TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

D O C U M E N T A T I O N R E V I S I O N S

02/10/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future.

2.0 SIGNIFICANT CHANGES

2.1 MULTIPLE SCANS

The most important change in this software release is the addition of multiple scan commands on the same line. This will allow for more than one channel type to be transmitted in a single burst. In radio based half-duplex systems, this will greatly speed the overall update rate in systems where there are a small number of channels in each type.

For example, a system with status, output, analog, meter, and counter channels would have previously required a download sequence as follows:

```
SCAN s1:s8 R
SCAN o1:o8 R
SCAN a1:a8 E
SCAN m1:m2 E
SCAN c1:c2 E@
```

The same results can now be obtained with a command like:

```
SCAN s1:s8 R o1:o8 R a1:a8 E m1:m2 E c2:c2 E@
```

The system will generate a single, long DATA response that has all of the channel types combined in one line.

WHen using this option, design the scan line so that the resulting DATA line will have 240 or less characters. If communications are poor, better results may be obtained by only using 1 or 2 channel types in a single SCAN command. This is

because the poor comm links often have trouble sending bursts of more than 2 or 3 seconds.

2.2 AGA3 METER ACCURACY

Problems with AGA3 meter accuracy in downstream, pipe tap applications have also been corrected. The method of calculations is also changed from earlier versions in that a separate AGA3 command is not needed for each calculation. The meters are calculated in real-time by the alarm scanning task.

2.3 NEGATIVE ANALOG CHANNELS

Analog channels now have a new setup option that will control the conversion of negative input values. This defaults to FALSE, and will cause numbers that otherwise would be less than zero to appear as zero. This cleans up displays that show small negative numbers when the transmitter is just below the calibrated zero point, as is often the case with delta-P and static-P transmitters. If negative numbers are desired, set this option to TRUE during channel programming.

This only affects the raw signal conversion, and a negative value can still be placed in a channel with a CALC or DATA statement.

2.4 DUMP SCREEN

A command to print the local CRT screen has bee added to allow for custom displays to be printed. To use this command, the SET PRINT command must have been entered at some time to allow local printing. The command will use the system's bios print screen driver, and will follow the screen image with a line telling the system name, rtu name, and time and date of the printout.

This command will be useful for a command file that clears the screen, paints a custom display using SAY and CURSOR commands, and then does the print screen.

2.5 DECIMAL PLACE CONTROL

New SET parameters have been included to adjust the number of decimals used in the data displays, reports, and DATA statements. The SET DISPLAY command affects displays and reports, and the SET DATA command controls the format of real numbers in the DATA responses to SCAN commands. The default is for 3 decimal places, but some systems may appear better with 2 or 1 places.

The parameter given in these commands is the same as the format codes illustrated in the SAY and CURSOR commands. For example, to set the display to a max of 6 digits with no more than 2 decimal places, use SET DISPLAY F6D2.

Note that each task has it's own settings for these parameters.

2.6 ALTERNATE COMM PORTS

A SET PORT command has been provided to allow selection of a COM port that has a different number than the task that is using it. In the past, COM1 had to be used by Task 1. This is still the default, but a SET PORT x command can now be placed in the startup file for a task to allow selection of an alternate port.

The port must still be selected for use in the main DAT file processed with the system is started.

2.7 ALARM DELAYS

Alarm delay times are now allowed for all channel types, including outputs. This allows an output to be turned on by any method, and go into "alarm" at some future time if it remains on long enough. One application is to have the output turn itself off by preparing a file to do just that. This file can be processed by the output when it goes into alarm by setting the "RUN COMMAND FILE" attribute for that channel during setup.

2.8 CHANNEL VALUE SETTING

All channel types can now be set to any value using a DATA or CALC command. Also, channels that have time and dates associated with them (i.e. counters and totalizers) will have the time and data automatically set to current time/date when ever these channels are set to exactly 0. This allows for daily processes to clear the counters and totalizers and reset the starting time in a single command. A separate RESET is not needed any longer.

2.9 ELEMENTARY STRING SUPPORT

Basic support for string (text) data is now provided. Each RTU task has 10 string variables, %0 thru %9. These variables can be set with simple assignment statements, and can be set by the operator with the new INPUT command. These strings can be referenced in displays and messages with the new $% operator, similar to the older $T, $D, and other $ operators.

This string support will be expanded in the future to allow for string operations.

2.10 INPUT Command

An INPUT command similar to that in BASIC has been implemented to allow for prompted operator entry of channel values and string data. This is intended to simplify routine changes to orifice plate sizes and other value channels by allowing direct entry of a new value. The sequences can be placed in command files, and these files executed in response to menu hits. This will assist in reducing operator training and improving system operation.

2.11 AUTO MENU RETURN

Older versions required a specific MENU command to be placed in each menu sequence that desired a return to the menu after completion. Automatic return is now the default, so menu files that have separate MENU lines at the end of each sequence must be edited to remove these lines. If an auto return is not desired after a menu hit, the last line of that hit's sequence should be FORCE CLEAR 0. This will cause the waiting MENU command in the task's input queue to be removed so that the menu will not re-appear. This is often desired when a sequence is started that sends messages to the console, as when forcing a comm task to do a callout and shutin program.

Note that menu's having cursor positioning codes in the title line of the menu file should have the codes as COL and ROW, not the reverse as some earlier documentation describes.

3.0 COMMAND DESCRIPTIONS

The following sections will appear as new or replacement sections in future manuals.

AGA3 Cause re-calculation of AGA-3 meter channels (0).

The AGA3 command is used to force re-calculation of a range of meters. This is required for meters that are not local to the system processing a command, or for meters that have long partial recalculation times.

AGA3 calculations are processor intensive, and the program allows for partial calculations of important variables to occur at all times. Other parameters that rarely change are calculated less often to produce the best response. Using the AGA3 command will force a complete recalculation of the meter, regardless of the meter's recalc frequency.

For example, a host computer only needs to re-calculate when new data comes in over the communications line. In this case, an AGA3 command line can be placed in the BYE file so that only one re-calc takes place per transmission.

```
Example:  AGA3 M1
          AGA3  M2:m3


          DUMP RTU
```

DUMP Show I/O, RTU, VARIABLE data (1).

This command is used to get a configuration display on the I/O point system, individual RTU setups, or an RTU's variable data. The options are:

```
TASK x  - Current or specified TASK.
RTU x     Current or specified RTU.
LINK x    Specified comm LINK.
GROUP x   Specified comm link grouping.
IO x      All or specified I/O drivers and maps.
VAR       Task variables used by expression evaluator.
COMM x    Comm port x
CHAN x    Text line for a single channel or channel range.
ATTR      CRT attributes, using example displays.
```

```
Example:  DUMP C1:C8
          DUMP Io
          DUMP RTU
```

ECHO Send the rest of the line to the comm line (1).

When the program is reading a command file, it can use this command to send a line of text to the communications channel. In the case of the local console, this is useful for displaying status messages from within a command file. For remote terminals, this command can be used to send configuration information to the communications controller (modem or PRC).

This command sends the line unaltered except for numeric values entered with the # key and special codes beginning with a dollar sign ($). Values following the # key are converted to a single character, normally a special control character. For example, #3 is a control C, #13 is a carriage return. These decimal values represent the numeric value of the ASCII code for these control codes. If you do not know what ASCII values are, you probably should not be using this command.

The special dollar parameters can be specified in an echo line by using the dollar ($) sign followed by a key letter. These special codes will be expanded to provide information within the text of the line. The special codes are:

$L Name of the current log file.
$D Current Date.
$T Current time of day.
$@ Last update time for the current RTU
$N Name of the RTU.
$I ID of the RTU
$$ A dollar sign.
$U User's identification from password system.
$0-9 Command file command line parameters.
$%0-9 Task string variables.

Unless the special quote code (') is used at the end of the line, the ECHO command will send a carriage return and line feed character at the end of the text message. If the special character is used, the line is sent without the CR/LF.

Example:
```
ECHO #7 TASK 1 Setup Complete
ECHO Alarm at $T on $D
ECHO Current log file is $L
```

Note: The #7 is the Ascii value for the BELL, or terminal beeper.

Related: HAYES, PRC, FORCE

INPUT Enter Channel value or string data from console.(1)

This command is used to prompt the operator allow entry of a new value. Channel values as well as string variables may be entered with this command. INPUT uses the same line editor as the PROG command described below.

The command requires a prompt, even if it is blank, followed by a channel or variable identifier. The command will time out in about 60 seconds, or longer it an additional parameter is specified after the channel identifier.

The new value takes place immediately after completion of the command.

```
Example: INPUT, Enter New Plate Size,V1     ; use default timeout
         INPUT, Enter New Flow Rate,V5,300 ; 300 second timeout
         INPUT, Enter Report Title, %1     ; string input
```

Related: Pause

PAUSE Display message and pause for confirmation (0).

Command file execution can be controlled with the PAUSE command. The command by itself will cause a default message to be displayed for the user's default command timeout period. If no response is received in the allotted time, then the PAUSE command will assume a negative unless an optional default is provided on the PAUSE command line. A negative response will terminate processing of the current RTU command file. If a positive response is received in time, then the file execution will continue with the next line.

An optional message, timeout period, and default response can be entered as options with the PAUSE command. Note that commas should be used as delimiters in the pause command so that the prompt section can have embedded spaces. These options must be in order on the line as follows:

1. Prompt message
2. Default response (yes or no)
3. Timeout in seconds

```
Examples:  PAUSE
           PAUSE, DO YOU WANT TO CONTINUE
           PAUSE, WANT TO CONTINUE, n,10      ; only 10 seconds
           PAUSE, OK TO GO ON, YES
```

REPORT Generate RTU data Reports (2).

The RTU data can be printed to the line printer or to a disk file for later printing. REPORT by itself goes to the printer (via the spooler if installed). REPORT followed by a disk file name will cause the report to be sent to that file. An existing file with the same name is overwritten.

Report entered from the keyboard or command file will cause the current task to print the report. Hitting the F4 print key will send a REPORT command to the utility task, which will print it as soon as possible. This prevents halting the current task while the report is being printed.

Off-line printers can hang the system, so be sure that the printer is always turned on and powered up.

Example: REPORT
REPORT Data.txt

Related: SET REPORT
SET PRINT

SAY Display a message or Channel value on CRT (1).

SAY is used in command files to display text messages, evaluator results, or channel values on the CRT. It is normally used after positioning the cursor with the CURSOR command. An expression may be used by enclosing the expression with parentheses. To display a text message, place a leading double quote to start the message. Any text not beginning with the double quote or "(" is assumed to be a channel reference (number or tag) and will be parsed as such.

When text is displayed, it is expanded as is done in the ECHO command. This allows for the special $ values to be displayed on the CRT from within a command file.

Special formatting codes can be used to indicate how a value will be displayed. These codes are placed after the value to be affected and are separated from the value by the accent character (‘). The format code is a list of letters and numbers that explain how the value should be displayed. The valid codes are:

Rxx Right Justify in a field of xx characters.
Lxx Left Justify as above.
Cxx Center in a field of xx characters.
Fxx Format real numbers to no more than xx characters.
Dx Decimal Places in floating point numbers.

A If an alarm channel, use CRT attribute in display.

For example, the code R09D2 will produce a number right justified in a field that is 9 characters wide. The number will have 2 decimal places in it.

The code R10F8d3 will produce a right justified number in a field of 10 spaces. The number will be no larger than 8 characters, and the max decimal places will be 3. There may be less places if the number will not fit otherwise.

The default format for real numbers (i.e. channel data) is the same as for screen displays as modified with the SET DISPLAY command. The default for string data is left justified.

Example:
```
SAY "The current time is $T'C40
SAY  A3'R10D2
say ( sqrt(v1) + 1)
```

Related: CURSOR

SCAN Request a range of data values from the RTU (2).

When getting data values for use in a computer, it is desirable to get them in a condensed format if possible. The scan command specifies a range of channels to be sent via a DATA command that the receiving RTU will provide.

Besides the required channel range, a keyword must be used to request options in the format or content of the data. These options allow receiving of raw (i.e. direct I/O data) rather than engineering unit data. In this case, the DATA command will represent raw input values rather than processed engineering values. Analog channels can also request the maximum and minimum values rather than the current values. Status channels can receive the off-on status in a packed format, with 8 channels per entry, rather than a separate entry for each channel. Totalizer and counter channels can request time and date information along with (or in place of) the actual channel value.

The options keyword is made up of a number of letters, each representing a specific option. Not all options apply to all channels. After processing the SCAN command, the DATA command will send back the keyword in the manner that it understood it, so any conflicting or invalid options will be eliminated in the DATA response.

The options in the keyword are as follows:

CODE	FUNCTION	MEANING
R	Raw	Send raw data (i.e. input from field) rather than engineering units values.
L	Latched	Send Latched rather than current Status point data.
E	Engr. Units	Send converted value, not raw value.
T	Time	Send time and date of totalizers and counters.
X	Max	Send maximums for analogs
M	Min	Send minimums for analogs.
@	Time Stamp	Cause receiver to update time to present. Also resets link for this RTU.

For example, a Counter channel may specify **T** for time and date data rather than value data. An analog may specify **EX** to get engineering value of maximums.

CHANNEL TYPE	AVAILABLE CODES
Status	R E L
Analog In	R E X M
Counter	R E T
Frequency	none
Timer	none
Aga3	none
Totalizer	T E

For example, issuing a SCAN S1:S32 R will cause the system to respond with DATA RTU.S1:S32 R a b c d where a b c d will be decimal equivalents of 8 bit values. These values represent 8 points of status input and output. Bits 1-8 will be in the first number, bits 9-16 in the second, and so on. Using the RAW mode allows for faster data transfers than engineering mode.

Note that the first channel identifier in the DATA response includes an RTU name in "dot" notation. This way, the receiver knows the RTU from which the data came so it can place it properly in it's own data table. The channel numbers are relative to the sending RTU, so the receiver has to know which RTU is sending the data. The dot notation is sort of prefix that is allowed in may instances where a channel name is required.

The program takes the values presented in a DATA line and places the values in the system data table as the current point values. Of course, there can be no local I/O scanning taking place or the values will be quickly overwritten by local I/O points. Therefore, a remote RTU uses the DATA command to send point status to a host computer.

If the time option is specified for a counter or totalizer channel, the time and date will be sent as single entries separated by a space. Time always comes first in the pair.

```
SCAN Q1:Q2 T
DATA, RTU.Q1:Q2,T,11:22:33 01/02/88,10:20:30 01/02/88

SCAN RTU.C1:C2 T
```

Data,RTU.C1:C2,11:22:33,10:20:00

Note that commas are always used as entry separators in the data response so that spaces can be used in the time and date field. The format of how data is transferred is up to the user.

MULTIPLE SCANS ON SAME LINE

Software after 02/08/90 allows for multiple transfers to be done in the same command. This speeds half-duplex radio systems that require a lot of overhead when sending each response. To set up multiple requests, simply enter the additional ones on the line after the first one. The SCAN command is not repeated, but the channel range and keyword must be present for each group. Also, note that all points must belong to the same RTU because only the first entry in the response will have the RTU ID.

For example, the following could be used on a small RTU to transfer all points in one command:

```
SCAN s1:s8 R o1:o8 R a1:a6 E m1:m1 E
```

Note that this SCAN command request data for 4 types of channels in a single line. The response might look like this:

```
DATA RTU1.s1:s8,R,12,/O1:O8,R,2,/A1:A6,E,12.3,4.33,2456.00,0,
12.45,22.67,/M1:M1,E,17.65
```

Multiple scans must be set up so that the resulting line will not exceed 200 characters. This is the max line size allowed in all transmissions. In noisy radio environments, shorted lines may produce better overall results due to the difficulty in sending long transmissions.

Each group begins with a slash (/) character preceding the channel range code. The DATA statement is programmed to look for this character and separates each group when it matches this code.

Example:
```
SCAN s1:s8 R
SCAN s1:s8 R a1:a6 E  q1:q4 E@
```

Refer to the DATA command for additional information on the responses to the SCAN command.

Related: DATA, DOWNLOAD

SET NEW SET PARAMETERS

PORT x
Sets the communications port for the current task to the specified number. Use this when linking a task to a port that has a number different from the task itself, as when using COM4 on task number 1.

DATA format
Set the format for floating point numbers used in DATA statement transfers. Use this to cut down the number of characters transmitted if many decimals are not needed. The format is similar to the SAY or CURSOR commands.

DISPLAY format
Set the format to be used for floating point numbers in all displays and reports for this task. Format is similar to that in the CURSOR or SAY command. Use to set decimal places in all displays.

STATus DISPLAY LINK, COMM PORT, AND TASK STATUS (1).

This is similar to the display command for channel data, except that comm port, link, and task status are display instead. The options as follows:

C - Communications Channel Parameters
L - Communications Link Parameters
T - Task variables and states

No options will produce a display of C and L combined.

Example: STATUS
STATUS T
STATUS L
STATUS C

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

DOCUMENTATION REVISIONS

03/13/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future.

2.0 SIGNIFICANT CHANGES

2.1 SIMPLIFIED CONFIG FILES

The most important change in this software release is the elimination of the channel alignments in the RTU.DAT configuration file. Previous to this release, each channel type had to be specified with a count and a starting position in the overall system channel list. The total channel counts also had to be specified during the definition of RTU 0. Both of these requirements have been eliminated.

All that is necessary now is to list the channel types and the channel count per rtu. The overall count and the starting positions are no longer needed. The software tolerates their presence but ignores any numbers other than the count per rtu. Please modify any existing DAT files to eliminate these additional numbers.

For example, a line like:

```
STATUS  8 17  ; 8 rtu points starting at system point 17
```

can now be entered as:

```
STATUS  8  ; 8 status points for this RTU
```

2.2 REPORT LISTING PER POINT

Each channel point now has a control setting to determine if this point should be skipped during reports. This allows for spare and other non-essential points to be skipped during report generation. This setting is part of each channel's configuration screen, and can also be modified with the CHANGE command using the new SR+ option (stands for Skip Report). This setting does not affect the presence of the point in CRT displays.

2.3 REPORT COMMAND RTU SPECIFICATION

Reports can now be specified for an RTU other than the current one. A new keyword "FOR" can be used to specify which rtu will be used in the report. For example:

REPORT for VR256E

will cause VR256E to be used rather than the current RTU.

An additional "TO" keyword can also be used to specify the output destination for the report. The default is to the printer, but any legal DOS file or device can be used as well. FOr example:

REPORT for WC329 to report.txt

will cause a report for WC329 to be sent to a disk file named REPORT.TXT.

2.4 ALARM MODE SETTING DISPLAY

The CHANGE command has been enhanced to allow a short display of the mode settings for a channel range. The new option, "DUMP", can be specified anywhere on the command line where a change option is allowed. The dump will represent the state of the range at the time the DUMP parameter is encountered. Thus, any changes caused by options specified after the DUMP keyword will not be reflected in the display.

For example, entering CHANGE V7:V15 CA- CR- DUMP

will cause channels V7 thru V15 to be changed to NOT call on alarms or resets. After the changes are made, a display will be generated showing the complete alarm mode settings for this channel range.

2.5 REPORT TO PRINTER PROCESSOR LOAD ADJUSTMENT

The program uses DOS to access physical printers as well as disk devices for all I/O activities. When printing to the printer, the program does not get reliable "printer ready"

information. Therefore, attempts to print when a printer is not available will cause the system to run slowly or hang completely. Even during normal printer operation, the size of the printer's internal buffer will cause erratic operation of the program as it waits for the printer to accept another line of output. TEST is currently seeking complete solutions to this problem, and have implemented the first step in this software release.

A new SET parameter, "PWAIT", allows for the user to specify the number of system ticks the program will delay a printing task after each line of output is sent to a physical printer. The goal is to match the printer's ability to output text with the system's rate of printing. Ideally, a printer will completely finish printing the previous line before the next one is sent out. The current version does not have the ability to detect printer ready status during printing, and always sends one line after another to the printer. Using this parameter will allow for delays between each line that are tuned to match the printer and the specific computer setup.

Each system tick is 1/36th of a second. The default PWAIT is 32, so the system will delay a task about one second between lines of output. During this second, other tasks will run uninterrupted. Setting a higher number will accommodate slower printers, while setting a lower number will shorten the delay for faster printers or printers having large buffer capacities. For example:

SET PWAIT 12

Tells the system to only delay the task 12 ticks between each line of output. As with other SET parameters, specifying just the keyword without a value will cause the current value to be displayed.

2.6 EXTERNAL CHANNEL REFERENCES

The term "external" refers to points that are outside the scope of an RTU's point range. For example, if a meter in RTU 3 needs to use channel information from RTU 1, then this is an external reference. Previous versions handled external references in different ways depending on the particular circumstance. This version provides a simplified, standard way to specify channels not belonging to the current RTU.

The rtu name "dot" notation has been extended to include all channel references in all locations. Particularly, channel references made during configurations can now have an optional RTU name as a prefix, using the standard dot notation. The absence of an rtu name implies that the channel reference is within the current rtu, as is the usual case.

During any type of display, any channel references belonging to other than the current RTU will have the RTU name as a prefix

to the channel ID. Using the meter example above, the external channel references would appear as RTU1.A10 rather than as just A10. Whenever the prefix is present, it indicates that the system is using an external channel reference.

During configuration saves, this external RTU indicator is written as part of all channels not local to the current RTU. When these channels are read back in, the external RTU prefix is noted and used as long as that RTU name is known to the system. An invalid name will cause the channel to be assigned to 90, which is the default null channel.

2.7 CMD FILE EXECUTION ON ALARM

Previous versions executed command files on abnormal conditions (if the mode setting for this option was set). This versions will also execute the command file when the point returns to normal if a SET CFILE ON command is executed. The program now passes a single command line parameter of 1 indicating that the point has gone abnormal, and a value of 0 indicating return to normal.

For example, a point tagged HORN will cause the file HORN.RTU to be executed by the utility task whenever its state changes. The actual line sent to the utility task will look like READ HORN 1 when the point goes abnormal, and will look like READ HORN 0 when the point returns to normal. This may affect some systems, so check logic to determine if any editing is needed.

If the SET CFILE ON command is not executed, the program will only execute command files on abnormal and will skip file processing on reset. This command can be entered anywhere at any time, and affects all channels globally. All future command files should contain simple logic statements to determine if the file is being processed as a result of a new abnormal or a clearing one. This is easily done as shown in the sample command file below:

```
if $1 = 1
msg This is a new abnormal condition.
; place commands for abnormal condition here
else
msg This is an abnormal condition clearing back to normal
; place command related to return to normal here
endif
```

2.8 SKIP CHANNEL ON REPORTS OPTION

Each channel now has an additional alarm mode setting that controls the presence of the channel on printed reports. In the past, channels that were disabled did not appear on reports. Because the disable function is intended for alarm control purposes only, this new mode setting has been added to allow individual channel inclusion on reports.

A Y/N option has been added to the config screen that controls this function. A new CHANGE command parameter, "SR" for skip reports, has also been added to allow control over this setting.

3.0 MINOR PROGRAM AND SYSTEM CHANGES

3.1 STATUS CHANNEL HOLD

Status channels could not be placed on HOLD in some versions of the program. This is fixed in this release.

3.2 REPORT HEADER TO WIDE

Locations having long RTU names and titles would cause the top line of the report page to be to long. This has been fixed with a shortened version of the first line.

3.3 MULTI-SCAN RTU SPECIFICATION

The multi-scan option from the previous release would use the current rtu rather than the one specified in the first SCAN channel range for the additional ranges on the line. This was fixed in the field by placing specific SELECT RTUx lines in the download operation. These lines are no longer needed as long as the proper RTU is specified on the first parameter.

3.4 MEMORY ALLOCATION

The pervious release used excessive memory for some operations resulting in a total crash in large systems. This version uses somewhat less memory. The EXE file size has also been reduced by about 12K due to internal program improvements.

3.5 AGA3 METER LATCHING

Certain configurations of AGA3 meters would result in a latching of the AGA3 error code. This caused all subsequent downloads to show the error number rather than the actual meter rate because the error condition had not been cleared. This is now fixed so that a correct AGA3 calculation always clears the error code to 0.

.cp 5
3.6 FILE TYPE ASSUMPTIONS

Using the /F=xxxxx command line option to specify the file to be processed now assumes a filetype of .DAT. Likewise, menu loads now assume a filetype of .MNU unless another one is specified.

3.7 NEW EXPRESSIONS FUNCTIONS

Due to popular demand, new functions have been added to allow testing of a task's on-line status, as well as the status of the task's comm channel. The new functions are:

@ONLINE(x)	Returns 1 if task x is online, else 0
@COMPORT(x)	Returns comm channel number for task x
@COMSTAT(x)	Returns 1 if port x is online, else 0

Using these functions, it is possible to determine communications status during file processing and control program flow based on the existing condition. Previously, only the WAIT command together with the SET ONLINE command were used to control program flow. Now, the @ONLINE function can be used after the WAIT with program flow depending on the on-line state.

3.8 PROGRAM FLOW AFTER ERRORS

Previous releases would always stop file processing after any type of expression evaluator error. This version introduces control over this with the SET ERROR command. When set to OFF (the default), file processing will continue after an error. When set to ON, file processing will stop with the line causing the error as was the case in the older versions.

3.8 DISPLAY SCREEN PROMPTS

The help line at the bottom of the standard display screen now includes + and - characters to show when it is possible to uses these keys and move among RTUs in multi-rtu installations. The location of the various channel options are now fixed along the bottom of the screen, with only the appropriate ones being shown for each RTU. Invalid channel types are blanked out indicating that they are not available for this RTU.

3.9 EXTRA KEYBOARD PRESSES

The function keys used to cause automatic operations like poll the rtu are often held down too long resulting in multiple function requests. To reduce this problem, all function keys presses are reduced to one keystroke by having the program wait till all keys are released before proceeding. Therefore, the 50 or so keystrokes built up by holding the F5 key for 10 seconds

will be read as a single keystroke.

4.0 USING THE UPGRADES

Although most of the newer features are designed to allow existing systems to use the upgrade without modification, some editing may be necessary in order to benefit from the improvements. For example, the multi-channel SCAN command from the previous release is not automatic unless the DOWNLOAD files are reprogrammed. Using the new features will speed downloads and improve overall program performance.

A check list for upgrading is:

1. Multi channel scans to speed downloads. In half-duplex radio systems, full downloads can be done in about half the time by combining channel types in one packet. Phone systems benefit also, although not as much.

2. Elimination of BLOCK SELECT xxxx commands in multi-rtu systems. This was required to insure that the HOST properly digested the SCAN commands in multi-scan operations. The system now uses the RTU specified in the first range on the line for all channels on the line. There is no need for specific select commands when moving from one rtu to the next. Simply specify the desired rtu on the first channels as follows:
   SCAN vr256e.s1:s8 R o1:o8 R a1:a16 E
   In this example, all channels would be assumed to be for VR256E regardless of the current rtu at either the RTU or the HOST system.

3. Elimination of separate BLOCK LINK RESET commands. This is now done automatically when the @ operator is encountered in a SCAN or DATA command. Thus, the last scan line that normally contains a @ to force a time stamp will also clear the current link at either the sender or receiver.

4. Check Menu files for COLUMN then ROW coordinates. Some previous versions used ROW then COLUMN in error. The correct sequence is COL then ROW to be compatible with the CURSOR command.

5. Clean up DAT file to eliminate extra channel data. You no longer need the separate total channel allocations at the start of the file, nor do you need the alignment numbers after each rtu's channel count.

6. Look for strange things in the various RTU files that were put in as quick fixes to older problems. Examples include specific AGA3 commands in LINK files to force AGA3 calculations, LINK Reset commands, Task start commands, and other things that look out of place. These should not be

needed in this version of the program.

7. Eliminate Calculator tasks in all but the most unusual circumstances. This was needed to process AGA3 channels at one time, but are no longer needed. AGA meters are processed in real time just like any other channel. So, take out the CALC task definition from the DAT file and remove any references to it in the start file. If the calc task is present for any other reason, leave it in but remove the AGA portions if any.

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

D O C U M E N T A T I O N R E V I S I O N S

03/30/90

and

03/31/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. This follows the revision notice dated 03/13/90. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future, or keep them as a record of improvements in the system.

2.0 SIGNIFICANT CHANGES

2.1 SETPOINT SCAN LINES

This release allows for low level channel information to be scanned just as the channel's value can be scanned. The normal application will be to have a HOST send setpoint information to an RTU using SCAN's from the HOST to the RTU. This is similar to the method used to change VALUE channels in many AGA3 meter applications.

To implement this new feature, the SCAN command options have been extended to include the following key letters:

L - Low Setpoint
H - High Setpoint
M - Alarm MODE.

The high and low setpoints for value type channels (all except STATUS and OUTPUT) are sent as real numbers using the current DATA format for the task processing the command. The low value for digital channels is sent as either a 0 or 1 to indicate the normal state of the channel. The high setpoint for digital channels is sent as a 0 or 1 to indicate the alarm condition for each channel. It is unnecessary to send both High and Low for digital channels. Either one will do. Note that the setpoints cannot be sent as RAW as can be done for a normal data scan.

These changes required a major revision to the internal operation of both the SCAN and the DATA command. During the conversion, several older (but un-used) options were eliminated in favor of cleaner code. Users should be aware of these changes to avoid possible problems when trying to use these old modes of operation.

The capability of having multiple scans on the same command line (introduced several revisions ago) eliminated the need for multiple code letters in the scan keyword. Now, only one key letter can be specified at any time. The previous capability to combine E and T, for example, is no longer supported. Each scan parameter muse be used by itself. The only exception is the @ operator used to signal the end of a scan sequence. This is still allowed to be used in combination with other letters.

As an example, consider the following command file that would be run at a HOST to send low and high setpoints to a remote:

```
sele WC329
dial
wait 30 conn
set online on
msg RTU $R is online.  Sending Setpoint data
scan a1:a16 L
scan a1:a16 h
scan s1:s8 L     ; only one line needed for status
scan v1:v8 E     ; send Value data while we are at it.
msg Setpoints sent.  Telling RTU to save data.
block save
msg Operation Complete.  Terminating call.
bye
```

Note the use of the H and L codes to specify the low and high setpoints. Note also that a scan line is included for Value channel DATA, not SETPOINT. This would be typical to allow setting of plate sizes and gas quality data from a HOST to the RTU. After sending the data, a SAVE command is blocked over to the RTU to have it save the new values.

When using the new M option, the computer will send its current alarm mode settings on the resulting DATA line. Normally, HOST and RTU systems have slightly different mode settings. The differences are usually related to Call on Alarm, Call on Reset, and Sound Horn setting. In order to change the RTU, the user will re-program the HOST to have the desired settings for the RTU. After sending the mode to the RTU, simply re-load the HOST's setup file for the affected RTU. This will restore the original HOST settings. This is done by using the READ command followed by the name of the RTU.

2.2 EXPRESSIONS IN ECHO PROCESSING

A new $ operator has been added to allow the use of expressions wherever $ operators are allowed. The new syntax is to place the desired expression, in parentheses, following the $ sign. The opening paren is actually the key letter, but the $ processor will pull out the complete expression through the closing paren. After parsing out the expression, it is evaluated and the result placed back into the line.

Optional formatting characters can follow the expression just as can be done with the SAY and CURSOR command. For example:

ECHO The sum of channels V1 and V2 is $(v1 + v2)`D4 Gallons.

This line, after evaluation, would look something like:

The sum of channels V1 and V2 is 413.9263 Gallons.

2.3 NEW FILE SYSTEM

As another internal program improvement, the text file processing system has been revised to use a central file processor. Previously, each task processed files using it's own DOS file handles. This worked pretty well, but was bulky and caused problems with DOS multi-tasking when there was lots of file activity going on. The new system uses a central pool of file handles, and each task requests and releases these handles as it needs them. The central processor accounts for these files and keeps track of what is happening with DOS file accesses.

To support this new feature, a new option to the STATUS command has been added to display the current state of the file processor. Enter STATUS FILES (or just STAT F) and the system will show the current mode for all file handles. The information shows the task using each handle, the file name, the current line number, and a number indicating internal file status.

2.4 SELECT IN COMMAND FILES

Two changes have been made for the SELECT command when used within command files. The first change involves SELECT followed by an rtu name or number. In the past, an incorrect RTU identifier resulted in the current RTU remaining unchanged. The command file would continue processing with what ever RTU was current when the command was processed. Now, command file processing always terminates when an incorrect SELECT is processed. This will avoid processing commands for the wrong RTU if a typo occurred in the command file.

The second change allows the use of the SELECT command without parameters within a command file. If SELECT is entered

all alone on the line, the system will pop-up a RTU select window and wait for the user to pick an RTU. This is similar to the action of the [F10] key. Of course, this will only work for the user on TASK 0, which uses the local CRT display console.

2.5 NEW LINK PARAMETER

The communications LINK system has been extended to include an additional parameter that determines a system's position as either a HOST or an RTU during link processing. Normally, there is no difference between an RTU and a HOST for any system operation (other than real I/O). The term HOST and RTU refers only to the logical use of the system, not its physical make-up.

The new parameter will allow the system to control its reaction during a callout without the use of a LINK file. Although the LINK file is still supported, it is no longer required. Most link files simply contain a line containing READ DOWNLOAD or BLOCK READ DOWNLOAD. Now, the system will generate this message automatically depending on the setting associated with the current link. This only affects callouts on a link. Incoming calls still depend on the caller (who is processing a callout link) to control the communications.

The older LINK file method will still work as before. The change simply eliminates the need for the LINK files in most systems because the action previously done with the LINK file will now be done automatically. On callout, the sequence is now as follows:

1 - Dial the number associated with the link.
2 - Wait the giveup seconds for a connect.
3 - After connect, look for a LINK file.
4 - If file existed, process it.
5 - If no file, then:
    If an RTU process READ DOWNLOAD
    If a HOST process BLOCK READ DOWNLOAD

This new parameter is also being used to reduce collisions between a HOST and an RTU that call one another at the same time. Previously, a dead-lock occured where both systems were processing LINK files. Typically, the HOST was processing BLOCK READ DOWNLOAD while the RTU was already processing the first line of the DOWNLOAD. This resulted in both systems waiting for an ACK to a command, and the systems would timeout and terminate communications.

Now, when a system is waiting for a numbered acknowledgment, and an ack number 0 is received, a system operating as an RTU will terminate file processing but remain on-line. This should result in the other system, which is probably acting as a HOST, to re-send the line. This will allow a priority relationship where the HOST has control of the session. The HOST will continue trying to send the command line (after the ACK wait

timeout), while the RTU will suspend file processing and wait to receive commands rather than send them.

This will only work properly if both systems have been correctly set up as either HOST or RTU. If two RTUs call each other, then both will suspend operation and a timeout will occur. If both are set up as HOSTS, then each will continue trying to get a command through and a similar timeout will occur.

It is anticipated that the new parameter will be used in future program revisions to provide more automatic default control related to the differences between HOST and RTU systems.

2.6 COUNTER CHANNEL ADDITION

A new parameter has been added to counter channel setups to assist in processing noisy, slowly pulsing channels. This new parameter, if non zero, sets the system to de-bounce counter inputs and add a delay between allowable inputs. This effectivly reduces all counter inputs received in one burst to a single pulse, and locks out any further pulses until the specified number of system ticks has passed.

For example, a system with a parameter of 36 that receives 17 pulses when a relay closure occurs will count all 36 as a single pulse and will wait approximately 1 second (36 ticks) before allowing another pulse.

3.0 MINOR CHANGES

3.1 WAIT COMMAND DELAYS

The timing of the WAIT command has been improved and should match entry provided on the command line. Previous releases used a less accurate timing method that was dependent on the amount of processing being done by the system.

3.2 AGA3 ERROR DETECTION

Several improvements have been made in AGA3 meter calculation error detection. A temporary internal variable is now used to track the error during each calculation attempt. The error result from the last attempt is held as the externally available error code throughout the next attempt. This avoids the possibility of downloading an incorrect (or missing) error code if the SCAN command happens to catch the calculation in progress at a point where the code was set to 0.

Error detection has also been improved. A small text file called AGA3.HLP contains a list of the possible error codes. You can view it online using EDIT or TYPE or by entering HELP AGA3.

3.3 INCORRECT TASK NAME DETECTION

The task name parser has been improved to better detect improper task names. Previously, an incorrect name may have been partially decoded resulting in a random task receiving a message. The improvement will return an error code indicating that an incorrect task name was specified.

For example, the command FORCE utiK (not FORCE utiL) may have returned the number 4, indicating the position in the phrase UTIK where an error occurred. Now, an error code of -1 is always returned if a task number is improperly parsed. Note that these codes are only used internally to the program and are not accessible by the user.

3.4 HANGUP ON PORTS DIFFERENT FROM TASK NUMBER

Most systems have com ports that are identical to the task numbers. The few systems with mix-matched ports and tasks had problems using hangup and Bye directly because the termination procedure would be directed to the incorrect port. For example, a com task number 1 using com 4 would have its hangup procedures done on com 1, not com 4. This has been corrected so that all comm procedures use the port number rather than the task number.

3.5 DATA @ENDS FIX

Systems doing a host to host download would not properly process a SCAN @ENDS statement for an RTU other than the current one. This required a specific SELE RTUNAME to be done at the sending unit so that it would send the time stamp for the proper RTU. The rtu name, if specified after the @ENDS, was not being processed by the sender. This has been fixed in the 03/31/90 release (but not the 03/30/90 one). This only affects host systems downloading to other host systems.

4.0 USING THE UPGRADE

There are no specific requirements when installing this upgrade. However, to take advantage of the improvements, the following should be considered:

1. Re-program all links using the PROG LINK x command to inform the system of the HOST and RTU status of the system when using this link. Don't forget to SAVE so the changes will be stored.

2. Eliminate all LINKx.RTU files that are no longer needed due to step 1 above.

3. Check all command files to insure that all task name references are correct. It is possible that incorrect names were previously detected correctly by accident. The new version will no longer tolerate incorrect task names.

4. Systems having port numbers that are different from task numbers can eliminate specific hangup codes in their BYE files. Specifically, this relates to systems using COM 3 and 4 on tasks numbered 1 or 2. This will avoid a double hangup and will save processing time on communication terminations.

5. Systems downloading from one host to another do not need to select each rtu being downloaded. As long as the rtu prefix is used in the SCAN lines to identify the desired RTU. a separate SELECT is not needed at either the sending or receiving end.

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

DOCUMENTATION REVISIONS

04/06/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. This follows the revision notice dated 03/31/90. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future, or keep them as a record of improvements in the system.

2.0 SIGNIFICANT CHANGES

2.1 FIRST OUT ALARM TRACKING

By popular request, a system for tracking first out alarms has been added to each logical RTU. A first out alarm will be the first alarm to go into alarm condition as long as another first out is not pending. Once an alarm is made the first out, no other alarm can become the first out until the current one is ack'd. Subsequent alarms will still show as NEW alarms, but they will not be the first out.

The log system is used to track the first out alarm (lets call it the "FO" alarm). Log table entry number 0 is used to hold the first out entry, and it can be examined at any time by entering LOG FIRST.

The current FO alarm can be scanned by using a special format of the SCAN command. This will cause the system to generate a special DATA line that contains the FO channel number, FO time, FO date, and FO value. This DATA line will be processed by placing this information into the FO table entry for the receiving RTU.

When scanning the FO channel, always make it the last scan. It must be on a line by itself as well. It must be the last scan because it is possible that previous scan lines caused alarms that quickly became the FO channel. This may not really be the FO because the order of download is not related to the order that the alarms come in. So, let the normal alarm scan think it found

the FO alarm, and let the final SCAN FIRST line override any FO found by the local alarm scanner.

Note that each RTU has a separate log table, and therefore a separate FO entry.

2.1 DISPLAY FIRST

The function of the SHIFT-F7 key has been enhanced and also made into a separate command function. This key will tell the system to locate new alarms that have not been ack'd by the operator. Previously, the system would search the current RTU for new alarms, and display them in an internal channel order. When no new alarms for the current RTU were found, the system would scan other RTUs for pending alarms.

This has been enhanced to cause the system to always display the First Out alarm initially. After the FO alarm is ack'd, the function works as before. Once no new alarms exist, a message will be displayed indicating there are no new alarms.

The function initiated by the SHIFT-F7 key has been moved into an option to the DISPLAY command. Now, the entry DISP FIRST will cause the same action as SHIFT-F7. This allows the function to be placed in command files and menu entries.

3.0 MINOR CHANGES

3.1 WAIT COMMAND DELAYS

The timing of the WAIT command has been improved (again!) and should match entry provided on the command line. Previous releases used a less accurate timing method that was dependent on the amount of processing being done by the system. Also, systems using phone links were not correctly processing the number of seconds specified on the command line.

3.2 OUTPUT NORMAL STATE

The normal state of a physical output was not being correctly processed during the configuration load. This has been corrected so that systems having normally energized outputs can consider these to be logically "off" rather than on. This only affects MESA rtus using normally on outputs.

3.3 SCREEN DUMP FILE CLOSES

A bug crept into the previous release that prevented the proper close of the file used to send the printout time and form feed to a screen dump. This resulted in the program running out of dos file handles after 10 or so dumps, requiring a restart

of the program. This has been fixed.

3.4 RESET FILES OPTION

A new option for the RESET command will reset all file handles in use by the current task. Using this command will allow for recovery from problems that may pop up in the future where files are not properly closed. It should only be used as a temporary fix, but it is there if we need it.

3.5 MENU REPEAT PROBLEM FIX

Under certain circumstances, the menu system would get confused and either refuse to respond to a selection, or would continue to redisplay the menu after a selection. This was a problem caused by the nature of the message passing system in the multi-tasker that runs the program. To eliminate the problem, a compromise was made that always allows the menu selection to be processed. The tradeoff is that any time the menu is displayed and a selection is made, any waiting messages from any source are deleted from the message queue. This insures that the messages waiting to be processed are the ones generated by the most recent menu selection. Older messages left in the queue for any reason, as well as messages sent to task 0 while the menu was displayed, will be lost.

In practice, this is not much of a problem because menus are usually not displayed very long, and most users do not send messages frequently to the local task. However, be aware that programs running on a comm task that send progress messages to the local console will only have those messages displayed while the local operator is not in menu mode.

An additional "key eating" procedure has been added to the start of all menu pop-ups. This causes a slight delay in the pop-up, but avoids the problem of operators pressing and holding keys when using the menu system. Previous attempts to avoid this problem have proven inadequate.

4.0 USING THE UPGRADE

There are no specific requirements when installing this upgrade. However, to take advantage of the improvements. the following should be considered:

1. Change any callout files that now use a temporary SLEEP command to use the WAIT command (that now works correctly).

2. Re-program any normally on output functions if desired. You can now turn an output ON that will actually de-energize it if the physical off state is programmed ON. This is confusing, and only affects MESA.

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

DOCUMENTATION REVISIONS

05/30/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. This follows the revision notice dated 04/06/90. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future, or keep them as a record of improvements in the system.

2.0 SIGNIFICANT CHANGES

2.1 SUBROUTINE PROCESSING

A new command, GOSUB, allows processing of command files in subroutine style program flow operations. It is very similar to the READ command, which is still in place. The READ command causes termination of the current file, and continues processing at the start of the named file. The GOSUB command holds processing of the current file, and begins execution of the named file. When processing of the new file is complete, the program continues execution at the line following the GOSUB command.

GOSUB reduces the need for so many pending FORCE messages by allowing processing to be nested within the original file. This command simplifies the use of standard routines by allowing them to be "called" from within other command files. It also simplifies startup procedures because the older FORCE 0 READ sections can be changed to GOSUBs.

Command line parameters can be passed with the GOSUB command just as they can are done with the READ command. Parameters are only local to a single file. In order for a called program to get the parameters, they must be passed on the command line. Consider the following 2 programs started with the command line READ FILE1 PARAM1 PARAM2 PARAM3:

```
; Command file FILE1
MSG This is the start of program 1 $1 $2 $3
gosub file2 $3 $2 $1
msg Back to the main file
return
```

```
; command file FILE2
msg This is from program 2 $1 $2 $3
return
```

When FILE1 is started, it receives the three parameters that become local to that file. When FILE1 calls FILE2, it passes three text parameters to FILE2 in the reverse order because they are defined in reverse order at the time of the call. Running FILE1 as described produces the following output:

This is the start of program 1 PARAM1 PARAM2 PARAM3
This is from program 2 PARAM3 PARAM2 PARAM1
Back to the main file

Note that the RETURN statement can be used to stop execution of the current command file. Execution also stops when the end of file is reached, so RETURN is only needed if termination is needed within the body of the file. A RETURN from the "topmost" file will return processing to the command line.

3.0 MINOR CHANGES

3.1 RTU SELECT COMMAND

A minor bug in the rtu selection code involving incorrect, out of range rtu numbers has been fixed. If an rtu number greater than 255 was specified, the system would return an incorrect rtu selection, often resulting in improper operation. Now, any invalid RTU name or number will result in a command error.

3.2 RTU SELECT WITH + and - KEYS

While in the display mode (F7 key), the plus and minus keys can be used to move from one RTU to another. This release provides for a wrap around so that the user can move from the last RTU to the first RTU and back again without stepping through all the RTUs on the system. This is similar to the way the arrow keys allow wrapping around in the menu and RTU pick list displays.

Previous versions would stop at the first or last RTU, and would ignore any additional plus or minus keys that would have moved beyond the first or last RTU in the list.

3.3 BAD LINE RECOVERY

The logic that controls recovery from garbled data transmissions has been improved to allow for better recovery. Previous versions would become confused during a long download over poor phone lines. The improved system allows for better recovery by ack'ing old messages as many times as necessary to keep the communications flowing. Previous versions used a time-out system to allow for duplicate or missing messages.

This improvement will be most noticeable on phone based systems using long link files. In these units, the host typically controls the download from the RTU rather than having the RTU process a local download file. Radio based systems, which normally let the RTU control the data flow, will not be affected.

3.4 Serial Port Driver Improvements

The serial port communications section of the program has had several minor improvements intended to allow recovery from abnormal situations. These include transient noise, random break character sequences, and other strange events that only seem to occur offshore. The improvements basically allow for the serial port system to recover internally without being manually reset.

4.0 USING THE UPGRADE

Assuming a system is being upgraded from the most recent release, no changes are necessary to existing systems to take advantage of this particular upgrade. Systems upgrading from older versions should consult all of the release notes issued since their previous version was installed.

The GOSUB command, while not absolutely needed, can be used to simplify start file processing. Other procedures can use the GOSUB command as well. Each installation will have to review its files to determine if this new feature is needed.

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

DOCUMENTATION REVISIONS

07/17/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. This follows the revision notice dated 06/07/90. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future, or keep them as a record of improvements in the system.

2.0 SIGNIFICANT CHANGES

2.1 PROCEDURE LIBRARY

It is now possible to combine several command files into a single file and define each group of commands using the line PROCedure procname. This procedure file can then be loaded into a buffer in memory which will act as a library of procedures. This eliminates the need to use DOS for most command file processing because the text of the file is already in memory. It also simplifies maintenance because there are fewer files per RTU.

When the program processes commands the library will be searched first before looking on disk for command files. This avoids disk accesses for common procedures. The search sequence is to check the library, then the disk for a fill with an .RTU extension. If the READ or GOSUB command references a file with a file type specified, then the library search is skipped.

```
READ START1          ; check lib then disk
READ START1.RTU      ; go directly to the disk
```

This is important because any existing files that specify a file type in the READ command line will have to be edited in order to benefit from the library function. Also, files that are normally on the disk (like setup and Link files) will be accessed faster if their file type is used on the command line that reads them in.

The library command options are:

```
LIBRARY LOAD filename  ;load a file of procs into memory
```

```
LIBRARY  MERGE  filename ;load entire file into memory as
                  a single procedure using the file name
                  (without the extension) for the name of
                  the procedure.
LIBRARY CLEAR   ; release all procs from memory
LIBRARY DUMP    ; display list of all procs in memory
```

If filename is not specified in the LIBRARY LOAD command the default file name RTU.LIB is used. If no extension is given in the filename for the LIBRARY LOAD command the default extension used is '.LIB'. If no extension is given in the filename for the LIBRARY MERGE command the default extension used is '.RTU'.

The size of the procedure buffer and the maximum number of procedure entries allowed can be adjusted by placing a line in the RTU set up (.DAT) file. The line must be of the form:

LIB buffer_size max_entries.

Example:

```
Lib 12000 120        ; set up bigger buff with lots of files
```

The default buffer size is 8000 bytes and the default number of procedure entries allowed is 64. The minimum values allowed for these settings are 2000 bytes of memory and 32 entries.

A small example library file is as follows:

```
procedure start1
;set port 2
set wait 10    ; time to wait for an ACK from RTU
set call delay 120  ; secs between callout trys
set call giveup 30; secs to wait for a connect to RTU
set call trys 4  ; number of callout attempts
set media phone
force local echo Task 1 started at $T $D

proc bye1
SET ECHO OFF
hayes S0=1 Q1 E0

proc daily
sele 1
calc v4 = v5
calc q1:q3 = 0

; end of library
```

When this file is loaded with the LIBRARY LOAD command, it will make the procedures START1, BYE1, and DAILY available quickly and without DOS accesses.

If a new procedure that is the same name as an existing procedure is added to a library via either the LOAD or MERGE

option, then the new one replaces the older one. The old one is still there, it is just de-activated. It will appear in the DUMP of the library.

2.2 VARIABLES

Variable storage space used by the expression evaluator is no longer allocated for each RTU but rather for each task. The default number of variables allowed for each task is 0. To change the number of variables allowed use the VARIABLES line in the RTU set up file. Each variable line is associated with the most previously defined task. To set the number of variables for task 0 use the VARIABLES line before listing any TASK lines.

Variables can be defined for a task at either the LOCAL or PUBLIC level. Public variables can be used by all procedures but local variables can only be accessed by the procedures that declare them. Upon exit of a procedure all local variables defined within that procedure are automatically released but all public variables remain defined. Publicly defined variables can be released by using the RELEASE command.

Examples:

```
LOCAL total1 total2 total3
PUBLIC new_value old_value
RELEASE varname
RELEASE ALL ;delete all publicly defined variables
DUMP VARS   ;display nesting level, name and value of vars
```

Varibles are useful for temporary calculations, operator prompts, or other numeric entries that do not require storage. Global (public) variables are resident and remain between command files, while the locals are gone when the program completes.

2.3 PRC RESET OPTION

The program now allows for a hardware reset of modified PRC units. These units must have pin 20 of the PRC connected internally to the CPU reset. If this modification is in place, then the computer can reset the PRC by toggling pin 20 as is done with Hayes modems.

The reset occurs whenever one of the following occur:

1. A SET MEDIA RADIO command is processed by the task. This always causes the reset sequence to occur.

2. A comm fail occurs because the callout trys has been exceeded. If there is no COMFAILx.RTU file, then the system will do the PRC reset as a minimum.

3. A COMM RESET command is executed on the task connected to the PRC. This can occur at any time for any reason. If a

COMFAILx.RTU file is being used, then it should contain this command because it is not automatically executed if the file is present.

The only modifications necessary to the PRC to do this reset function are as follows:

1. Connect a .02 micro farad (or any small cap) between pin 20 on the DB-25 connector and pin 4 of U6 (a 1489 RS-232 receiver chip). This lets a hi-to-low transition on pin 20 pulse the reset pin on the PRC's computer chip.

2. Make sure that pin 5 or 20 is connected to pin 4 on the DB-25 connector. This holds the RTS line active at all times.

3. Make sure that pin 4 has been removed from the RS-232 cable coming into the PRC. This line is normally used for the watchdog toggle, and it messes up the PRC if connected. The jumper provided in step 2 above will hold pin 4 active, so no other connections to pin 4 are allowed.

3.0 MINOR CHANGES

3.1 LOG COMMAND

The log command now has the option of specifying a range of channels to display from the log file. This command is of the form:

LOG FROM logfile TO outfile FOR range

The FROM, TO and FOR parts of this command are optional and may be listed in any order. To display the entire contents of the log file simply use the command LOG all by itself (used to be LOG LIST).

3.2 LINK CONFIGURATION

The programming of a link can now be done using the command CONFIG LINK. This is an alternative to using the command PROG LINK.

3.3 PASSWORDS

The existing password system has been cleaned up and works better than before. It is necessary to have a PASSWORD OFF line in the control file (.DAT) in order for passwords to be eliminated.

3.4 EDITOR BLOCK READ AND WRITE

The editor has been improved to prompt for the file name to read or write. The previous version always used a temporary file

with a fixed name. The editor will now ask for the name at the top of the screen before executing the function.

This can be used to read in all the existing RTU files into a single LIB file by doing control-K-R to request a block read. Simply specify each existing RTU file to read them into the single LIB file.

4.0 USING THE UPGRADE

Assuming a system is being upgraded from the most recent release, the only required change is to place a PASSWORD OFF statement in the .DAT file if it does not exist already. No other changes are necessary to existing systems to take advantage of this particular upgrade. Systems upgrading from older versions should consult all of the release notes issued since their previous version was installed.

To take advantage of the lib function, create a new file that consists of the RTU name with a file type of .LIB. Then, read in all of the non-changing command files and indicate each one as a separate procedure by placing a PROC fname line ahead of each one. Then, place a LIBRARY LOAD $R line in the start0 file. This will cause the library to be set up when the system starts up.

PRC RESET MODIFICATION

07/19/90

The AEA-PK90 Packet Radio Controller can be modified to allow for a hardware reset via pin 20 DTR. RTUMON software version 07/19/90 and later will use this pin to do the reset at various times if the task is set to use a RADIO (set media radio).

The reset pulse on pin 20 is about 1/2 second, after which the program waits 5 seconds for the PRC to complete its startup sequence. Then, a series of 6 stars (*) are sent to the PRC to synchronize the baud rate. This is not normally needed unless the PRC is totally trashed, but it is always provided just in case.

The PRC reset can be initiated by any of the following:

1. A SET MEDIA RADIO command is processed by the task. This always causes the reset sequence to occur.

2. A comm fail occurs because the callout trys has been exceeded. If there is no COMFAILx.RTU file, then the system will do the PRC reset as a minimum.

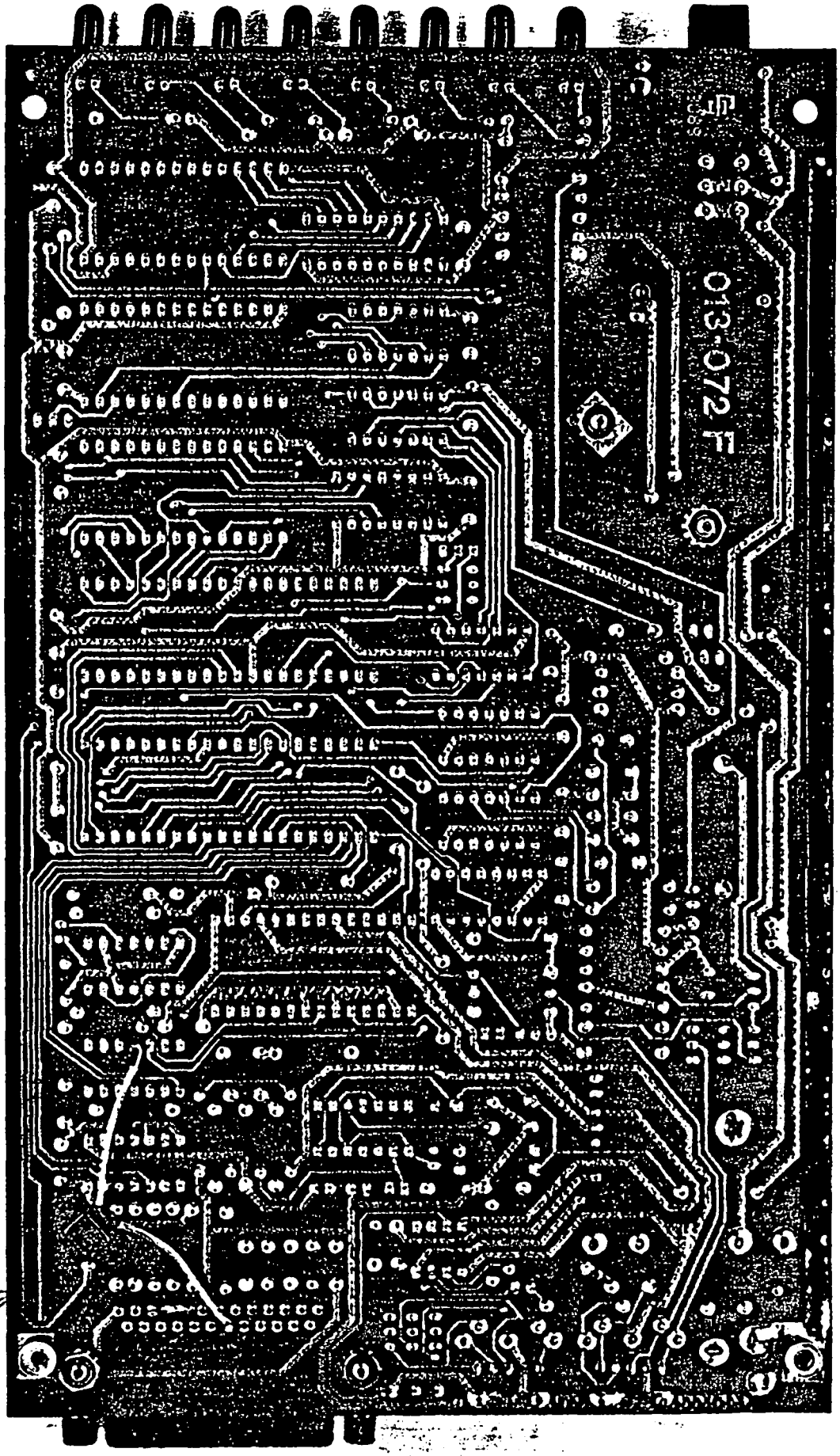
3. A COMM RESET command is executed on the task connected to the PRC. This can occur at any time for any reason. If a COMFAILx.RTU file is being used, then it should contain this command because it is not automatically executed if the file is present.

The only modifications necessary to the PRC to do this reset function are as follows:

1. Connect a .02 micro farad (or any small cap) between pin 20 on the DB-25 connector and pin 4 of U6 (a 1489 RS-232 receiver chip). This lets a hi-to-low transition on pin 20 pulse the reset pin on the PRC's computer chip.

2. Make sure that pin 5 or 20 is connected to pin 4 on the DB-25 connector. This holds the RTS line active at all times.

3. Make sure that pin 4 has been removed from the RS-232 cable coming into the PRC. This line is normally used for the watchdog toggle, and it messes up the PRC if connected. The jumper provided in step 2 above will hold pin 4 active, so no other connections to pin 4 are allowed.



SMALL CAP- PIN 20 to IC6-PIN 4

JUMP 4-5 or 20-4 on DB-25

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

DOCUMENTATION REVISIONS

08/24/90

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. This follows the revision notice dated 07/17/90. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future, or keep them as a record of improvements in the system.

2.0 SIGNIFICANT CHANGES

2.1 MENUS IN PROCEDURE LIBRARY

MENU files can now be contained within the procedure library just like RTU command files. The menus are identified with the PROC keyword followed by the name of the menu. The PROC keyword is used even though these files are not actually procedures. The keyword simply serves to identify and separate each section of the library files.

When using menus in the lib file, it is important to not use a .LIB extension when referencing the file. Just as is done with RTU command files, the file processor will skip the library if the referenced file has a specified file type. So, entering LIB LOAD MYMENU.LIB will force the program to look directly at the current disk, bypassing the library. However, entering LIB LOAD MYMENU will allow the program to first check the library. If not found there, the program will append the .MNU filetype to the name and look on the disk.

2.2 NESTED MENUS

Because menus have become quite complicated, we have added a nesting procedure that allows "calls" to a sub-menu followed by a "return" to the previous menu. Previously, the name of a menu had to be known so it could be loaded. Now, the system will track the names for you and allow you to go back to a prior menu without knowing its name.

The new keywords are:

```
MENU READ [menuname]     ; original transfer to new menu
MENU GOSUB menuname      ; nested transfer to new menu
MENU RETURN              ; go back to previous menu
```

Note that the nesting only works if the GOSUB and RETURN commands are used. The system maintains a list of menu files used, and each GOSUB moves one position down the list. Each RETURN moves up one position on the list. A READ does not change position, but does change current menu files. This is very similar to the method used for nesting RTU command files with the normal READ, GOSUB, and RETURN commands.

2.3 ALARM VALUE DEADBANDS

All value type channels (analog, counter, totalizer, meter, etc), can now have a deadband associated with the alarm value. The deadband provides an increment that separates the into-alarm value from the out-of-alarm value. The deadband affects both the high and low alarms, but in opposite ways.

The high alarm will still operate whenever the channel reaches the high value (or above) and the alarm delay times out. After entering the alarm state, the high alarm will not clear until the channel value drops to the high alarm minus the deadband. The low alarm works in just the opposite way. The low alarm occurs as before, but does not clear until the channel value rises above the sum of the low alarm and the deadband.

This is being provided to reduce nuisance alarms on channel types that hover near the alarm point. Without the deadband, the channel would continually re-alarm in response to small changes in the channel value. Now, the deadband allows for customization of each point to determine how much of a change is required to clear an existing alarm.

The deadband value appears in the channel CONFIG screen, and can be scanned or set with the SCAN and DATA commands just like high and low setpoints. The command line option for referencing the deadband is the letter "D" as follows:

```
SCAN a1:A8 D
DATA A1:A8 D 1 2 3 4 5 6 7 8  ; sample data line
```

2.4 DELAYED FILE EXECUTION REMOVED

A change has been made in the way many automatically activated files are being processed. Previously, file execution associated with the BYE command, LINK connections, comm CONNECTIONS, and a few others were processed by sending a message to the current task to read a specific file (like BYE1.RTU). This worked OK in many instances, but prevented certain sequences

from processing the files in the proper order. This was mostly during processing of command files containing BYE commands.

The new method, which uses the recently added GOSUB capability, causes these files to be read immediately rather than being deferred until a later time. This way, a BYE will cause a disconnect and processing of the BYEx.RTU file (or lib entry) right away. After the BYE file is completed, execution resumes at the interruption point of the previous file.

This process is also now used for SAVE, LINKx.RTU, CONNECTx.RTU, and COMFAILx.RTU file processing. Other auto processing, such as into-alarm and logging procedures, are still processed by sending messages because the tasks that originate these requests do not themselves process files. The timing of the processing for those files is not as critical, and delayed execution should not cause any sequence problems.

This change, when used with the SAVE command, will also prevent the loss of the setup file (rtuname.RTU) if a SAVE is started during communications and the link is lost. Because the actual save was done with messages, the actual storage took place after the SAVE command was executed, not while it executed. However, the output file was already opened for use, and a loss of comm link automatically closes any open output files. Thus, the save took place to a closed file and the file actually created was empty.

3.0 MINOR CHANGES

3.1 TIMER OUTPUT DISPLAY FIX

The current status of an output channel controlled by a timer was not displaying correctly in some configurations. This has been fixed so that all TIMER channel displays will correctly indicate the current status.

3.2 LIB MERGE

The newly added LIB command has had an extra option added that allows for a normal RTU text file to be added to the currently loaded library. The LIB MERGE fname command will cause a new lib entry to be created that has a procedure name that matches the command file name. The text of the file is then added to the current library. After the merge, the program appears as if were originally part of the library.

This is provided to allow simple checking of a new RTU file without having to actually adding it to the lib file. After checkout, it is assumed that the RTU file will be placed in the actual LIB file, so this is only a temporary use command.

3.3 EDITOR DEFAULT NAME

The text editor now remembers the most recently used file specification when started without any specified file. Previously, the filespec defaulted to *.RTU if none was entered after the EDIT command. Now, any filespec containing a start character will become the default filespec for any future edits.

So, entering EDIT *.LIB will cause *.LIB to become the default filespec. However, entering EDIT MYFILE.LIB will not affect the default because it does not contain a wildcard (*) character.

3.4 VARIABLES NOT REQUIRED

A bug in the previous release required that all RTU type tasks be assigned at least one variable with a VARIABLE X statement in the DAT file. Now, all RTU tasks default to 0 variables without any consequence unless a VARIABLE statement is used. So, there is no longer a need to force variables for each task if it does not actually use them.

3.5 MULTIPLE LINK BUG

Certain multiple link configurations confused the alarm callout system so that it would activate all links for an alarm point rather than only the ones assigned to that point's alarm group. This occurred when a system was acting as both an RTU and a HOST where it received alarms and forwarded alarms to different links. This problem has been fixed so that only the links in each point's group are activated regardless of the RTU or HOST nature of the installation.

4.0 USING THE UPGRADE

Nothing is required to use this release, although the advantages will not be found without some re-arrangement of some procedures. Any field fixes used to work around the delay in BYE file processing can probably be eliminated. Also, the menu files can be placed in the location's LIB file if desired.

Assuming a system is being upgraded from the most recent release, the only required change is to place a PASSWORD OFF statement in the .DAT file if it does not exist already. No other changes are necessary to existing systems to take advantage of this particular upgrade. Systems upgrading from older versions should consult all of the release notes issued since their previous version was installed.

To take advantage of the lib function, create a new file that consists of the RTU name with a file type of .LIB. Then,

read in all of the non-changing command files and indicate each one as a separate procedure by placing a PROC fname line ahead of each one. Then, place a LIBRARY LOAD $R line in the start0 file. This will cause the library to be set up when the system starts up.

TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA PROGRAM

D O C U M E N T A T I O N R E V I S I O N S

03/01/91

1.0 INTRODUCTION

This document describes significant changes in the RTU/SCADA system that are not included in otherwise current documentation. This follows the revision notice dated 08/24/90. Only the sections that have new or revised information are included here. Attach these notes to your existing documentation until the entire set is replaced in the future, or keep them as a record of improvements in the system.

This documentation identifies many changes made during the numerous test versions released since the 8/24/90 version of the program. If you are a test site, you may have only some of the changes in your exact version. For example, the test releases of 02/27 and 02/28 do not have all of the items mentioned in this document.

2.0 SIGNIFICANT CHANGES

2.1 INTERNAL REORGANIZATION

Although this is not obvious to the user, the internal operation of the program has been reprogrammed to accommodate the many changes provided by this update. The multi-tasker, the command processor, the file processor, and many individual commands have been completely rewritten to improve performance and enhance their features. Many of the changes were required to prevent program failures in some systems having multiple ports and numerous RTUs. One effect of this reorganization is that the program is about 60% larger than the previous version, which has introduced a whole new set of memory space problems. These problems are solved for the near term, although another internal revision will be required to allow use of larger RTU and point counts in the future. This will probably be done during 1991.

ROM based systems will notice that the RTUMON EXE program and OVR overlay now occupy 3 roms instead of 2 as in the previous release. Some of the older versions were also supplied on 3 roms. The actual rom count depends on the size of the program as well as the squeezing program used to compress the EXE file. As the program design changes, the ability to fit into 2 roms also

changes. At this point, it appears that 3 roms will be required for all future releases.

2.2 AUTOMATIC VALUE SAVES

The program now has an automatic, transparent data save system that continuously saves many dynamic values while the program is running. This service eliminates the many small data save files that were needed to save plate sizes, gas quality values, and production history values. This feature creates an image of the data to be saved in the actual internal format of the program. Normally, the program saves values in text format so they can be interchanged with other programs or edited with a text editor. The image, however, uses the internal format because it requires no data conversion. This allows for the fast performance of the image system, making its use transparent in most applications.

So, the image is stored on the default disk drive in the internal format of the program, and this file is not usable in any other way. It cannot be edited or modified, and it should never have to be used in any other way. The name of the file is the same as the system name and has a file type of ".IMG". Therefore, a system called VR167 will have an image file called VR167.IMG.

The image save system does not automatically start operation when the program is loaded. This is because we want to be able to load a previously saved image before we start saving to the image file. The normal sequence is to have the startup file (START0) load in the normal channel description file, then load in a saved image file, and finally start the image save system. An example segment of a start file would be like this:

```
gosub $S.lin    ; load in the link information
gosub rtu1.rtu  ; load in setups for rtu1
gosub rtu2.rtu  ; load in setups for rtu2
image load      ; retrieve last saved values
image on 10     ; start saving every 10 seconds
```

Note that the "image on" line has a number specifying the number of seconds between saves. It is not practical to save the data constantly because that is all the system would have time to do. A compromise must be made between disk speed, processor speed, and how often data actually changes. For RTU systems with ram based disk drives, every 5 or 10 seconds is fine. For hard disk systems, every 30 seconds is a good starting point. Floppy based systems, which are very slow, may have to use 5 minute intervals or longer.

The image save can be forced to occur at any time by

processing the command "IMAGE SAVE". This can be used after any critical value changes to insure that the image save occurs as soon as possible. The command sets an internal flag that tells the background process (actually the One second task) to do the save as soon as possible.

The list of values and parameters saved in the image are:

Alarm mode
Alarm State
Seconds till alarm

Digital channel status
Output channel state
Latched Status input

Current channel value
Counter and Totalizer start Time and Date
Frequency channel seconds left to sample
Timer channel seconds left
Timer channel active state

There are other internal information saved as well, but it is not of importance to the operator.

2.3 NEW MAIN PROGRAM MENU

The menu system for the local user has been greatly enhanced to provide a typical PC type "pull down" main menu. This menu allows many of the typical SCADA program operations to be done with one finger and the arrow keys. The new menu does not add any additional capability to the program itself. The main menu simply makes it easier to do many of the common functions.

This menu is started in exactly the same way as the old custom menu system: press F9 or enter the MENU command. This causes the new menu to appear, and the user can use arrow keys and the [ENTER] key to make selections. The old custom menu system is available from the main menu under the USER heading, which indicates the user's menu rather than the system's menu.

The User's menu is identical to the old menu system, and is used for any custom functions unique to each location. To directly access a user menu, replace the existing references to MENU (with no other parameters) with the new keyword USER. All of the other MENU functions work as usual, so no program changes will be needed there. Only functions that are intended to put up a user menu need attention. Therefore, the only time the USER keyword must be used is when it is on a line all by itself.

These main menu functions are divided into categories such as DISPLAY, EDIT, CONFIG, and DIAGNOSTICS. Under each of these

headings is a sub-menu that provides specific functions related to the main heading. For example, under DISPLAY is a list of channel types. Selecting the proper one (with only the arrow keys and the [ENTER] key) will cause the program to put up a display much like the F7 key or the DISPLAY command does. This is the initial application of this type of menu system, and it will probably be enhanced based on user's feedback.

A new SET command option can be used to set the program in "Auto-Menu" mode. In this mode, the main program menu will automatically pop-up whenever the user is at a command prompt. This will prevent the untrained user from being left in manual command mode by accident. From the main menu, the operator can select a Menu Exit that will escape to command mode as well as cancel the auto menu function.

The main menu also displays some system status at all times on the top line of the screen. A signal indicating waiting messages indicates that the user should exit menu mode to look at the messages that have come from other tasks or from some other source.

2.4 ENHANCED TEXT EDITOR

The built-in text editor has been greatly improved so that it resembles a small word processor program. In addition to the simple editing tricks that the old editor could do, the new one offers many other features such as find and replace, block moves, block copies, block erase, text printing, word wrap, and many other features. Like the previous editor, the keystrokes emulate those of Wordstar, Dbase, Sidekick, the Turbo Languages, and many other popular PC programs. An on-line help file (called RTUEDIT.HLP) can be displayed by pressing F1 while in the editor. This will cause a small screen to appear with hints on the various editor functions.

A new SET command option is available to determine if the editor creates a backup file every time a file is saved. On systems with limited disk space, the accumulation of backup files causes a storage space problem. Using SET BACKUP OFF will cause the editor to skip the generation of backup files.

2.5 FUNCTION KEY FILES

All of the function keys can have a shifted-key action assigned to them. This is done by having Task 0 run a command file, with a separate file for each key. These files are named for "Shift F1, Shift F2" etc, and of course have the ".RTU" file type. The file names are SF1, SF2, SF3,...SF10 for each of the function keys. When the shifted function key is pressed, the system will look for the proper file, and if it exists, will send

a message to task 0 in the form of "READ SF1". The contents of the file is completely up to the user.

In addition to sending the command, the system also sends an escape and a control-c to task 0. This will force the task out of any screen or menu so that the command can be processed.

TEST has no plans at this time to standardize the use of these keys. The uses of these keys are completely up to the individual operators.

2.6 POLL WITH REPORT OPTION

In previous program versions, there was no facility to have the system poll an RTU and print a report after a successful data transfer. The normal procedure was to start a poll, wait a while, and then print a report. This was fine as long as the communications system was operable, which is not the case at some of the installations. A new POLL option provides for an automatic report at the time the data transfer is time-stamped. Using this option, a report will be printed as soon as a successful transfer is complete, but will not occur if the transfer cannot be done. This eliminates reports that contain old data.

The new option to the POLL command is the keyword REPORT in place of the keyword RTU or NOW. So, agenda files and other processes that contain POLL NOW or POLL RTU can be changed to read POLL REPORT. An optional RTU name can follow the REPORT keyword, and the command defaults to the task's current rtu if none is specified.

2.7 INDIVIDUAL DECIMAL PLACE SETTINGS

The previous versions used a global decimal place setting that determined how all value type channels were displayed. This version introduces a new parameter for each channel that determines how the channel will be displayed and transferred to another unit. This new parameter is the number of decimal places, and is an optional parameter for each channel.

If the parameter is 0 or greater, than the program will format the text version of the channel to the specified number of decimals. If the setting is negative, then the program will use a single, global decimal place count established with the SET PLACES x command. Therefore, most channels can be set easily by setting the system default and leaving each channel at the system default setting. Certain channels, such as orifice plates, will display better with a higher number of decimal places. Other channels, such as AGA-3 flow rates, will display better with 0 or 1 decimal places. These channels can be individually configured

as desired, while the majority of the channels can be left at the default system setting.

Note that setting the decimal places will limit the precision of the value on the current system as well as on any system that receives the value via a transmission. If only 1 decimal place is used at an RTU, then the transfer will only provide 1 decimal place as received by the HOST. Setting a higher number of places at the HOST will not result in any additional precision because the precision has been removed at the RTU prior to transmission.

3.0 MINOR CHANGES

3.1 NEW SCAN OPTIONS

The SCAN command uses various code letters to indicate the format and content of the data to be scanned. These codes go on the SCAN command line just after the channel range specification, and are repeated in the resulting DATA line that the SCAN command generates. Most options are used one at a time, while a few can be used in conjunction with other codes. The current code letters are:

R	Raw data format
E	Engineering Units data format
T	Time and Date format
H	High alarm level (in engineering units only)
L	Low alarm level (in engineering units only)
D	Deadband (in engineering units only)
G	Callout Group (whole numbers only)
M	Minimum value (totalizer channels only)
W	Time delay (wait) till alarm, whole seconds only
@	Time stamp and link reset signal
#	Time stamp clear

The @ and # codes are the ones often used with other codes, where the # is used on the first scan line and the @ is used on the last one. The # will cause the system processing the DATA line to clear out the update time and date. This allows transfers that are terminated abnormally to be indicated by a blank time and date. The @ code at the end of a transfer will cause the sending unit (the one processing the SCAN command) to reset the current link (if the unit had called out). The @ code at the receiving unit (the one processing the DATA command) will update its current time and date for that rtu.

3.2 RTU SELECT BY NUMBER ELIMINATED

Previous versions of the program allowed for RTU selection by name or by number. For example, both SELE RTU1 and SELE 1 were valid selections. This has proven to be a bad feature because inadvertent selection of the incorrect RTU was to easy to do. Command files are often transferred from one system to another, with minor changes made to suit the new location. The selection by number option was often overlooked and would provide unreliable results if an incorrect number was present. Selection by name always works, and an incorrect rtu name causes file processing to stop.

Starting with this program version, all selects must be by name. This will prevent the above mentioned problems, as well as problems caused when the order of RTU's in the main system setup is changed. So, SELE RTU1 will still work but SELE 1 will not. **CHECK ALL COMMAND FILES FOR SELECT LINES AND CHANGE THEM TO SUIT THE NEW METHOD.**

Note that selection of the system level RTU, RTU 0, is still done by number. This is the only RTU that can be selected by number, and this is the only way to do it.

3.3 GOTO ERRORS STOP FILES

If a label is not found when a GOTO command is processed, the program now stops processing of the file completely. Previous versions did unpredictable things when the GOTO target could not be found.

3.4 SAVE COMMAND CHANGES

The SAVE command used to save both the current RTU data as well as the system's link information. The link system now has its own save option, so using a general SAVE command no longer involves the links. To save links, use the LINK SAVE command. This change was made to speed up the save process.

3.5 NEW SET COMMAND OPTIONS

The SET command is now bigger than ever. Several new options have been added to allow tailoring of task and RTU parameters. The new options are:

```
BACKUP  Off|On  - Controls backup file generation by editor.
TRYS number     - Sets communications retrys for each block
                  for the task processing the command.
Link  number    - Sets default link for current RTU.
Places number   - Sets default number of decimal places.
Menu  Off|On    - Turns Auto-Menu off and on
```

3.6 AGENDA ON FIRST DAY

A bug in the way agenda processing occurred on the first day a system was started has been fixed. The symptom is that a restart in the middle of the agenda list caused the remainder of the list to be ignored until the midnight reset took place. After that, agenda processing was normal. The fix now updates the agenda time pointer to the correct time based on the time the system was started. Agenda items on the list after the startup time are processed when their time comes due.

3.7 TASK 0 STOP NOW INVALID

The TASK STOP command can no longer be used on TASK 0. This was an oversight in previous versions, and was not much of a problem. However, stopping TASK 0 causes the multi-tasker to crash, so this fix will prevent any inadvertent stops of task 0.

3.8 DEADBAND STAGE OF ALARM PROCESS

The addition of the deadband to any value type channel was made in the previous release. However, some confusion resulted for alarms that had been in alarm, and were returning to normal but were still in the deadband range. The previous release showed these points as in alarm, although their value was within the normal range (but not the deadband range). The new version adds a new stage to the alarm process to accommodate alarms that are in the process or returning to normal but have not quite made it. They are displayed with the flag DB in the alarm column on the right side of the display. The complete alarm cycle is now as follows:

No Alarm	- A point within alarm limits.
Timing	- Out of range but not long enough.
New Alarm	- Out of range and not acknowledged.
ALM	- Out of range and acknowledged.
Deadband	- Back in normal range, but within deadband
RESET	- Back in normal range, and outside deadband

Note that alarms that are set to RESET after ACK will skip the RESET stage of the cycle. Alarms with a deadband of 0 will skip that stage as well.

3.9 LINK COMMAND OPTIONS

The LINK command can now specify the keyword ALL to have the command apply to all links rather than just one link. This keyword appears wherever the link number can appear. For example, to activate all links you can enter LINK NOW ALL. This can replace the several lines previously needed to activate all

of the links in a host system. It will most likely be used in agenda processing where all RTUs are polled at the same time each day. Instead of listing all of the RTUs on separate lines, a single line can be used. Any additional RTUs added to this HOST will automatically be processed by the LINK command when the ALL option is used.

As mentioned earlier, the LINK command also has a SAVE option that causes only the link data file to be updated. This had previously been done with the general SAVE command, which no longer includes the link in its processing.

Another LINK subcommand has been added to allow a retry of a failed link. This would be a link that has exceeded its normal retry count. When in the failed state, subsequent attempts to activate the link are ignored until the link is reset (via a LINK RESET command or an F5 Key press). The new option, LINK RETRY, will cause the specified link (or the current link if not specified) to be made active again. An example application would be to have an agenda command that periodically re-activates all failed links. The links may have failed because the communications equipment was inoperable, so a periodic RETRY command will kick off the link process. The LINK RETRY only affects links that have failed. Links that are idle or active will not be affected by this command.

The new ALL option works for the RETRY option, and is the most logical choice for most applications. So, to have all failed links automatically clear themselves (and start calling again) put the command LINK ALL RETRY in one or more agenda command positions.

3.10 HALT COMMAND

The HALT command is used to stop operation of the SCADA program. Previous versions required that the word HALT be repeated twice on the line, and the second HALT had to be in upper case. So, to stop the program, you would enter HALT HALT or have the equivalent line in a command file. This release allows the HALT command to be used by itself, in which case the program will prompt the user to see if a termination is really wanted.

Program termination can also be done from the new main menu, which simply starts the HALT command.

3.11 SAY AND DISPLAY COMMANDS

These commands will be affected by the internal program changes in how values are displayed. The SET PLACES command now sets the default number of decimal places that are used when the program formats value type channels. The older format commands that may have been used on variables and channels in custom screen programs will require revision if they used an optional format specification for individual values. The older format codes are no longer supported, and they have been replaced by a more standard type format specification similar to the ones found in Basic "print using" statements.

Individual variable formatting is now done with a format string (sometimes referred to as a "picture mask") attached to the end of the variable identifier much as was done in previous versions. The only difference is that the format codes have been changed. The use of the codes is similar and should be easy to re-code. The format string still appears after the variable or channel identifier, but takes the following form of a string of characters, each of which represents a specific format action. The valid code characters are:

- `#` A digit position. If the field contains no * or @ characters, unused digits are left blank. If the field contains no sign positions (+ or -) and the number is negative, a floating minus is used.
- `*` A digit position. Unused positions are set to * instead of blanks. Only one * is needed in the picture mask to set this option.
- `@` A digit position. Unused positions are set to 0 instead of blank. Only one @ is needed to set this option. The sign will not be returned unless the mask has a + or - code in it.
- `$` A digit position. A floating dollar sign is placed in front of the resulting number.
- `-` A negative sign is placed in front of the number provided it is less than 0. Positives have a blank.
- `+` A plus sign is placed in front of the number provided it is 0 or positive. Otherwise, a - is placed in front of the number.
- `^` A decimal comma or a separator comma (see note below).
- `.` A decimal period or a separator comma. The last period or comma in the field is considered the decimal delimiter. This allows Euro style formatting where the comma and decimal have reversed uses from the US style.

A Signals the display command to use the alarm video attribute for this value. The A is stripped out of the picture mask and does not occupy a character position. When used, this must be the first character in the format string.

Cxx Center the resulting string in a character field that is xx spaces wide. Note that 2 digits must be provided, as in C10 to indicate a field of 10 or C06 for a field of 6.

Lxx Left justify in a field xx spaces wide.

Rxx Right justify in a field xx spaces wide.

Some formatting examples of the number 123456.789 are:

PICTURE MASK	RESULT	
######	123457	Note Rounding
######.#	123456.8	Note Rounding
###^###.##	123,456.79	Comma & Decimal
+######.###	+123456.789	
@@@@@@@@@@.###	0000123456.789	Leading Zeros
##########.###	123456.789	Leading Spaces
A######	123456	Alarm Video
L12######.#	123456.8____	Left Justify
R12######.#	____123456.8	Right Justify

The format string is added to the end of a variable or channel identifier by putting the accent character (`) after the name followed by the picture mask. For example, to format a channel named TOTFLOW, the entry may look like TOTFLOW`######.## where the ### part is the picture mask to be used.

COMMA CHARACTER NOTE: Because the program's command line parser is sensitive to the use of commas and spaces, it is necessary to use a non-comma character to indicate the desired comma position in the format string. To be consistent with other parts of the program, the up-caret character (^) found above the 6 key will be used. Whenever the formatter sees the ^, it will replace it internally with a comma. This is slightly different than the way other programs (such as Dbase and Basic) use picture masks. In this case, the ^ characters simply means the same thing as a comma character.

4.0 USING THE UPGRADE

There are many changes in the new version that will require modification of previous command files in order for any benefits to be obtained. Only the changes that are absolutely required are detailed here.

One is that the SELECT command must be used with an RTU name and never a number (except for the system level RTU, RTU 0). Check all command and menu files to make sure that the RTU name is being used on all Select lines and on all menu lines that feed parameters to a command file.

The new Image Save option will eliminate many of the individual value save processes that existing systems have. Implementing this feature will require that all command files be examined so that unnecessary value file processing can be identified and eliminated. This is mostly found in the start files and in any value change and save procedures.

The new SET PLACES x command replaces the older SET DISPLAY and SET DATA commands. These older commands used a format specification that is no longer supported by the program. These should be replaced with the new SET PLACES x command to establish the default setting for the program. Any channels requiring special decimal place settings should be individually configured as desired.

Also, any format strings used in command files will have to be modified to use the new style formats discussed in this document. The changes are fairly easy to do, but care must be taken to insure that all required files are modified.