W E N D E L

R H Y T H M A U G M E N T A T I O N S Y S T E M
AUDIO SAMPLER
ADSR ENVELOPE & TRIGGER SYSTEM

0 1 / 0 1 / 79

S Y S T E M O P E R A T I O N

COPYRIGHT NOTICE

DISCLAIMER

Roger S. Nichols and WENDEL LABS makes no representation or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantibility or fitness for any particular purpose. Further, Roger S. Nichols or WENDEL LABS reserves the right to revise this manual or the programs supplied and to make changes from time to time in the contents of either the manual or the programs without obligation of Roger S. Nichols or WENDEL LABS to notify any person or organization of such revision or changes.

TRADEMARK

The names Wendel, WENDEL LABS, and RHYTHM AUGMENTATION SYSTEM are trademarks of WENDEL LABS.

SYSTEM OVERVIEW

Wendel is the latest in "state of the art" applications involving digital audio storage. Wendel can be used to repair or to augment rhythm tracks in an almost infinite number of ways. Wendel will not easily become obsolete because the entire operating system is loaded into the system each time it is used. Updating to future enhancements is done simply by obtaining a new system diskette.

<METRONOME:>
This mode allows Wendel to put down a perfect click track for the musicians to play along with. This need not be the every day digital click track that we are all used to. Wouldn't it be great to play along with machine steady hand clapps instead of "thwap - thwap - thwap - thwap". How about 1/4 note wood blocks with accents on 2 and 4? Have you ever wanted to have both 1/4 note and 1/8th note clicks on tape for future overdubs? Tired of playing along with a rhythm box that doesn't have quite the right feel for the tune? "Gee, it sure would be nice if the snare drum were layed back a little more on the rhythm machine!" Wendel can do all of these things, and much more.

<MANUAL:>
The manual play mode lets you select a number of sounds from the disk library and play them manually. If you can tap your finger in time to the music, you can be a drummer.

<AUTOMATIC PLAY:>
This mode will come in very handy if you have trouble keeping time by taping your fingers. As a matter of fact, Wendel plays so steady that we included the ability to make the playing as loose as you want it to be. You can lay back the snare drum till any normal drummer would fall off the stool. Have you ever listened to a tune and heard a couple of bars where you thought, "Wow, if this guy could lay back like that all of the time, he would be greatest drummer in the world!"..... Enter Wendel.

AND TALK ABOUT STEADY!!! Too bad disco is dead..... Oh well, maybe in the next life.

<REPAIR:>
Repair mode could save your whatchamacallit. Yup, that live performance when the snare drum broke. "OOOOOH NOOOOOOOoooooo. That take was perfect! we'll never be able to fix the snare track. Boooo hoooo, hoooo glub." Oh yes, you can fix the snare track, or just about anything else in the rhythm track that didn't quite make it on to the tape the way you had hoped.

There are two basic ways to fix a broken snare. You can replace the entire track with a sound already stored in Wendel, or you can store the sound of the snare drum from the begining of the tune, and use it to repair the broken snare sound at the end of the tune. This method also works well when you intercut two or more takes of the same tune, and the snare sounds different from one piece to the next. Take the sound you like, and use it for the whole tune.

The same concept works on tracks for an album that were cut with two different drummers, or 6 months apart. One snare drum sound can be used throughout the album to make it sound more like it was the same band playing all the tunes.

Keep in mind that in the repair mode, Wendel will play exactly what the original track played. If the player rushed, Wendel will rush. If he was perfect (a person, perfect), then Wendel will be perfect. Of course if you want it to be different, Wendel can do that too.

<AUTOMATIC SEQUENCER:>
This mode enables Wendel to sequence a synthisizer as well as control the envelope, attack and decay of the notes played on the synthisizer by a keyboard player. This mode is used generally for playing perfect rhythm keyboard parts on which to build a track.

<STORE:>
Store mode lets you add your favorite drum or percussion sounds to the disk library. This is accomplished simply by plugging Wendel into the console, and playing the sound you want stored. Wendel samples the sound, cleans up all the extraneous noises, and saves it on disk for future use. Since the bulk of the stored sounds are on floppy disks, and only the ones that you are going to use on the session are loaded into Wendel, the total number of sounds available becomes limitless.

SYSTEM CONFIGURATIONS

There are five basic configuration groups associated with Wendel. These are:

GROUP-1 MEMORY
GROUP-2 OUTPUT CHANNELS
GROUP-3 DISK STORAGE
GROUP-4 MULTI-TASKING
GROUP-5 PACKAGING

<MEMORY OPTIONS:>
Standard memory configuration is 64k bytes, and is expandable to 256k, 512k, up to 16 megabytes of internal memory.

<OUTPUT CHANNELS:>
Standard configurations are four and eight output channels. This is expandable to 128 channels for special applications.

<DISK STORAGE:>
There are two standard disk configurations. The first is a single 630k floppy disk system that can be expanded to four disk modules of 630k each. The second is a 5 megabyte hard disk with 630k floppy drive for system backup. There is also a hard disk option that includes up to eight drives of 32 megabytes each with up to four 630k floppy drives for backup.

<MULTI TASKING:>
The multi tasking option includes add on slave processors that allow Wendel to perform as if there were additional Wendels without duplicating disk storage or memory options.

<PACKAGING:>
Wendel is available in two different packaging configurations.
"System One" includes Wendel installed in a portable carrying case much like a typewriter. The keyboard folds down to reveal a single disk drive, (either the 630k floppy disk or the 5 megabyte hard disk) and the 9 inch CRT display. "System One" also comes with a seperate interface adapter that plugs into the rear of Wendel and contains the necessary access jacks and electronics for attchment to the recording console. "System One" comes with two flight cases.

"System Two" consists of a terminal unit which includes the computer, a 12 inch CRT, and the keyboard. There are two additional units included in the "System Two" configuration. One of the units contains the disk drive units, (a 630k floppy drive and a 5 megabyte hard disk) while the other unit contains the console interface adapters. Both units plug into the rear of the terminal unit.

SYSTEM INSTALLATION

<S Y S T E M O N E:>
Flight case number one contains Wendel. Open the case and remove the unit. Place the unit on a flat surface (producers console or desk area) and open the latches that lock the keyboard closed. Lift the unit in the area of the keyboard hinge and rotate the keyboard away from the unit until the keyboard is fully extended and it's feet are flat on the desk top. Make sure the keyboard has a firm footing, as the front of the unit is now fully supported by the keyboard.

Open flight case number two. In the middle compartment is the console interface unit. Remove it from the case and place it along side or behind Wendel. In the left compartment of the flight case you will find an AC cord and two four foot long ribbon cables. Each end of the ribbon cables is labeled. Locate the end of each cable labeled "INTERFACE UNIT". Plug the cable labeled "INTERFACE UNIT RIGHT" into the connector on the interface unit labeled "RIGHT". Plug the cable labled "INTERFACE UNIT LEFT" into the connector on the interface unit labeled "LEFT".

The other end of the ribbon cables are marked "DIGITAL" and "ANALOG". Connect these to the back of Wendel. The labels on the ribbon connectors should exactly match the labels on the connectors at the rear of Wendel. To make it even plainer, connect "DIGITAL" to "DIGITAL", and "ANALOG" to "ANALOG".

Make sure the power switch on the rear of Wendel is in the off position. Connect the AC cord to the AC receptacle just to the left of the power switch. Plug the other end into an appropriate 117VAC grounded outlet.

<IMPORTANT> Do not use a ground lifting adapter. Wendel MUST be grounded. Static could damage any computer system. Make sure it is grounded.

In the right forward compartment of flight case number two is located a set of orange cables with Cannon type connectors on one end, and TT patch bay plugs on the other end. These cables will be used to connect the interface unit to the console patch bay. For now, we will use three of the cables.

Select two cables with female Cannons. Plug one of them into the interface unit at the jack labeled "OUT 1F". Plug the other end of this cable into a line level input to the console. Mark this input "WENDEL ONE" at the console. Plug the other cable into the interface unit at the jack labeled "OUT 1E". Plug the other end of this cable into another line level input to the console. Mark this input "WENDEL TWO" at the console.

Select a cable with a male Cannon. Plug it into the interface unit at the jack labeled "TRIGGER IN". Plug the other end into a line level console output over which you have control of the audio level.

All connections are now complete.

SYSTEM INITIALIZATION

<STEP 1:>
Turn on the power to Wendel. After about 15 seconds, you will see a sign-on message in the top left corner of the CRT.

Wendel 8517 monitor
>

<STEP 2:>
Insert the disk labeled:

"WENDEL SYSTEM DISK VERSION xxx"

with the label facing toward your left. Push it in all the way and close the disk drive door. (This step may be skipped if Wendel is equipped with a hard disk).

<STEP 3:>
While holding down the key marked "CTRL", depress the key marked "D". The disk drive light should illuminate, and in a few seconds the CRT will display the current Wendel status report, and a sign-on message.

```
  X0   X1   X2   X3   X4   X5   X6   X7   X8   X9   XA   XB   XC   XD   XE   XF
FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF

 S-O-L      S-1-L      S-2-L      S-3-L      S-4-L      S-5-L      S-6-L      S-7-L
4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001

 S-8-L      S-9-L      S-A-L      S-B-L      S-C-L      S-D-L      S-E-L      S-F-L
4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001

    buff port = 1F              ramp port = 1F

G R O U P - 7  S Y S T E M S
(C) Copyright 1980
All rights reserved
>>>Buffer11 ver X.xxxx. .vxxx..ready
```

<STEP 4:>
Depress the key marked "Z", and then "RETURN". A "MENU" of system commands will appear on the lower portion of the CRT. Wendel is now ready for commands.

<STEP 5:>
On floppy disk only systems, open the drive door and remove the "SYSTEM" disk. Insert the disk labeled:

"WENDEL PERCUSSION DISK #01"

and close the drive door.

You are now ready to procede to the next section of this manual.

SYSTEM OPERATION

After performing the steps outlined under SYSTEM INITIALIZATION, the following display should be present on the CRT:

```
   X0   X1   X2   X3   X4   X5   X6   X7   X8   X9   XA   XB   XC   XD   XE   XF
  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF

   S-O-L        S-1-L        S-2-L        S-3-L        S-4-L        S-5-L        S-6-L        S-7-L
 4000 0001    4000 0001    4000 0001    4000 0001    4000 0001    4000 0001    4000 0001    4000 0001

   S-8-L        S-9-L        S-A-L        S-B-L        S-C-L        S-D-L        S-E-L        S-F-L
 4000 0001    4000 0001    4000 0001    4000 0001    4000 0001    4000 0001    4000 0001    4000 0001

      buff port = 1F               ramp port = 1F
```

G - go run main program
K - clicks. uses 1E.1F.X0
M - assign ramp I/O port
P - output 1 ramp..X1=S1 etc.
R - display all registers
T - assign trigger port
X - set delay counter
Z - display command list
- continuous A/D-D/A
]

I - assign buffer I/O port
L - set buffer length
O - output 1 buffer (immediate)
Q - re-init buffer program
S - set buffer start
W - warm start WDOS
Y - init delays & buffers
* - input one buffer (immediate)
$ - continuous display A/D byte

The system prompt "]", when displayed, means that the system is ready to accept command input from the console.

SYSTEM OPERATION

<**CONSOLE INPUT**:>

There are three basic types of input accepted from the console. These are "numeric data" in hex, "subsystem commands" which are non-hex characters, and the "RETURN" key which initiates "command execution".

As an example, suppose that you wanted to output a click with a tempo of 60 beats per minute to port 1F. The typed sequence would be as follows.

0600KF<cr>

The numeric value is input as 4 digits, followed immediately by the sub-command to be executed and a one digit numeric value for the port number.

As data is typed on the keyboard, it is placed in one of three areas of memory. Any hex numbers entered from the keyboard following the system prompt are shoved onto a 4 character stack. Only the last 4 numbers typed are saved. Therefore;

if you meant to enter:	]2345
but you typed:	]2355
then just keep entering	
until the display is:	]23552345

Only the last 4 entries are used, even though all previous digits are still displayed on the screen.

Some commands require that only the sub-command be entered followed by a <cr>, while others require more information to operate correctly. If a required field is not input, but is required by the sub-command, the data previously entered into the buffer is used.

As an example, if you wanted to set the start of buffers 1, 2 and 3 to an address of 5000hex, the commands would be:

]5000S1<cr>
]S2<cr>
]S3<cr>

The sub-command used the 4 digit number placed on the stack during the first operation for the next two operations, because there was no new data pushed onto the stack before the next sub- command was entered.

Once the method used to process command and data entry is understood, it can greatly reduce the amount of time spent entering values into the various data areas. For instance, if you wanted to set the start of buffer 1 at 5000hex, the start of buffer 2 at 0000 and the length of buffer 2 to 0400hex, the entry could be done the long way:

]5000S1<cr>
]0000S2<cr>
]0400L2<cr>

Or the same results could be attained form the sequence:

]5000S1<cr>
]0S2<cr>
]400L<cr>

Take your pick, they both do the same thing.

SYSTEM OPERATION

<SUBSYSTEM COMMANDS:>
There are three subsystems that make up the total operating system in Wendel. These are:

DISK FILE EXECUTIVE
MAIN OPERATIONS PROGRAM
APPLICATION DEFINITION & SETUP

In the next three sections of this manual, each subsystem will be explained throughly, including explanation and examples for each command under the subsystem.

The DISK FILE EXECUTIVE subsystem does just what is implied. It enables the user to save newly created sounds, save tune setup data from a current work file, load special programs, load drum sounds from disk and all the things disk operating systems should be able to do.

The MAIN OPERATIONS PROGRAM is the meat of Wendel. This is the portion of the system that actually does the sequencing, playing, clicking, sampling and all around good things that Wendel was designed for.

The APPLICATION DEFINITION & SETUP program is an interactive module designed to enable the user to tell the computer what instruments to play, when to play them, what rhythm patterns to play and that kind of stuff. Believe me, it is much easier than without this subsystem.

(So turn the page!)

SYSTEM OPERATION

<**DISK FILE EXECUTIVE:>**
To enter the disk file executive from the main operations program, type the following:

]W<cr>

That is, in response to the "]" prompt character, the user must type a "W" followed by a "carriage return". The display on the CRT will be:

GROUP - 7 MDOS VS. X.X - COPYRIGHT 1981
>

Notice that in this subsystem that the prompt character is ">". The disk executive is now waiting for a command.

<ENTERING EXECUTIVE COMMANDS:>
Executive statements are entered by typing characters in sequence on the console keyboard. An executive statement is terminated by pressing the "RETURN" key. Error correction is handeled differently in this subsystem. There are two control features that enable the user to correct input errors.

<1:> Each time the "BACKSPACE" or "DELETE" key is pressed the next previously typed character will be deleted from the line.

<2:> Holding down the control key and typing X (^X) will cause all of the current line to be cancelled. The cursor will be positioned at the beginning of the next line to except entry of a new line.

<EXECUTIVE STATEMENT FORMAT:>
An executive statement has the following form:

[unit:]NAME ["ASCII" "ASCII" "ASCII" hex hex hex]

The NAME in an executive statement may be the name of an explicit command or the name of a disk file. There are 23 explicit commands allowed under this subsystem. Explicit commands are upper case only and must not be preceded by any spaces. In addition, executable machine language programs may be loaded from the disk and run just by entering the filename.

When an executive statement is entered the executive program searches for a match to the command. If the match is successful the command will be executed immediately. If the match is not made the executive will treat the command as an implicit command and must be found on disk. Implicit command filenames may be prefixed by an optional unit number for systems with multiple disk drives. No unit number need be specified for single drive systems. If a unit number is specified it must be seperated from the first character of the NAME by a colon (:). The executive processes the implicit command by searching the directory of the specified disk drive for the file. If the file is found and is of the right file type, it is loaded into memory and control is transfered to the program. If the file is not found on the unit specified, an error message is output to the console; COMMAND NOT FOUND. If the file is found in the disk but is not an executable file then the error message output to the console is; WRONG FILE TYPE.

Executive statements consist of a NAME followed by parameters, as necessary. Parameters can be ASCII or numeric. There can be up to four ASCII parameters and up to four numeric parameters. There must be at least one space between the NAME and any parameters. All parameters must be seperated from each other by at least one space. Entry of an executive statement with too many parameters of either type, or without the required spaces between fields will result in a SYNTAX ERROR.

ASCII parameters consist of from 0 to 10 ASCII characters. These may be any character produced by the keyboard except "BACKSPACE", "DELETE" and double quotation marks. ASCII parameters must be enclaose in balanced double quotation marks. Entry of an executive statement with unbalanced qoutation marks or illegal characters in an ASCII parameter will result in SYNTAX ERROR.

ASCII parameters in executive statements are generally used to spacify disk filenames. In usage a unit number may be prefixed to the ASCII filename within the quotation marks by typing the unit number followed immediately by a colon (:). If no unit is spacified, unit 0 or the single drive in a one drive system is assumed. For example, "WENDEL11" and "1:WENDEL11" are bothe valid ASCII parameters in an executive statement.

Numeric parameters in an executive statement are unsigned hexadedimal values from 0 to FFFF. They represent such elements as memory addresses, filetypes, and databytes. Entry of a numeric parameter greater than FFFF or with illegal characters will result in a SYNTAX ERROR.

<CANCELLING AN OPERATION:>

All explicit commands can be cancelled in progress by holding down the control key and typing a "C" (^C) on the keyboard. The operation will be terminated as soon as the ^C is recognized and the message CANCELLED will be output to the console. Control is returned to the executive.

<DISPLAY CONTROL:>

All explicit commands can be temporarily stopped in progress by holding down the control key and typing "S" (^S). The process will stop upon recognition of the ^S. Typing any key other than ^S or ^C will cause the process to resume. This function is very useful in controlling commands that output display at high speed. For example, output of the DUMP command may be viewed at reading speed by stopping and resuming the output as necessary.

<EXPLICIT EXECUTIVE COMMANDS:>

Command syntax for each of the explicit commands is illustrated in this section with the aid of the following notation:

[ ] Option brackets. Any parameters enclosed between brackets are optional.

< > Symbol brackets. This space should be replaced by the item described.

<The COMP command:>

COMP <start addr. block1> <end addr. block1> <start addr. block2>

The COMP command compares two blocks of memory and displays address locations that do not compare and the data at those locations. Example:

```
>COMP 5000 500F 5010
5004 01 09 5014
```

The block of memory from 5000 to 500F is compared with the block of memory from 5010 to 501F. One location fails to compare. Location 5004 contains 01 while the corresponding location, 5014, in the second block contains 09.

<The DUMP command:>

DUMP <start addr.>[<end addr.>]

The DUMP command outputs to the system console a formatted hex display of the contents of a block of memory. Sequential memory locations are shown 16 to a line with the memory address at the left margin. If the optional end address parameter is not entered, only one byte is displayed. Example:

```
>DUMP 5000 5011
5000  50 C0 27 77  4F 33 4F CD  7D 9E 98 00  6A FD 82 90
5010  77 2B
```

<The ENTR command:>

ENTR <start addr.>

The ENTR command allows data to be entered into memory directly from the console device. Example:

```
>ENTR 7000
>78 89
6F /
```

Three bytes were entered starting at location 7000 hex. These were 78 at 7000, 89 at 7001 and 6F at location 7002.

Typing in an ENTR command places the executive in a special enter mode. While in the enter mode each line of values that is typed is entered into memory when the RETURN key is pressed. Until the RETURN key is pressed the standard backspacing and ^X tools are available for line correction. The last value on the last line must be followed by a slash (/) to properly terminate the enter mode. Entry of a illegal hex value in any line will also cause termination of the enter mode with the message SYNTAX ERROR.

<The FILL command:>
FILL <start addr.> <end addr.> <byte>

The FILL command fills a block of memory with a specified byte. Example:

>FILL 7000 8000 9

Each byte of memory in the block from 7000 to 8000 is changed to a 09 by this command.

<The MOVE command:>
MOVE <source addr. start> <source addr. end> <dest. addr. start>

The MOVE command copies the source block of memory to the destination block. The source block is not changed. The destination block is changed to be an exact copy of the source block. Example:

>MOVE 3000 4000 7000

Each byte in the memory block from 3000 to 4000 is copied into the corresponding position in the memory block from 7000 to 8000.

<The SEAR command:>
SEAR <start addr.> <end addr.> <byte>

The SEAR command searches a block of memory for all occurrences of the specified byte and displays all locations with a match. Example:

>SEAR 3000 3020 9F
3004 9F
3018 9F

The block of memory from 3000 to 3020 is searched for all occurrences of a 9F. Location 3004 and location 3018 both contain 9F. No other locations in the block contain 9F.

<The SEARN command:>
SEARN <start addr.> <end addr.> <byte>

The SEARN command searches a block of memory for all non-occurrences of a specified byte and displays all locations that do not match. Example:

```
>SEARN 3000 3010 67
3002 09 67
3006 76 67
```

The block of memory from 3000 to 3010 is searched for all non- matches with the mask 67. Location 3002 contained a 9 rather than a 67, and 3006 contained a 76 rather than a 67.

<The CREATE command:>
CREATE "[unit:]<filename>" [<file type>]

The CREATE command creates a new file in the directory of the diskette in the specified unit and allocates the initial track for the file. If no unit is specified, unit 0 is assumed. The second parameter optionally gives the file a TYPE designation. If no type is specified the type is defaulted to 0.

<The DISP command:>
DISP "[unit:]<filename>" [<record number>]

The DISP command outputs a formatted hex display of the data contents of a file to the system console. The unit number indicates the disk drive on which the file is to be found. If no unit is specified, unit 0 is assumed. The optional record number indicates on which record in the file the display is to begin. If no record number is specified, record 1 is assumed.

Each record is displayed with a header line that contains the record number, the address in memory where the record is to be loaded, and the number of data bytes in the record. Data lines follow the record header. Each data line has up to sixteen data bytes preceded by the index position in the record of the first data byte on that line.

```
>DISP "1:TEST" 29
0029  3C00 0022
00  12 2A BD 76  8F ED 54 41  89 00 00 82  BC CC 76 89
10  78 88 3B BB  88 54 58 56  90 88 32 31  30 0D 00 00
20  89 55
002A  3C80 0003
00  FF FF FF
002B  3F00 0009
00  45 43 4B 4C  31 37 38 0D 00
002C  2B00 000
END-FILE
```

<The FILES command:>
FILES [<unit>]

The FILES command outputs a formatted display of the file information in a diskette directory to the system console. The unit number indicates which disk drive directory is to be displayed. If no unit is specified, unit 0 is assumed. Example:

```
>FILES 1
```

DIR	03	0000
RES	03	0013
MDOS	0F	001C
LINEEDIT	15	000C
ASSM	15	0010
SYMSAVE15	0003	
FILECOPY15	0003	
DISKCOPY	0F	0009
BASIC	0F	004B

The files on drive one are displayed on the console. The left column contains the filename, the second column is the file type, and the third column contains the number of sectors the file uses. All numbers are in hex.

<The FREE command:>
FREE [<unit>]

The FREE command outputs to the system console the number of tracks left unallocated (free) on a diskette. The unit number indicates which disk drive. If no unit is specified, unit 0 is assumed. Example:

>FREE 1
003B

The diskette on drive one has 3B tracks available to be allocated.

<The SCRATCH command:>
SCRATCH "[unit:]<filename>"

The SCRATCH command removes a named file from the directory of a diskette and returns its allocated tracks to available status. Disk drive 0 is assumed if no unit is specified.

Note: Some files cannot be SCRATCHed without first changing the file TYPE.

<The LOAD command:>
The LOAD command loads (reads) a named file from a diskette into the computers memory and then returns control to the executive. If no unit number is specified, the file is expected to be found on unit 0.

The LOAD command can be used in conjunction with two categories of files, OBJECT files and DATA files. The specific nature of the load that is performed depends on the category of the specified file to be loaded. The process of LOADing an OBJECT file is described in THE LOAD COMMAND FOR OBJECT FILES. The process for LOADing a DATA file is described in THE LOAD COMMAND FOR DATA FILES.

<The LOAD command for object files:>
An OBJECT file is defined as any file with a file type value in the range 08 - 0B hex or 14 - 1B hex. These ranges include ASSM object files, BASIC 'save memory' files, executable system files, and executable user files.

The format of the LOAD command for OBJECT files is:

LOAD "[unit:] <filename>" [<start addr.>]

OBJECT files are LOADed by using the address and length information in the header of each record of the file. This is called a 'scatter load' because it permits records in the file to be loaded into non-contiguous portions of memory depending on the associated addresses. The LOAD is terminated when the first 0 length record in the file is encountered.

If the load address of the first record is less than 2B00 hex, the message LOAD ADDRESS ERROR is displayed because file may not be loaded beneath the application area.

If the optional start-address is specified in the LOAD command, then the first record of the file is loaded starting at the specified address. The load address in the record header of the first record is subtracted from the start-address to produce an offset. When the records following the first record of the file are loaded, the calculated offset is added to the load address in the record header and the record is loaded starting at the calculated address. This is called an 'offset scatter load'.

If the optional start-address is less than 2B00 the message LOAD ADDRESS ERROR is displayed.

<The LOAD command for data files:>
Any file which is not an OBJECT file and not an OVERLAY file is treated as a DATA file by the LOAD command. DATA files thereby include file type values in the ranges 0-7, 10-13 hex, and 1C-FF hex. These ranges cover BASIS DATA files, ASSM and LINEEDIT source files, BASIC program files and all of the unassigned file types.

The format of the LOAD command for DATA files is:

LOAD "[unit:] <filename>" <start addr.>

The start address parameter is mandatory. If a start address is not specified a SYNTAX ERROR message will be displayed. If the start address is less than 2B00 HEX a LOAD ADDRESS ERROR will result. This prevents accidental destruction of the operating system.

Data is loaded starting at the specified address and continuing until the number of records in the file as shown in the directory have been loaded. The data is loaded into memory sequentially and contiguously. Only the number of data bytes in each record are loaded. The LOAD command does not pad records of less than 256 bytes. If a file were loaded at location 3000 and the first record had only 4 data bytes in it, then the first data byte from the next record would be loaded at location 3004. Records with zero length are skipped over. The load address in the sector header has no meaning when doing a data LOAD.

<The SAVE command:>
SAVE "[unit:]<filename>" <start addr.> <end addr.> [<file type>] [<exec. addr.>]

The SAVE command saves (writes) a new file to a diskette from a block of memory. The file is written sequentially from the memory start address through the memory end address into full sequential records. If no unit number is specified, the file is written to unit 0. If a file type is not specified, the execution address of the file will be set to the start address of the memory block. Note that the type and execution address parameters are position dependent such that if an execution address is specified then a file must also be present. Example:

>SAVE "1:NEWFILE" 2B00 3700 0 3000

A file is created on the diskette in drive one with the name NEWFILE and the memory block from 2B00 to 3700 is written to that file. The file is given a type of 0 and the execution address saved with the file is 3000. If no execution address had been specified then 2B00 would be saved as the execution address.

<The RENAME command:>
RENAME "[unit:]<filename>" "new name>"

The RENAME command changes the name of a diskette file to a specified new name. If no unit number is specified, the file to be renamed is expected to be found on unit 0. Example:

>RENAME "1:OLDFILE" "NEWFILE"

The file named OLDFILE on the diskette in drive one is changed to NEWFILE on the diskette in drive one. The file type is unchanged by the renaming process.

<The TYPE command:>
TYPE "[unit:]<filename>" <type>

The TYPE command changes the type designation of a specified file. The type designation is a single hex byte. Example:

>TYPE "1:PROGRAMX" 15

The type of the file PROGRAMX on disk drive one is changed to a value of 15.

<The APP command:>
APP ["<ASCII>"..."<ASCII>"] [<hex> <hex>...<hex>]

The APP command transfers program control from the executive to the start of the applications area at 2B00 hex. It expects a valid executable program to be in the applications area with its entry point at the beginning. Up to four ASCII parameters and four hex parameters can be passed to the program. For example, if you are doing several assemblies, the assembler need only be read into memory once from diskette as it does not change itself in the process of assembling a program. After it is once in memory the APP command can be used to communicate with the assembler. Example:

>APP "1:SOURCE" "OBJECT" "P"

If the assembler were already in memory, the above example would transfer control and the necessary parameters to the progam and the assembler would assemble the source file called SOURCE from drive one; produce an object file on drive 0 called OBJECT; and output a paginated listing on the print device.

The APP command functions like the EXEC command in that it PUSHes the address of the operating systems warm start entry point onto the system stack. Therefore if the program in the applications area does not provide its own stack, a RET would return control to the operating system.

<The EXEC command:>
EXEC <address>

The EXEC command transfers processor control directly to the specified memory address. It expects a valid program to begin at that address. The address of the operating systems warm start entry point is PUSHed onto the 8080's hardware stack by the EXEC command. Therefore, if the executed program does not set its own stack, a final RET in the program will return to the operating system. This feature allows subroutines to be exercised separate of the rest of a system under development.

<The MATH command:>
MATH <hex number> <hex number>

The MATH command performs 16 bit integer math functions on the two specified hex numbers. It displays the sum, difference, product, quotient, and modulus. Example:

>MATH 4 5
0009 FFFF 0014 0000 0004

The results are displayed from left to right: 4+5=9 ; 4-5=FFFF ; 4*5=14 ; 4/5=0 (integer division) and a remainder (modulus) of 4.

<PROMPT "<ASCII>":>
The PROMPT command sets the executive prompt string to the value of the ASCII string. The string can be up to ten characters long. Spaces are not allowed. The prompt is initially > when the system is configured. Example:

```
>PROMPT "**"
**
```

The prompt is changed from > to a **

<The INIT command:>

INIT <unit>

The INIT command initializes a diskette in the specified drive. The drive unit number must be specified. The INIT command formats the diskette by writing an empty block with the correct track and sector identification on every sector of the diskette and reading each sector to verify the media. It creates a blank directory and places a system loader on the diskette. The INIT command essentially cleans the diskette of any data previously on the diskette and prepares it for new use. Accidental use of the INIT command could destroy the entire content of a diskette. Therefore, the system prompts ARE YOU SURE?. It waits for a 'Y' or 'N' response to indicate a yes or no. An 'N' cancels the command without doing any damage. Example:

```
INIT 1
ARE YOU SURE?
```

The diskette on drive one will be initialized if a 'Y' if typed. All other replys will result in the command being canceled. Control returns to the executive.

SYSTEM OPERATION

MAIN OPERATIONS PROGRAM

The MAIN OPERATIONS PROGRAM is active upon initial system start- up. The MAIN OPERATIONS PROGRAM can also be entered from the DISK FILE EXECUTIVE. This is done in one of two ways. First by entrering the implicit command from the console in response to the system prompt as follows:

>WENDEL11<cr>

The system diskette must be in the disk drive unit. The MAIN OPERATIONS PROGRAM will be loaded into system memory from the disk and executed.

The second method is by typing the explicit command from the console in response to the system prompt as follows:

>APP<cr>

The MAIN OPERATIONS PROGRAM must already reside in system memory to invoke this command. This method is often used to jump back and forth between the DISK FILE EXECUTIVE and the MAIN OPERATIONS PROGRAM. The system diskette does not need to be present for this method.

If you have performed some action while in the DISK FILE EXECUTIVE which has erased the MAIN OPERATIONS PROGRAM from system memory, then you must invoke the implicit command as stated in the section previous.

The MAIN OPERATIONS PROGRAM is the meat of the system. This is the program that outputs the drum sounds to the console, samples in sounds from the outside world, sequences the synthisizers, plays the click track etc. All the good stuff, you know what I mean?

The MAIN OPERATIONS PROGRAM will sign on with the following CRT display:

```
   X0    X1    X2    X3    X4    X5    X6    X7    X8    X9    XA    XB    XC    XD    XE    XF
  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF

  S-O-L       S-1-L       S-2-L       S-3-L       S-4-L       S-5-L       S-6-L       S-7-L
 4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001

  S-8-L       S-9-L       S-A-L       S-B-L       S-C-L       S-D-L       S-E-L       S-F-L
 4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001

      buff port = 1F              ramp port = 1F

G R O U P - 7  S Y S T E M S
(C) Copyright 1980
All rights reserved
>>>Buffer11 ver X.xxxx. .vxxx..ready
]
```

<CRT DISPLAY:>
The CRT display is devided into two halves. The upper portion of the CRT does not scroll, or move off the screen at any time. This displays the current values used for timing delays, sound buffer beginning addresses and lengths and current ramp and buffer port assignments. The lower half of the screen displays console messages from the computer, and echos console input from the user.

An important feature of the lower half display is the way in which the data entered scrolls, or moves up the screen as new data is input from the keyboard. When a command is typed follwed by a <cr>, the cursor does not move to the beginning of the next line, but to the middle of the screen on the same line. This allows 24 previous commands to be viewed as opposed to only 12 if the bottom half of the screen scrolled up in the normal manner.

The top half of the screen is devided into three sections. The upper section displays the data used for delays. There are 16 delay counters numbered 0 thru F. Each counter contains four digits in hexadecimal notation. As the data in the internal delay counters changes, so does the display.

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

The center section displays the data for each of 16 internal sound buffer areas. The center section takes two rows on the screen because there is twice as much data to display. The data is buffer start addresses, located under the "S", and buffer length, located under the "L". (Notice how well that seems to work out?). The number of the buffer is located between the "S" and the "L" like this:

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001
S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

This display shows that all buffers start at 4000 hex and are only 0001 byte long. This is the normal initial set-up condition

The lower section displays the current assignable port through which the drum sounds will emerge, and looks like this:

buff port = 1F ramp port = 1F

The sign-on message only appears when the MAIN OPERATIONS PROGRAM is entered from initial system startup or upon return from the DISK FILE EXECUTIVE.

<SUBSYSTEM COMMANDS:>
There are several subsystem commands associated with the MAIN OPERATIONS PROGRAM. These will each be discussed in detail with examples of the command entry and a sample CRT display for each command. Multiple commands including short form commands will be used where possible. Each CRT display will include all previous commands as it would on the actual CRT.

The lower half of the CRT display below shows a list of the subsystem commands.

```
   X0   X1   X2   X3   X4   X5   X6   X7   X8   X9   XA   XB   XC   XD   XE   XF
 FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF

  S-O-L      S-1-L      S-2-L      S-3-L      S-4-L      S-5-L      S-6-L      S-7-L
4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001

  S-8-L      S-9-L      S-A-L      S-B-L      S-C-L      S-D-L      S-E-L      S-F-L
4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001

      buff port = 1F             ramp port = 1F
```

G - go run main program
K - clicks. uses 1E.1F.X0
M - assign ramp I/O port
P - output 1 ramp..X1=S1 etc.
R - display all registers
T - assign trigger port
X - set delay counter
Z - display command list
- continuous A/D-D/A
]
I - assign buffer I/O port
L - set buffer length
O - output 1 buffer (immediate)
Q - re-init buffer program
S - set buffer start
W - warm start WDOS
Y - init delays & buffers
* - input one buffer (immediate)
$ - continuous display A/D byte

Z - display command list:
Typing the following in response to the system prompt "]" will result in the above display of the command list.

```
]Z<cr>
```

X - set delay counter:
This command changes the value of the delay counter. The delay counter is used to set the tempo of the click track and to vary the amount of delay before outputing a drum or percussion sound, or to vary the delay between the trigger and a sequence pulse to a synthisizer. The command is issued by entering the amount of delay in hexidecimal followed by the command "X" and the number of the specific delay counter to be set. An example command sequence is:

```
]6700X3<cr>
]X6<cr>
]X8<cr>
```

This command would place the value 6700 hex into delay counters number 3, 6, and 8. Note that the higher the number, the shorter the delay time. That is, FFFF would be the shortest amount of time (no delay) and 0000 would be the longest delay possible.

The top half of the screen would immediately update to show the change to delay counters number 3, 6, and 8 and the cursor would advance to the next portion of the screen and re-display the cursor.

```
  X0   X1   X2   X3   X4   X5   X6   X7   X8   X9   XA   XB   XC   XD   XE   XF
FFFF FFFF FFFF 6700 FFFF FFFF 6700 FFFF 6700 FFFF FFFF FFFF FFFF FFFF FFFF FFFF

 S-O-L      S-1-L      S-2-L      S-3-L      S-4-L      S-5-L      S-6-L      S-7-L
4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001

 S-8-L      S-9-L      S-A-L      S-B-L      S-C-L      S-D-L      S-E-L      S-F-L
4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001  4000 0001

      buff port = 1F              ramp port = 1F

]6700X3                                  ]X6
]X8                                         ]
```

S - set buffer start:
This command changes the value of the buffer start address. This data is used by the drum program so that it knows where in system memory to get the drum sound you want to put out. The command is issued by entering the starting address of the drum sound followed by the command "S" and the number of the buffer to which that address is to be assigned. An example of the command sequence would be:

```
]B400S1<cr>
]S2<cr>
]C000S5<cr>
```

This command would place the address B400 hex into buffer start number 1 and buffer start number 2, and place the address C000 hex into buffer start number 5.

The top half of the screen will be updated as follows:

```
   X0    X1    X2    X3    X4    X5    X6    X7    X8    X9    XA    XB    XC    XD    XE    XF
 FFFF  FFFF  FFFF  6700  FFFF  FFFF  6700  FFFF  6700  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF

  S-O-L       S-1-L       S-2-L       S-3-L       S-4-L       S-5-L       S-6-L       S-7-L
4000 0001   B400 0001   B400 0001   4000 0001   4000 0001   C000 0001   4000 0001   4000 0001

  S-8-L       S-9-L       S-A-L       S-B-L       S-C-L       S-D-L       S-E-L       S-F-L
4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001

      buff port = 1F              ramp port = 1F
```

```
]6700X3                                    ]X6
]X8                                        ]B400S1
]S2                                        ]C000S5
]
```

L - set buffer length:
This command changes the value of the buffer length. This data is used in conjunction with the buffer start address to tell the drum program how much data to output during a drum sound. The command is issued by entering the length of the buffer followed by the command "L" and the number of the buffer whose length you are changing. An example of the command would be:

]3100L1<cr>
]1A00L2<cr>
]L5<cr>

This command would place the length 3100 hex into buffer 1, and the length 1A00 hex into buffers 2 and 5. The screen would show the changes thus:

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1F ramp port = 1F

]6700X3
]X8
]S2
]3100L1
]L5

]X6
]B400S1
]C000S5
]1A00L2
]

I - assign buffer I/O port:
This command manually assigns the current buffer output port. This command is issued by entering the command "I" followed by the last digit of the desired port. The valid range of port assignments is from 19 to 1F. Any value entered less than 9 will default to a 9. To assign I/O port 1C the command sequence would be:

]IC<cr>

M - assign ramp port:
This command manually assigns the current ramp I/O port. The ramp port is used for synthisizer VCA and ENVELOPE generation. The forms and limits of the command are the same as buffer I/O above. An example of assigning ramp port 19 would be:

]M9<cr>

CRT display update would be:

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

```
]6700X3
]X8
]S2
]3100L1
]L5
]M9
```

```
]X6
    ]B400S1
    ]C000S5
    ]1A00L2
      ]IC
      ]
```

O - output 1 buffer (immediate):
This is the manual command for Wendel. When this mode is entered, the system will manually output a sound each time a key on the numeric key pad is typed. If you type "0", the computer will output the sound assigned to buffer 0 out port 1F. If you type "1", the computer will output the sound assigned to buffer 1 out port 1E. This allows you to manually play drum sounds and have each sound appear at a different fader for mixing and equalization.

KEY TYPED	BUFFER	PORT
0	0	1F
1	1	1E
2	2	1D
3	3	1C
4	4	1B
5	5	1A
6-F	6-F	19

To enter manual mode type "O<cr>". To exit type <cr> or any non hex key. When in manual there is no prompt and a message is displayed as illustrated below.

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

]6700X3]X6
]X8]B400S1
]S2]C000S5
]3100L1]1A00L2
]L5]IC
]M9]O - output 1 buffer (immediate)
]

*** - input one buffer (immediate):**
This is the command that is used to input drum sounds manually into Wendel. When this mode is entered, Wendel will input the sound presented to the currently assigned I/O port and store the information in the specified buffer area. After entering the input mode, each subsequent typing of the return key will again sample the input port into the buffer, overwriting the data existing in the buffer previously.

It is always a good idea when inputing data to have a totally empty buffer area, and assign a starting address of 4000 and a length of 9FFF to the buffer used. This will help in case you are a little slow on the trigger when inputing new drum sounds. The command sequence to input one buffer full of data from the currently assigned I/O port to buffer 1 would be:

]*1<cr>

The command sequence is terminated by typing any non hex key, such as the space bar.

CRT display update would be:

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

]6700X3]X6
]X8]B400S1
]S2]C000S5
]3100L1]1A00L2
]L5]IC
]M9]O - output 1 buffer (immediate)
]*1 - input one buffer (immediate)]

P - output 1 ramp..X1=S1 etc.:
This command initiates one ramp output to the currently assigned ramp port for single shot sequences to a synthisizer. This command is usefull when testing VCA or envelope ramps one at a time to make sure that they are of the proper duration before running a program.

The command sequence consists of the "P" command followed by the number in hex of the ramp to be output. An example of outputing ramp 4 to the assigned ramp port would be:

]P4<cr>

CRT display update would be:

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

6700X3]X6
]X8]B400S1
]S2]C000S5
]3100L1]1A00L2
]L5]IC
]M9]O - output 1 buffer (immediate)
]*1 - input one buffer (immediate)]P4 - output 1 ramp..X1=S1 etc.
]

T - assign trigger port:
The assign trigger port command changes the assignment of the trigger input to a port of your choosing. The normally assigned trigger port is 19 hex. The trigger signal conditioning circuits are physically tied to this input port, and under almost all circumstances the assignment should be left to this port. If there are some applications where it becomes necessary to bypass the trigger signal conditioning circuits, then this command is used to re-assign the port to the desired value. The command sequence is:

]TE<cr>

This sequence would assign the trigger port to 1E hex.

CRT display update would be:

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

]6700X3]X6
]X8]B400S1
]S2]C000S5
]3100L1]1A00L2
]L5]IC
]M9]O - output 1 buffer (immediate)
]*1 - input one buffer (immediate)]P4 - output 1 ramp..X1=S1 etc.
]

Y - init delays & buffers:
The "Y" command initializes all of the delays to a minimum delay value of FFFF hex, initializes all of the buffer start addresses to 4000 hex and initializes all of the buffer lengths to 0001 hex. This is used to clear out all of the garbage left from a previously run program. The command sequence is:

]Y<cr>

R - display all registers:
The "R" command updates the top portion of the screen to reflect the values currently in effect. During program execution the screen is not updated due to the relatively large amount of time it takes compared to the program operations that are being performed. Therefore the "R" command allows the operator to update the display at a more convenient time. The command sequence is:

]R<cr>

Q - re-init buffer program:
The "Q" command is the only one that will allow you to exit from a running program. After exiting from a running program, the top half of the screen is updated to reflect the current data values, the bottom half of the screen is cleared and the system prompt is displayed. The command sequence is:

]Q<cr>

CRT display update would be:

```
   X0    X1    X2    X3    X4    X5    X6    X7    X8    X9    XA    XB    XC    XD    XE    XF
 FFFF  FFFF  FFFF  6700  FFFF  FFFF  6700  FFFF  6700  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF  FFFF

  S-O-L       S-1-L       S-2-L       S-3-L       S-4-L       S-5-L       S-6-L       S-7-L
4000 0001   B400 3100   B400 1A00   4000 0001   4000 0001   C000 1A00   4000 0001   4000 0001

  S-8-L       S-9-L       S-A-L       S-B-L       S-C-L       S-D-L       S-E-L       S-F-L
4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001   4000 0001

      buff port = 1C              ramp port = 19

>>>Buffer11 ver X.xxxx. .vxxxx..ready
]
```

$ - continuous display A/D byte:
The "$" command is used to check the level of the in-comming A / D conversion. The value of the byte input from the currently assigned I / O port is displayed in real time in the upper left hand corner of the top half of the screen. The command sequence is terminated by typing any key on the keyboard. The command entry is as follows:

]$<cr>

CRT display update would be:

current A / D byte = F6

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

>>>Buffer11 ver X.xxxx. .vxxxx..ready
]$ - continuous display A / D byte]

 # - continuous A/D-D/A:
The "#" command is a couninuous flow of data into the currently assigned I/O port input, and out through the currently assigned I/O output port. This allows the operator to listen to the sound that is being digitized to adjust the input level for optimum signal to noise level. This is similar to recording on a tape machine and monitoring the output. The command sequence is:

]#<cr>

Any key typed will terminate this command sequence.

CRT display update would be:

current A/D byte = F6

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

>>>Buffer11 ver X.xxxx. .vxxxx..ready
]$ - continuous display A/D byte]# - continuous A/D-D/A
]

W - warm start WDOS:
The "W" command is used to jump from the MAIN OPERATIONS PROGRAM to the DISK FILE EXECUTIVE. The command sequence is:

]W<cr>

To return to the MAIN OPERATIONS PROGRAM, type APP<cr> in response to the DISK EXECUTIVE prompt ">".

CRT display after the "W" command will be:

GROUP - 7 MDOS VS. X.X - COPYRIGHT 1981
>

The commands available are listed under SYSTEM OPERATION, DISK FILE EXECUTIVE in this manual.

G - go run main program:
The "G" command executes a system program for playing drum sounds by following the trigger from an external source. Thorough explanation of this command will be found under SYSTEM OPERATION, USER PROGRAMMING and RUNNING USER PROGRAMS. The command sequence is:

]G<cr>

The cursor will remain at the end of the prompt line until one bar of music has passed, then the cursor will advance to the beginning of the next command line after the prompt. This tells the operator that Wendel is receiving the trigger information properly.

While Wendel is executing a user program, any of the MAIN OPERATIONS PROGRAM commands may be executed. The command is entered with the normal command sequence, but the command will not be executed until the end of the current bar of music being played. When the command is executed, the cursor will advance to the beginning of the next command line and the prompt will be displayed.

CRT display update would be:

current A / D byte = F6

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	XA	XB	XC	XD	XE	XF
FFFF	FFFF	FFFF	6700	FFFF	FFFF	6700	FFFF	6700	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF

S-O-L	S-1-L	S-2-L	S-3-L	S-4-L	S-5-L	S-6-L	S-7-L
4000 0001	B400 3100	B400 1A00	4000 0001	4000 0001	C000 1A00	4000 0001	4000 0001

S-8-L	S-9-L	S-A-L	S-B-L	S-C-L	S-D-L	S-E-L	S-F-L
4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001	4000 0001

buff port = 1C ramp port = 19

>>>Buffer11 ver X.xxxx. .vxxxx..ready
]$ - continuous display A / D byte]# - continuous A / D-D / A
]

This concludes the description of the MAIN OPERATIONS PROGRAM commands. A word of caution is in order for commands entered during the ronning of a USER PROGRAM. The "R" command takes a significant amount of time to perform, and may have an effect on the performance of the program being run. Usually other commands entered during a running USER PROGRAM are during the testing and fine tuning stages, so a missplaced beat is secondary to the command entry that is being performed. Most commands, such as the "S", "X", and "L" commands will have no adverse effects on the running program. These will be entered to change the pre- programmed delays and buffer lengths to get "just the sound you were looking for".

Experimentation is encouraged. Wendel was developed by experimenting with different parameters with a "what if" attitude. Your imagination is the only limit to the number of things Wendel is capable of doing.

Keep up the good work.