TOTAL ENGINEERING SERVICES TEAM, INC.
(TEST INC.)

SCADA SYSTEM SOFTWARE

CONVERSION NOTES

RTUMON TO SCADAWARE AND SCADAWARE LITE

Revision 3

Feb 17, 1994

1. INTRODUCTION

This document lists the changes that must be made to the program files when converting from RTUMON2.EXE to RTUMON3.EXE, also referred to as SCADAWARE. Notes are also provided for the use of the "lite" version called RTULITE3 which is intended for smaller (8086) type processors. Although these changes are minimal, they are necessary for proper operation of the TEST SCADA program. The main difference between the old and new programs is that the internal structure has been modified to work under the command of a revised Multi-tasking program manager. The full version of the program, RTUMON3, also uses the DOS Protected Mode Interface (DPMI) to access the advanced features of the Intel 80386, 80486, and later processors which are required for its operation.

The Lite version, RTULITE3, does not use protected mode and will therefore run on an 8086 processor. RTULITE3 will use XMS memory to a limited degree and is therefore useful on some larger processors that do not need the advantages of the full SCADAWARE program. The two programs (RTUMON3 and RTULITE3) are similar with the differences in the Lite version listed below:

1 Small Disk Space Requirement (less than 500K total for exe)
2 No need for 386 CPU or extended memory (XMS), although XMS will be used to store task stacks.
3 No Data Base or Graphics functions.
4 No Modbus, ROC, or other Open Architecture features.
5 Simplified Image Save System with no network support.
6 Limit of 10 tasks and 10 RTUs per computer.
7 Real Numbers are stored in Borland 6 byte format in place of the IEEE 4 byte format. No math coprocessor is required.

Computers operating the full version of SCADAWARE require at least 2Mb of ram with at least 1Mb of free XMS memory. SCADAWARE also uses an 80x87 compatible math coprocessor if available. If no math chip is present, SCADAWARE will use an emulator program to simulate the action of the math chip, although with

somewhat of a speed penalty. However, the math chip is not required for most applications on high speed 386-486 processors. All internal real numbers are IEEE Ansi standard 4 byte format. Internal integers are 2 and 4 byte IEEE Ansi standard.

SCADAWARE has options for Open Architecture interfaces for Modbus and other protocols which also use the IEEE formatted numbers. Contact TEST for information on this feature.

SCADAWARE Lite will work with a much smaller processor and memory system. Basic requirements are 640K main memory with at least 580K free, and the math chip is not needed or used. XMS memory, if present, will be used with a load of approximately 200K.

In most respects, conversion from older programs is the same for SCADAWARE or SCADAWARE LITE. Only SCADAWARE will be addressed directly, with any special requirements for SCADAWARE LITE mentioned where required.

2. QUICK CONVERSION CHECKLIST

Conversion from old to new software setups that do not use new features will take only a few minutes. This document provides detailed information on the differences found in SCADAWARE. For those in a hurry, a quick summary of what needs to be done is as follows:

1. Change the START.BAT file to use the new program name (RTUMON3 or RTULITE3). Delete any obsolete command line parameters.
2. Edit the DAT file to modify the RTU and DRIVER task definition. Delete SCAN and SEC tasks from the DAT File.
3. Edit startup procedure to include task ID assignments for each task other than the serial port tasks (usually 1 and 2). This requires adding lines to identify the UTIL and DRIVER tasks with TASK x ID nnnn.
4. Modify task priority and delay lengths to shorter periods to account for the longer system tick (its now 1/18th sec instead of 1/36th).
5. Change Image load and save sequence to use new method which allows skipping of individual RTU and Link files.

A more detailed explanation of these changes is provided below.

3. REQUIRED FILES AND DOS CONFIGURATION

The new program operates as a very tame DOS which conforms to rigid DPMI specifications for memory use and processor allocation. In order for SCADAWARE to operate properly, the PC's Config.Sys file should have at least the following lines:

CONFIG.SYS

DEVICE=HIMEM.SYS	Microsoft DOS Extended Memory Manager
FILES = 24	Minimum Open files
BUFFERS = 20	Typical file buffer count

The older RTUMON2 program consisted of two files, RTUMON2.EXE and RTUMON2.OVR. SCADAWARE no longer uses an overlay file and is contained in a single file called RTUMON3.EXE (or RTULITE3.EXE). However, the RTUMON3 program requires several additional support files that the RTUMON2 program did

not. They must be located in the RTU program directory:

DPMI16BI.OVL	Borland's DPMI Server (interfaces with HIMEM.SYS)
RTM.EXE	Borland Pascal's run-time manager.
EGAVGA.BGI	Borland's Graphics Driver (if graphics are to be used).

(Note: these are not required for the lite version.)

4. START.BAT FILE OPTIONS

The START.BAT file is a DOS batch file that is used on all systems to kick off program operation and set certain configuration options that must be supplied in order for the program to start properly. The most important change for SCADAWARE is that the program name must be changed from RTUMON2 to RTUMON3.

Several program line parameters are no longer valid including:

/V - Virtual Stacks
/B - Overlay Buffer Size
/O - Overlay File name

A new parameter has been added to allow selection of the system tick interrupt used by SCADAWARE. This topic is covered in detail later in this document. The new parameter is /T and has the following options:

/T=H	Hardware Timer Tick, PC interrupt 8 at 18.2 per second.
/T=D	Dos Multiplex Interrupt 1Ch at 18.2 per second
/T=S	Soft interrupt called as often as possible with average rate at 18.2 per second.

Another parameter is available for setting the address of the security lock (dongle). Details on this option (/D=) are provided below.

A typical startup line now looks like:

```
RTUMON3 /F=EI258.DAT /T=H /D=2
```

Parameters unique to the Lite version are:

```
/B=xxxx     Increase Overlay buffer by xxxx bytes
/X          Do not use XMS for stacks even if XMS available.
/E=xx       Reserve xx K for Shelling to DOS from program.
```

5. .DAT FILE

The DAT file is an information file which is processed by RTUMON3 while it is initially starting. Changes to the DAT file which affect SCADAWARE relate to the definition of TASKS which will be used by each particular configuration. All other aspects of the DAT file are identical to earlier versions of the program.

The statement which defines each task has been changed from:

```
TASK TaskType TaskName [StackSize] [DefaultDelay] [MsgCount]
```

to the less complex:

```
TASK TaskType TaskName [DriverType] [ControlFileName]
```

Previous versions had several task types. SCADAWARE has only two:

```
RTU  - Task type which processes TSP or other protocol commands
DRIVER - Hardware interface program for physical I/O
```

If the keyword RTU is specified as the task type, only the TaskName parameter can be specified. There are no other options for an RTU task. The DriverType and ControlFileName parameters have no effect if specified for an RTU type task.

If a DRIVER task is specified as the task type, the DriverType parameter must be specified to inform SCADAWARE which type of I/O is to be used. The available keywords for specifying a driver type are:

```
METRABUS
        RELAY
        T100
        GENERIC
        CTM5
        DAC02
```

If the keyword DRIVER is specified as the task type, an additional optional parameter can be used to specify a particular control file. If the ControlFileName parameter is not specified, the default file names that will be used for each of the drivers listed above are:

```
RTU.MB
        RTU.RLY
        RTU.LB
        RTU.GIO
        RTU.CTM
        RTU.D02
```

Normally a specific file name related to the location is used in place of the above defaults. If a control file name is specified, but it does not contain a file extension, the extensions used for the default file names shown above will be assumed dependent on the type of I/O driver specified.

Older DAT files used the StackSize, DefaultDelay, and MsgCount parameters to set up memory requirements for each task. These are no longer required or allowed. Stack sizes are now fixed at 32K for RTU tasks and 16K for drivers. The message count for each task can be set at any time after the system has started with command of the form

```
TASK xx MSG xx
```

The default delay for all tasks now defaults to 2. this is the number of system ticks that a task will wait after completely running before attempting to run again. **A system tick is not 1/18 sec instead of 1/36 as in previous program versions.** This task delay value can be changed for any task by using the

```
TASK  xx DELAY yy
```

command. In this command the xx is replaced with the number of the task whose delay setting is to be changed and the yy is replaced with the number of ticks to delay. The number of ticks to delay cannot be set below 0. The STAT TASK or STAT commands can be used to display a list of all tasks and their delay values.

The message count specifies the number of messages that can be queued up for a task at one time. The default number of messages allowed for RTU type tasks is 24 and for all other tasks it is 0. This value can be changed for any task by using the TASK xx MSG yy command after the program has started. In this command the xx is replaced with the number of the task whose message count is to be changed and the yy is replaced with the number of messages allowed. The minimum setting for RTU type tasks is 10 and for other tasks it is 0. The FORCE DUMP command can be used to display a list of all tasks and the maximum number of messages allowed for each.

The **DRIVER statement** is no longer available for use in the DAT file because each I/O driver has its own task setup with a TASK DRIVER statement. In the RTUMON2 program, DRIVER statements in the DAT file were used to define the hardware drivers. Several drivers could be defined which would all be scanned by a single driver task. In the RTUMON3 (SCADAWARE) program, each hardware driver is associated with a separate task. Each driver is defined in the TASK statement that defines each driver task.

The **LOAD statement** is no longer available for use in the DAT file. In the RTUMON2 program, the LOAD statement was used to reduce the amount of memory required by the program in order to run. Keywords following the word LOAD were used to load parts of the program's executable code from the overlay file and lock them in memory during program execution. Each additional keyword used with the LOAD statement increased the amount of memory required by the program. Since there is no overlay file associated with the RTUMON3 version of the program, there is no longer a need for the LOAD statement.

6. SCAN and SYSTEM TASKS

Older program versions required definition of an alarm scan and a system task for each system. The SCAN and SYSTEM tasks are now automatically defined during program startup and can be deleted from older DAT files. The default names and IDs of these tasks are as follows:

TASK	NAME	ID
SCAN	Point Scan	Scan
SYSTEM	System	Sec

Note that the SCAN and SYSTEM tasks are not automatically defined but they are not automatically started. To start each of these tasks the commands

```
TASK SCAN START
TASK SEC START
```

are used as part of the overall system startup. The SCAN and SEC IDs are always used to identify the SCAN and SYSTEM tasks, respectively.

7. TASK IDS

Older program versions automatically assigned a fixed Task ID (nickname) to each task. SCADAWARE is more flexible and allows each task to have any nickname associated with it. By default, the ID of each task is simply its task number, except for the alarm scan and system process tasks whose IDs are SCAN and SEC, respectively. The ID of any task can be changed using the command

```
TASK TaskNumber ID TaskID
```

In this command the TaskNumber parameter is replaced with the number of the task whose ID will be changed. The TaskID parameter is replaced with the new ID string. Normally, the commands to set task IDs are put in a TSP file which automatically gets processed when the program is started. A task's ID is a convenient way to specify a particular task in a command without having to know the task's number. For example, the command

```
TASK SCAN START
```

can be used to start the SCAN task without knowing the number of the SCAN task.

8. UTILITY TASK

Many of the program's background functions must be processed by a Utility task. However, no Utility task is automatically defined by the program because it is no longer a special task. The Utility task is simply a standard TSP processor which may or may not have a physical I/O port connected to it. To define a Utility task, a task's ID must be set to UTIL as part of the startup procedure. Although this could be any task, a separate task is usually defined so that a dedicated UTIL task is always available. A Utility task is created by defining an RTU type task and then changing its ID to UTIL during startup. For example, the DAT file statement

```
TASK RTU Utility      ; define new RTU type task with name of Utility
```

can be used in the .DAT file to define an additional RTU type task. During startup, a TSP command such as

```
TASK 4 ID UTIL
```

could then be used in the startup procedure to set the task's ID to UTIL. This would be done by referencing the Task's number which is defined by the order in which it appears in the DAT file. Once the ID is supplied, the task can be referenced by that ID instead of the task number. Defining a separate Utility task allows many of the program's background functions to be processed without having to use a task that is dedicated to other things, such as communication with other units.

9. TASK DISPATCHER OPTIONS

The Task Dispatcher is a SCADAWARE function which controls the program's internal operation by allowing various parts of the program to operate as often as possible. The SCADAWARE dispatcher performs a similar function to the older RTUMON task manager which tied into (and modified) the PC's internal system clock. This timer is often called the "system tick" which operates at 18.2 ticks per second. This means that the task manager has a precise hardware interrupt which signals the processor 18.2 times per second. The old program increased this rate to 36 times per second, but this is not done in the new program.

The older task manager used the principle of "Preemptive Multitasking" which took control of the program regardless of what function it was executing at any particular instant. This allows precise control over RTUMON, but presented many problems with modern PC's which have so many varieties of hardware and software. SCADAWARE uses the principle of "Cooperative Multitasking" which requires that all tasks give up the processor as often as possible so that other tasks get a chance to run. The older task manager ruled with an iron hand; when a task's time was up, it had no choice but to give up the processor. SCADAWARE's task dispatcher is a "kinder and gentler" program which allows each task to run as long as it likes and expects each task to check in frequently to see if its allowed time has expired.

Cooperative Multitasking is the type used by Windows programs and most other PC based multitasking systems. The tradeoff over the older method is that Cooperative Multitasking is not time critical. This is because each task determines the exact instant in which it gives up the processor. The older Preemptive method used the hardware timer to start and stop each task. This made the timing more precise, but presented numerous problems with stopping tasks when they were in a critical or delicate step of their operation.

The end result is that SCADAWARE will run properly on any PC configuration that also follows all the DOS and PC rules. SCADAWARE does not modify any interrupts, BIOS settings, or DOS functions and will peacefully coexist with other programs which behave equally as well. This does not mean that SCADAWARE will multi-task on every PC in the world. It means that SCADAWARE will operate on any PC which does not have programs which modify the basic operation of the system in ways which are not documented in DOS or PC manuals.

There are three ways in which SCADAWARE's task dispatcher can control the time slicing of the various tasks. They are:

DOS: /T=D The standard DOS multiplex interrupt 1C hex is used as a time keeper to signal that a system tick (at 18.2 per second) has elapsed. *This is the default method unless overridden by a /T= parameter on the RTUMON3 command line.*

HARDWARE: /T=H The standard PC clock interrupt is trapped to precisely measure the 18.2 tick per second period.

SOFTWARE: /T=S No physical or soft interrupt is used. Instead, the task dispatcher calculates its own clock period based on the elapsed count of the PC's tick counter that is updated 18.2 times/sec.

Each of these methods has performs a similar function in a slightly different

way. The reason that the alternate methods are available is so that various operating environments can be best accommodated. The DOS method is the default and should work for most systems. Systems with particularly unusual time critical processes may get better performance by using the Hardware timer tick. Systems with networks or other complex operating environments may not work well with any interrupt related time keeper. Those systems should use the SOFT option to let SCADAWARE emulate a real interrupt with its own software routine. The desired method must be selected on the startup line with one of the following options:

```
RTUMON3  /T=H      <- Hardware
RTUMON3  /T=D      <- DOS
RTUMON3  /T=S      <- Soft (simulated)
```

10. SYSTEM TICK TIMING

The system tick rate for older versions was 36 times per second because RTUMON reprogrammed the PC's internal clock chip to produce a faster interrupt rate. SCADAWARE does not modify any PC hardware and therefore cannot increase the task switch rate. This is of little consequence to the user except in cases where the SET DELAY and SET PRIORITY commands were used to adjust performance. SCADAWARE has significantly different processing characteristics from the earlier version, so tick adjustment will likely be needed when optimizing a new setup. Generally speaking, SCADAWARE is faster and more efficient in every aspect of its operation. Note that each system tick now represents 1/18.2 seconds instead of 1/36 second as in older versions.

11. SECURITY LOCK (DONGLE) OPTIONS

A special Security Key Lock (also called a dongle) is required to operate SCADAWARE in all applications. A DEMO mode is available without the dongle which restricts certain program functions but allows most operations to be tested. Normally, SCADAWARE can locate and operate the dongle without any information from the user. Certain computer configurations, particularly Windows or OS/2, prevent the program from locating the proper parallel port to which the security key is attached. In these cases, the user must notify the program of the proper port. This can be done by logical location (lpt1, lpt2, lpt3), or by the physical address of the parallel port.

A command line parameter is provided to set the proper port. This parameter would appear after RTUMON3 as part of the normal START.BAT file. The allowable options to set the Dongle address are:

```
/D=1  LPT1  ( address determined by bios )
/D=2  LPT2  ( address determined by bios )
/D=3  LPT3  ( address determined by bios )
/D=5  Physical Port address 03BCh
/D=6  Physical Port address 0378h
/D=7  Physical Port address 0278h
```

The dongle address can also be set after the program is running with the new SET DONGLE x command. The numbers 1-3 and 5-7 can be used in place of the x to identify the location of the security key.

12. DATA BASE CONVERSION

Several differences with internal data formats prevent older SCADA database files from working with SCADAWARE. Older files will be detected and rejected automatically by SCADAWARE. Existing data base configurations will have to be re-configured with the file size, field names, and other data base file information before they can be used with SCADAWARE.

13. EXAMPLE DAT FILES

Following is an example of two DAT files. The first file is an example configuration file that could be used by the RTUMON2 program. The second file illustrates how the first file could be modified to be used by the RTUMON3 program yielding the same configuration.

RTUMON2 Version of a DAT File

```
LOAD display program report dbase
            PASSWORDS off
            VARIABLES 20
            LINKS 8

            COM 1 2400 PHONE N 8 2 200 300

            TASK, RTU, Com1 Phone, $3100, 2        ; task 1
            TASK, UTIL, Utility, $3100, 2          ; task 2
            TASK, DRIVER, I/O Driver, ,16          ; task 3
            TASK, SCAN, Point Scan, $3100, 18      ; task 4
            TASK, SEC, One Second, $2100           ; task 5

            DRIVER GENERIC gall.gio
            DRIVER METRABUS gall.mb

            NAME, GALL, CONOCO RTU/HOST

            MSG Starting RTU #1
            RTU, GALL, CONOCO RTU, 0, LOCAL
            STATUS    16
            OUTPUT    16
            AIN       15
            PID       2
            AGA3      5
            TOTAL     5
            COUNT     3
            FUNC      5
            TIMER     20
            VALUE     20

            MSG Starting RTU #2
            RTU, WEAVER, CXY WEAVER RTU, 1, REMOTE
            STATUS    16
            OUTPUT    8
            AIN       15
```

```
PID       2
AGA3      5
TOTAL     5
COUNT     3
FUNC      5
TIMER     10
VALUE     20
```

RTUMON3 Version of Same DAT File

```
PASSWORDS off
VARIABLES 20
LINKS 8

COM 1 2400 PHONE N 8 2 200 300

TASK, RTU, Coml Phone                                  ; task 1
TASK, RTU, Utility                                     ; task 2
TASK, DRIVER, Generic I/O, GENERIC, gall.gio           ; task 3
TASK, DRIVER, Metrabus I/O, METRABUS, gall.mb          ; task 4

NAME, GALL, CONOCO RTU/HOST

MSG Starting RTU #1
RTU, GALL, CONOCO RTU, 0, LOCAL
STATUS    16
OUTPUT    16
AIN       15
PID       2
AGA3      5
TOTAL     5
COUNT     3
FUNC      5
TIMER     20
VALUE     20

MSG Starting RTU #2
RTU, WEAVER, CXY WEAVER RTU, 1, REMOTE
STATUS    16
OUTPUT    8
AIN       15
PID       2
AGA3      5
TOTAL     5
COUNT     3
FUNC      5
TIMER     10
VALUE     20
```

The following lines would be added to the startup procedure:

```
TASK 2 ID UTIL

image load              ; try and load existing image
if  @image(0)   ; Image was OK
  msg Image Loaded Properly
else
  gosub GALL
  gosub Weaver
  gosub $S.Lin
endif
image on 120              ; save image every 120 seconds
```

END OF COVERSION NOTES