science from the University of Houston. Following graduation from A&I, he worked at Manned Spacecraft Center, now Johnson Space Center in Houston, until beginning graduate work at U of H. He has worked at Duke and Middle Tennessee State University utilizing computer-based equipment in a number of behavioral experiments. In addition, he taught various graduate and undergraduate psychology courses at MTSU.

In 1981, Bill was employed by U.S. Tobacco, Inc., in Nashville, TN, as a Computer Applications Engineer, developing single board controllers for machine applications. It was during the course of evaluating alternate software for these systems he first encountered OS-9. Impressed by the modular design and functionality of OS-9, he purchased a Level I 6809 machine for his personal use, which allowed him to become familiar with the inner workings of OS-9, as well as learning how to use Basic09 and C. He has written an article on OS-9 which appeared in ACCESS magazine.

Bill will be designing and presenting a variety of training programs. These will include tightly focused sessions tailored to the specific needs of individual customers and OEMs, as well as regional seminars intended for a broad spectrum of general users, in order to provide enhanced training support for Microware's rapidly growing user base. If your company or customers are interested in OS-9 family software educational seminars please contact Bill Moore of Microware.

## NEW FACES

Tony Hoffman joined the Microware Research and Development staff in February. He graduated from the University of Iowa with a B.S. in Computer Science. While in college he focused his studies on systems software, mathematics and graphics. Tony obtained an OS-9/6809 development system and utilized it for school projects as well as for personal use.

Tony was the recipient of one of Microware's travesty awards at the 1985 annual OS-9 seminar.

Molly Knee, Accounting Assistant, is Microware's newest family member. She graduated from N.E. Missouri State, (NMSU), in Kirksville, with a certificate in Accounting and Data Processing. She was Treasurer of the NMSU Hall Government for a year and a half. Molly has word processing experience, and is familiar with COBOL, Pascal and Basic.

# ON THE C SIDE

In the Winter 1985 edition of Pipelines a few strategies were shared to help clear up some of the misapprehensions concerning C. The topics discussed were C library I/O, C documentation and choosing the correct I/O function. Included in this edition are more helpful hints to help both the novice and the pro programmers better understand the C language. Topics covered will be stream I/O buffering, the FILE stream structure, changing the stream I/O defaults and stream file update mode.

### Stream I/O Buffering

An often confused topic is the nature of the stream buffering provided by the stdio functions.

The low-level functions provide no buffering other than that within the appropriate device driver. The read/readln() functions are direct hooks to the I$Read/I$Readln system calls. Likewise, the write/writeln() functions are direct hooks to the I$Write/I$WritLn system calls. These functions simply transfer the data to or from the program's buffer.

The stdio functions provide a full buffering facility referred to as a stream. In addition to providing an operating system independent method of performing I/O, these streams also reduce the system call overhead associated with the I/O operation. The amount of system overhead to transfer one character to a device is the same as that required to transfer 256 characters. Because of this, the stdio function will attempt to transfer the maximum number of characters possible.

### The FILE stream structure

The stdio functions communicate via a structure as defined in <stdio.h>:

```
typedef struct _iobuf {
    char *_ptr,          /* I/O buffer pointer */
         *_base,         /* I/O buffer base address*/
         *_end;          /* I/O buffer end address*/
    WORD _flag;          /* file status flag */
#define _READ          1 /* read access allowed */
#define _WRITE         2 /* write access allowed */
#define _UNBUF         4 /* stream is unbuffered */
#define _BIGBUF        8 /* stream is buffered */
#define _EOF        0x10 /* stream is at end-of- file */
#define _ERR        0x20 /* stream has encountered an error */
#define _SCF        0x40 /* stream is attached to a character
                            device */
#define _RBF        0x80 /* stream is attached to a block
                            device */
#define _DEVMASK    0xc0 /* device bits mask */
#define _WRITTEN  0x0100 /* buffer status flag for fseek/get/put
#define _INT      0x8000 /* stream has been initialized */
   WORD _fd;             /* file path number */
   char _save;           /* for "ungetc" when unbuffered I/O */
   int (* _ifunc)();     /* function to use for input */
   int (* _ofunc)();     /* function to use for output */
}FILE;
```

### Changing the stream I/O defaults

Some of the values in this structure can be set by the programmer to control the operation of the stdio functions. Once I/O is performed on a stream, the values in the struct cannot be changed until the stream is closed. Doing so causes unpredictable results.

The _flag value controls the operation of the stream and provides the operational status of the buffer. Normally, the flag values are supplied by the stdio functions when the stream is first accessed. If certain flags are set before access, explicit actions can be forced. The following flags can be set by the user program.

| Flag | Action |
|---|---|
| _READ | Open stream for read |
| _WRITE | Open stream for write |
| _READ/_WRITE | Open stream for read/write |
| _UNBUF | Unbuffered stream |
| _SCF | Use readln() and writeln for I/O |
| _RBF | Use read() and write() for I/O |

The _READ and _WRITE flags need only be modified for the standard streams, stdin, stdout and stderr. The fopen() functions set the correct flags from its open mode argument. The _UNBUF, _SCF and _RBF flags are set when the first I/O is performed on the stream. These flags are normally set by the action of putc() and getc(). The remaining flag values are reserved for use by the internal stdio routines.

The _bufsiz value can be changed to set the size of the I/O buffer. Sometimes, a larger I/O buffer can result in faster program execution because the I/O overhead is reduced. The _bufsiz value can be changed as follows:

```
fp->_bufsiz = 5120;
```

**NEXT ISSUE:**
Stream I/O buffering continued.

## OS-9 ON IBM PC/XT/AT

Plug in CPUs providing OS-9/68000 support on the IBM PC, XT, and AT are available now from Hallock Systems Company and TLM Systems, Inc. The "Pro 68" from Hallock and the "PC 68K" by TLM were demonstrated at the 4th Annual Microware Seminar where they attracted considerable attention. For more information call:

Hallock Systems Co.
267 North Main St.
Herkimer, NY 13350
Ph:(315)866-7125

TLM Systems, Inc.
4704 W. Jennifer
Suite 105
Fresno, CA 93711
Ph:(209)276-2333

# NEW SHELL ENVIRONMENT VARIABLES

The feature program for the Spring edition of Pipelines is "Initialized Environmental Variables." This program is used to set environment variables for a shell forked from login or chained from the command line. It looks for a file called ".envfile" in the current working directory and reads this file into an environment array. It then chains to a new shell. All child processes will then inherent these environment variables to use as they please. This program is for OS-9/68000 systems only, and has been provided by the Pipelines staff, not the support service staff; therefore, ON ERROR do not GOTO 515/224-0458.

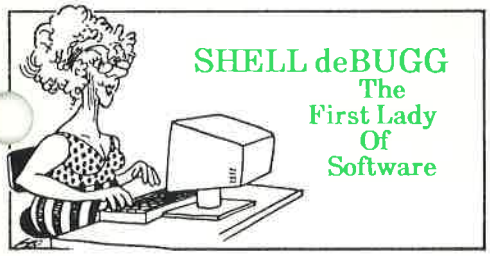
```
/*The format for this call is:  initenv <parameter list>
            eg.  initenv scred
                        This would chain to a shell which would fork scred passing
                        environment parameters contained in file ".envfile
```

This command could be used in the password file to be used to setup the environment variables for your login shell or program.

```
            Example password file entry: rick,rick,12.5,128,,,/h0/rick/call,initenv ex scred -e
```

In order for this command to work you must have a file called ".envfile" in your working directory.

```
                        This is an example ".envfile" file:
                                    NAME=rick
                                    TERM=abm85
#include <stdio.h>                  DEV=/dd   */
#define EOF         0
#define MAXENVSZ 128
#define MAXSTRNG 256

extern int chain();
char *envfile = ".envfile",
      *envp[MAXENVSZ],
       envstrng[MAXSTRNG];

main(argc,argv)
register int argc;
register char **argv;
{
/* load up the environment array */
      getenv();

/* chain to shell passing the environment array */
      os9exec(chain,"shell",argv,envp,0,0,0);
}

getenv()
{
      FILE *envpath;
      register char **envptr;
      register char *ptr;

      envptr = envp;

/* open path to environment file in user's working directory */
      if ((envpath = fopen(envfile,"r")) == -1)
         error("Can't read environment file\n",errno);

/* get a buffer,read entries from environment file */
/* and put in environment array until end of file */
      while ((envptr < &envp[MAXENVSZ-1])
            && (fgets(envstrng,MAXSTRNG,envpath) != EOF)) {

      /* request memory for string */
         if ((*envptr = malloc(strlen(envstrng))) == NULL)
            error("Out of memory\n",errno);

      /* copy string into environment array */
         strcpy(*envptr,envstrng);

      /* replace "CR" with a null character */
         (*envptr)[strlen(envstrng)-1] = '\0';
         envptr++;
      }
      fclose(envpath);
}

error(mesg,err)
register char *mesg;
register int err;
{

      puts(mesg);
      exit(err);
}
```



# SHELL deBUGG

The First Lady Of Software

**Dear Shell,**
It seems like I'm always waiting for a time slice to run an application that needs to execute every 10 milliseconds. Is it possible to run a process on an OS-9/68000 machine with a real-time clock? -**No Name**

**Dear No Name,**
It is possible to have a process run every 10 milliseconds. On most systems with a real-time clock, the CPU is interrupted by the clock at a rate 100 times per second. This interval of time is referred to as a "tick", and has a duration of 10 milliseconds. At any occurence of a tick, the operating system can suspend the execution of one program and begin another. Active processes (those not awaiting a signal) are run for a specific period of time called a "time slice." How often a process receives a time slice depends on the process' priority in relation to the priority of the other active processes.

All active processes are members of the active process queue, which is kept sorted by process age. Process age is a count of the number of process switches that have occurred since the process was last given a time slice, plus the process' initial priority. The initial priority can be set using the F$SetPrior System Call or the utility command, SETPR.

When a process is moved to the active process queue, it's age is initialized by setting it equal to the process' assigned priority. Processes having relatively higher priority are placed in the queue with an artificially higher age. When a new process is placed in the active queue the ages of all other processes are incremented, but never beyond $FFFF. Processes are run in descending order of age.

During a demanding real-time application, very fast interrupt response may be required. This can be done by preempting the current executing process when a process with a higher priority becomes active. The lower priority process loses the remainder of its time slice to the new process and is reinserted into the active process queue.

Two system global variables, D_MinPty and D_MaxAge, affect task switching. Both of these are accessed by the F$SetSys system call. D_MinPty, usually set to zero on most systems, defines a minimum priority below which processes are neither aged nor considered candidates for execution. D_MaxAge, usually set to 0 on most systems, sets a maximum age that processes are not allowed to mature above. When used, D_MaxAge divides tasks into low and high priority classes. Low priority tasks stop aging at the MaxAge cutoff. The high priority tasks will receive the entire available CPU time. Low priority tasks will be run only when the high priority tasks are inactive.

The drawback of requiring a process to run every 10 milliseconds is that the task could tie up all CPU time and never allow other processes to execute.

**Dear Shell deBugg,**
Recently I came upon a situation that's against my moral principles. I'm talking about euthanasia. I have a C language program which uses OS9fork() to spawn a child process. When the child process dies, it's process descriptor is not deallocated until the parent process dies. What is happening?
**-Terminal in Ohio**

**Dear Terminal,**
I'm sure my readers share your sadness in the passing of your parent and child processes. However, termination is part of creation. OS9fork() will create a process that will run concurrently with the calling process. When the new process terminates, its exit status is available to the forking (parent) process. The parent process should execute a "wait()" to suspend the current process until the child process has terminated. Wait() will return -1 if there is no child process to wait for.

**Dear Ms. deBugg,**
When I was a ComSci student I got hooked on speed. It started innocently. On weekends, I'd push the step rate of my floppy disks to the max. Then one night at a party, some of my frat brothers had some hard stuff, five meg, 15 meg and even 40 meg. I couldn't help myself, I had to have one. I pawned my stereo and bought a 15 meg Winchester. When I got home I copied everything from my floppy system disk to the hard disk, but I couldn't get it to boot from the hard disk. The bootfile was there, why wouldn't it boot? Desperation set in. Next thing I knew, I woke up in the hospital undergoing psychiatric evaluation. I explained my predicament to my analyst. His words were so wise and comforting, I felt I should share them with your readers.

"Merely copying OS9Boot to a new disk will not make it bootable. It is very likely that the OS9GEN utility was not used on the new bootfile. OS9GEN is used to create and link the OS9Boot file that is required on any disk from which OS-9 is to be bootstrapped. If using the DSAVE utility to make a new system disk, the '-o' option may be used to invoke OS9GEN to create and link the bootfile on the destination device." Since then I've learned that an OS9Boot file must be stored on physically contiguous sectors. Therefore, it is a good practice to use OS9GEN on a freshly formatted disk.
**-Still Hooked, but Under Control**

**Dear Hooked,**
Thank you for sharing you story. By the way, have you tried Microware's RAM disk? Talk about speed!

* * * * * * * * * * * * * * * * * * * *

Send your OS-9 questions to:
Microware Systems Corporation
c/o SHELL deBUGG
1866 N.W. 114th Street
Des Moines, IA 50322

* * * * * * * * * * * * * * * * * * * *

## ACCEPTING APPLICATIONS

Microware is expanding it's programming and marketing staff, and currently accepting resumes for job openings. This is an excellent opportunity to get involved with the design and implementation of 6809, 68000 and "beyond" innovative software concepts. Programming applicants should be well oriented with OS-9, operating systems and compiler technology. Marketing candidates should have a computer related background, good communications skills and a superior track record in sales.

Send a resume containing your educational background, work experience and salary requirements to the attention of the Personnel Director at Microware. No phone calls please!

## MICROWARE SOFTWARE UPDATE/ORDER FORM

The adjacent update/order form has been enclosed for your ordering convenience. If you wish to order either the OS-9/6809 Level Two FORTRAN compiler or the Version 2.0, OS-9/68000 C compiler, simply fill out the form below. Please allow three to four weeks delivery on software. Customers outside the continental United States should contact the Microware distributor or representative nearest you for information on delivery costs.

The OS-9/6809 Level Two FORTRAN compiler includes the compiler, linker, assembler and a comprehensive users manual that fully describes the OS-9 implementation. The cost is $253.00, which includes shipping charges.

The Version 2.0, OS-9/68000 C compiler update includes the disk and manual which is delivered in an attractive gray binder with slipcover. The cost is $103.00, which includes shipping charges. For those customer who already have Version 1.3 of the C compiler and wish only to order the updated disk and 38 page manual update, (not the complete manual), the cost is $78.00, which includes shipping charges. **Please note: Your order will not be processed unless the original disk is returned. Purchase orders are not be accepted on update orders.

### MICROWARE UPDATE/ORDER FORM

Company Name: _____

Address: _____

_____

_____

Purchasing Contract: _____ PO#: _____

Phone Number: _____

Mastercard __ Visa __ Card # _____ Ex Date _____

Check Enclosed __ Signature: _____

Format: _____

I wish to order:

___ The FORTRAN compiler ($253.00)

___ The C compiler complete manual update ($103.00)

___ The C compiler partial manual update ($78.00)

| | | |
|---|---|---|
| Disk size: | 5": __ | 8": __ |
| No. of sides: | 1: __ | 2: __ |
| Density: | 1: __ | 2: __ |
| No. of tracks: | 40: __ | 80: __ |

MICROWARE®

Microware Systems Corporation
1866 N.W. 114th Street
Des Moines, Iowa 50322

Bulk Rate
U.S. Postage
Paid
Des Moines, IA
Permit #6238

ADDRESS CORRECTION REQUESTED