NASA Technical Memorandum 104174

THE MULTI-ATTRIBUTE TASK BATTERY FOR HUMAN OPERATOR WORKLOAD AND STRATEGIC BEHAVIOR RESEARCH

J. RAYMOND COMSTOCK, JR. AND RUTH J. ARNEGARD

January 1992

National Aeronautics and Space Administration

Langley Research Center
Hampton, Virginia 23665-5225

CONTENTS

ABSTRACT

The Multi-Attribute Task (MAT) Battery provides a benchmark set of tasks for use in a wide range of laboratory studies of operator performance and workload. The battery incorporates tasks analogous to activities that aircraft crewmembers perform in flight, while providing a high degree of experimenter control, performance data on each subtask, and freedom to use non-pilot test subjects. Features not found in existing computer-based tasks include an auditory communications task (to simulate Air Traffic Control communication), a resource management task permitting many avenues or strategies of maintaining target performance, a scheduling window which gives the operator information about future task demands, and the option of manual or automated control of tasks. Performance data are generated for each subtask. In addition, the task battery may be paused and onscreen workload rating scales presented to the subject. The MAT Battery requires a desktop computer (80286/386/486 processor) with color graphics (at least 640 x 350 pixel). The communications task requires a serial link to a second desktop computer with a voice synthesizer or digitizer card.

TASK BATTERY DESCRIPTION

The MAT Battery primary display, depicted in Figure 1, is composed of four separate task areas, or windows, comprising the monitoring, tracking, communication, and resource management tasks. A scheduling window is also presented to the subject. These are described individually in the sections to follow.

The experimenter has control of many task parameters through a setup program. Details about setup and operation of the MAT Battery are presented in the section on "Installing and Running the Battery." Events presented to the subject are controlled by a script file which can be easily edited to manipulate task loading. Details about making script files are covered in the section on "Script Files." Information and examples of performance and rating data files generated by the task battery are presented in the section entitled "MAT Data Files." Research conducted using the task battery may be found in Arnegard (1991) and Arnegard and Comstock (1991).

Monitoring

The system monitoring tasks are presented in the upper left window of the display. The demands of monitoring gauges and warning lights are simulated here. The subject responds to the absence of the Green light, the presence of the Red light, and monitors the four moving pointer dials for deviation from midpoint.

The two boxes in the upper portion of this window represent the warning lights. The light on the left is normally "on," as indicated by a lighted green area. The subject is required to detect the absence of this light by pressing the "F5" key when the light goes out. The light on the right is normally "off." When the red light comes on, the subject's task is to respond by pressing the "F6" key when he or she detects the presence of that red light. If using the mouse, correct responses to the warning light task requires pointing to the proper boxed area and pressing a mouse button. If the subject does not detect either abnormality, the situation reverts to normal status after a selected timeout period (set by the experimenter).

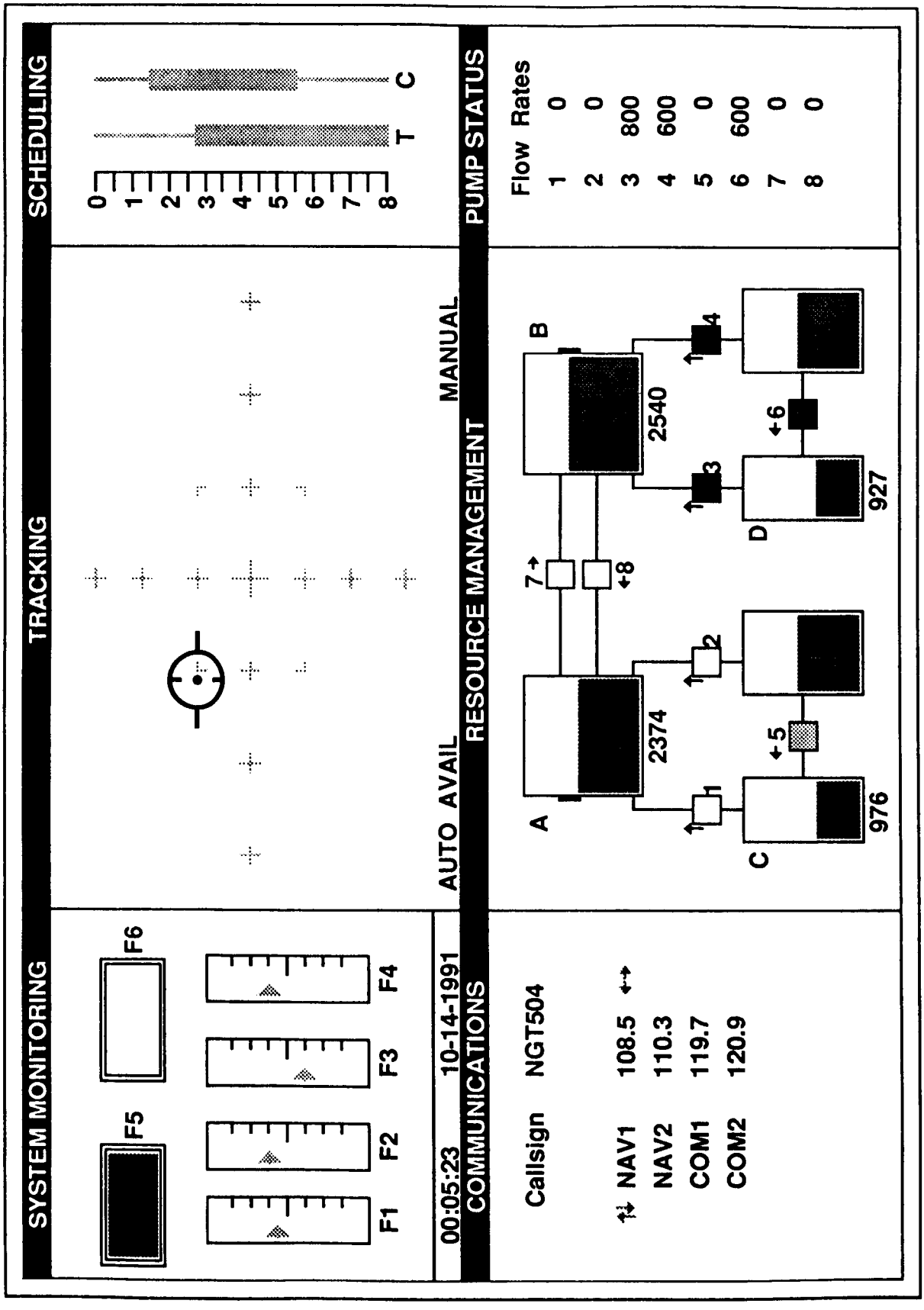

Figure 1. Multi-Attribute Task Battery Display

The moving pointer monitoring task consists of four vertical scales with moving indicators. In the normal condition, the pointers fluctuate around the center of the scale within one unit in each direction from center. If cued by the script, a scale's pointer shifts its "center" position away from the middle of the vertical display to a position either one deviation above or one deviation below the center. The subject is responsible for detecting this shift, regardless of direction, and responding by pressing the corresponding function key (F1 to F4). The appropriate response key is identified below each vertical display.

When this out-of-range status is correctly identified by the subject, feedback is given in two ways. First, the pointer of the correctly identified scale moves immediately back to the center point and remains there without fluctuating for a period of 1.5 seconds. Additionally, a bar at the bottom of the dial is illuminated in yellow. If the subject fails to detect an abnormality in the probability monitoring task, the fault disappears after a preset time interval (selected by the experimenter).

If using the mouse version, the subject responds by placing the mouse cursor on the appropriate moving pointer scale area which is out of range and pressing a mouse button.

Tracking

The demands of manual control are simulated by the tracking task. This task is located in the upper middle window. Using the Joystick, the subject keeps the target at the center of the window. This task can be automated to simulate the reduced manual demands of autopilot.

This task is a compensatory tracking task and the subject's task is to keep the target in the center of the window, within the dotted lines which form a rectangle. This task can be operated in either manual or automatic mode with the current mode shown by "MANUAL" or "AUTO" displayed in the lower right corner of the window. In the AUTO mode no action is required of the subject.

Scheduling Window

The scheduling window lets the subject "look ahead" for up to eight minutes in the future at activity of the Tracking and Communication tasks. The scheduling window displays the beginning and/or ending (and duration) of these two tasks. The timelines are identified by "T" for the tracking task, and "C" for the communication task. The scheduling window allows the subject to "see" from 0.0 (present time) to 8.0 minutes into the future. The bold bars indicate times when one of these tasks can be expected to be performed by the subject. The thin lines indicate times during which these tasks are usually not required of the subject.

Communications

The communications task simulates receiving audio messages from Air Traffic Control. The subject must pay attention to messages for his or her own callsign "NGT504" and make frequency changes on the proper Navigation or Communication radio. The task messages are presented to subjects through headphones. These messages begin with the six-digit call sign, repeated once, and a command to change the frequency of one of the radios listed on the screen. The subject must discriminate his or her callsign, "NGT504,"

from other three-letter, three-number combinations. The subject's callsign is displayed at the top of the Communications window. Subjects change navigation and communication frequencies by using the keyboard arrow keys. The up and down arrow keys are used to select the appropriate navigation or communication radio and the left/right arrow keys increase or decrease the selected radio frequency in increments of 0.2 Mhz. Once the appropriate frequency selection is made, pressing the "Enter" key acknowledges the completed change.

If using the mouse, the appropriate navigation or communication radio is selected by placing the mouse cursor on the desired radio or its frequency (such as "COM2 120.9") and pressing a mouse button. Frequency changes are made by placing the mouse cursor over the left or right arrow and pressing a mouse button. Positioning the mouse cursor over the "Enter" box and pressing a mouse button acknowledges the completed change.

Resource Management

The demands of fuel management are simulated by the resource management task. The goal is to maintain tanks A and B at 2500 units each. This is done by turning On or Off any of the eight pumps. Pump failures can occur and are shown by a red area on the failed pump.

Both the Resource Management and Pump Status windows are utilized for the resource management task. The Resource Management window provides a diagram of the fuel management system. The six large rectangular regions are tanks which hold fuel. The green levels within the tanks represent the amount of fuel in each tank, and these levels increase and decrease as the amount of fuel in a tank changes.

Along the lines which connect the tanks are pumps which can transfer fuel from one tank to another in the direction indicated by the corresponding arrow. The numbers underneath four of the tanks (Tanks A, B, C, and D) represent the amount of fuel in units for each of the tanks. This number is updated every 2 seconds as the amount of fuel in the tanks increases or decreases. The maximum capacity for either Tank A or B is 4000 units. Tanks C and D can contain a maximum of 2000 units each. Finally, the remaining two supply tanks have an unlimited capacity.

Subjects are instructed to maintain the level of fuel in both Tanks A and B at 2500 units each. This desired level is indicated graphically by a tick mark in the shaded bar on the side of these two tanks. The numbers under each of these tanks provide another means of feedback for the subject. The shaded region surrounding the tick mark represents acceptable performance. Tanks A and B are depleted of fuel at a selected rate (default of 800 units per minute). Therefore, in order to maintain the task objective, subjects must transfer fuel from the lower supply tanks.

The process of transferring fuel is accomplished by activating the pumps. Each pump can only transfer fuel in one direction, as indicated by the corresponding arrow. These pumps are turned on when the corresponding number key is pressed by the subject. Pressing the key a second time turns that particular pump off and so on. If the mouse is used, placing the mouse cursor on the appropriate pump and pressing a mouse button toggles the pump from "off" to "on" or from "on" to "off." The pump status is indicated by the color of the square area on each pump. When that area is not lighted, the pump is off. A green light in this area indicates that the pump is actively transferring fuel.

The flow rates for each pump are presented in the "Pump Status" window. The first column of numbers represents the pump number, one through eight. When a pump is activated, its flow rate is presented next to the pump number in this window. When a pump is off, its flow rate is zero.

At the discretion of the experimenter pump failures (faults) may be presented to the subject. These are indicated by the appearance of a red light in the square on the pump. When this occurs, the pump which is in the fault mode is inactive. Fuel cannot be transferred through that pump until the fault is corrected. The subject has no control over the fault correction. The duration of the fault is written into the script that directs the program. When the fault is corrected, the status of that pump is automatically returned to the "off" mode, regardless of its status before the fault condition.

Likewise, when a tank becomes full to capacity, all incoming pumps are automatically turned "off." For example, if all of the pumps were activated and Tank A reached its capacity of 4000 units, Pumps 1, 2, and 8 would automatically turn "off." Furthermore, if a tank were to become totally depleted of fuel, all outgoing pumps would be deactivated.

At the onset of each experiment, Tanks A and B contain approximately 2500 units of fuel each and Tanks C and D contain approximately 1000 units of fuel each. All pumps are off at the beginning of the task, leaving all strategic action to the operator's discretion.

The resource management task can be automated. In the automatic mode switching on and off of the pumps is accomplished without subject intervention. To indicate automatic mode, all numeric quantities are no longer displayed and the label "AUTO" appears beneath tanks "A" and "B." When switched back to Manual mode, pumps remain on or off depending on their last status during "AUTO."

Workload Rating Scale

In order to obtain a self reported assessment of the subject's workload during task performance, the task display may be "frozen" and a workload rating screen presented. The rating scale used is the NASA Task Load Index (NASA TLX), developed by the Human Performance Research Group at the NASA Ames Research Center (Hart and Staveland, 1988), and is a multi-dimensional rating scale. A weighted average of ratings on six subscales provides an overall workload rating. These subscales are: Mental Demand, Physical Demand, Temporal Demand, Own Performance, Effort, and Frustration. Subjects are required to rate their perceived exertion on five of these subscales (except Own Performance) on a graded scale from "Low" to "High." The Own Performance subscale ranges from "Good" to "Poor." For further information about the NASA-TLX and scoring of the scale, refer to the NASA Task Load Index users manual available from the Human Performance Research Group, NASA Ames Research Center, Moffett Field, California.

The TLX rating scale can be presented to the subject at any time during the operation of the battery. A code for the onset of rating scale presentation can be added to the script which generates events for the MAT Battery. At the time of TLX presentation, specified by the script, a second screen will appear in place of the MAT Battery. This screen is depicted in Figure 2. During the presentation of the second screen, all MAT Battery activity is paused and elapsed time frozen until the subject either exits from the TLX screen or the maximum time for that screen is reached (Preprogrammed at 60 seconds). Upon return to the MAT Battery screen, normal operation of the battery resumes.

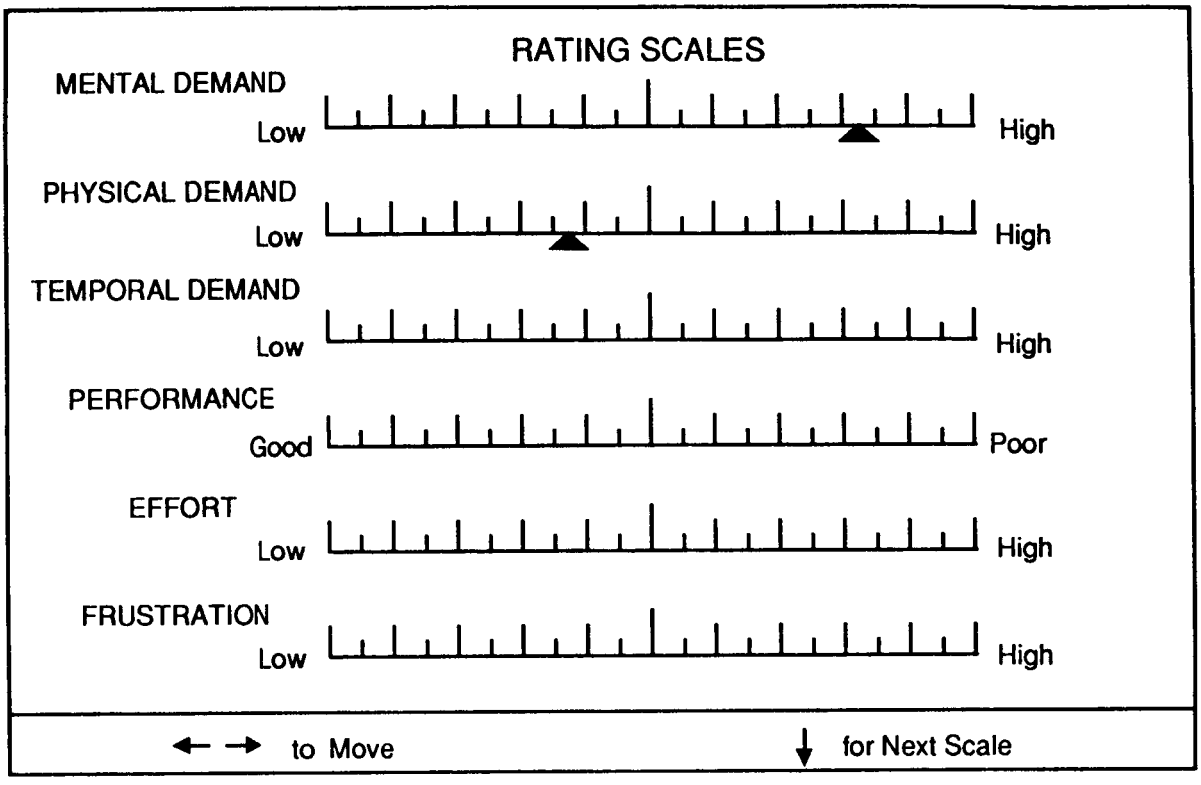

Figure 2. Workload Rating Screen Showing Initial Instructions

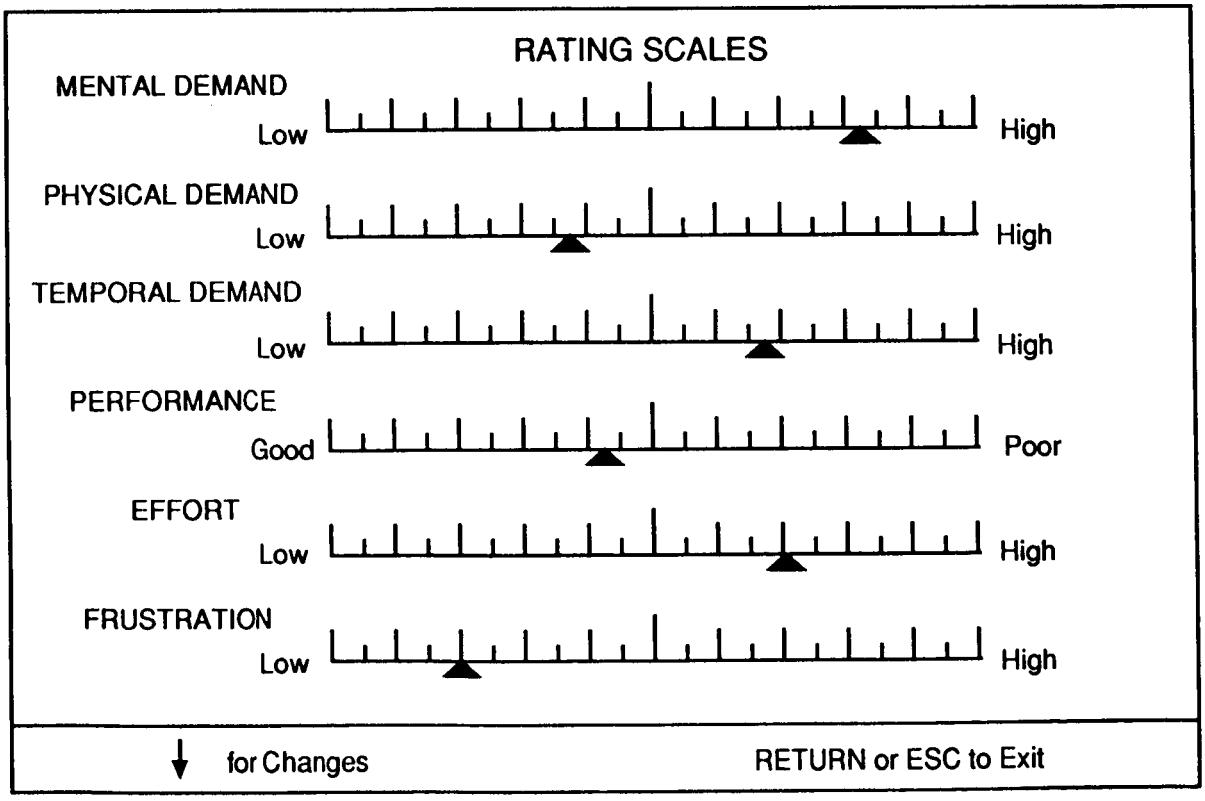

Figure 3. Workload Rating Screen Showing Exit Instructions

When the TLX screen is first presented, a pointer appears in the middle of the first subscale (i.e., 50). Each gradation represents 5 points; thus, potential scores on each subscale range from 0 to 100. The subject will begin with the first subscale and use either the mouse or the right and left arrow keys to select a rating value. After making a rating on the first subscale, the subject can press the left mouse button, the space bar, or the down arrow key to progress to the next scale. After doing this, the pointer of the second subscale will become yellow and the first pointer will turn gray. The pointer of the scale which is active will be slightly larger and illuminated in yellow as opposed to the gray coloring of the other pointers. This process continues until the subject has responded to all six subscales.

After the subject has entered a response for the sixth subscale, the command options shown at the bottom of the screen change, as shown in Figure 3. At this point, the subject can press the right mouse button, ESCape key, or Return key to exit to the MAT Battery primary display. However, if the subject wishes to change any of the responses that have already been made, this can be done by pressing the left mouse button, the down arrow key, or the space bar. For instance, the subject can press the down arrow key three times to return to the third subscale without changing any of the scores on the first two subscales. The pointer will turn yellow when a particular rating scale is active and can be changed. Changes are made the same way that initial ratings were made, by mouse movement or left /right arrow keys. After the subject is satisfied with all responses, the right mouse button, ESCape key, or Return key will return him or her to the MAT Battery primary display.

Rating Scale Weightings

After completing the experimental session in which workload ratings were obtained, the program AFTERMAT may be used to obtain scale weightings from the subject. These weightings are required to calculate the NASA TLX weighted workload scores. AFTERMAT requires the subject to make factor comparisons of each pair of scale titles (i.e., Effort vs. Performance). This program is run following any session requiring the subject to utilize the NASA-TLX Workload Rating Scale. Each pair of scale titles (total of 15) is presented individually on the computer screen. The subject uses the mouse or up/down arrow keys to select the variable that he or she felt was more important to the experience of workload. After each choice is made, the next pair of scale titles appears on the screen, until all comparisons are made.

Another support program, APPLYWT, is used by the experimenter to apply the rating scale weightings obtained through AFTERMAT to the appropriate rating data file or files. APPLYWT prompts the experimenter for the appropriate weighting file and the appropriate data file to apply the weights to. The result of running APPLYWT is a new rating data file like the original with the addition of mean ratings of the six subscales and the weighted combination of the six subscales. Sample data files generated by these two programs appear later in the annotated data file section.

RUNNING THE TASK BATTERY

Installation

The MAT Battery was written using Microsoft QuickBASIC 4.5, using Screen mode 9 (EGA mode). Hardware requirements are a PC or compatible with EGA or VGA color graphics. An 80286/386/486 processor machine is highly recommended.

The MAT Battery was designed to use a serially linked second computer to generate the voice messages for the Communications task. The recommended procedure is a serial link between the COM1 port of the task computer through a null modem adapter and cable to the COM1 port of the voice computer. While most any voice synthesizer or voice digitizer card may be employed in the voice computer, the MATREMX program and support files were designed to work with the Heath Voice Card (Model HV-2000). (This Voice Card is available through the Heath Company, Benton Harbor, Michigan 49022). The voice hardware and software has been tested on a Heath (Zenith) 158, a Leading Edge Model D, and a generic 8088 machine. The MAT Battery task computer may be used independent of the voice computer if no computer synthesized voice messages are desired or if some other method of presenting auditory messages is devised.

Although the MAT Battery may be run from floppy disks, it is highly recommended that it be installed on a hard drive. This is recommended in order to minimize the time required for program loading and data file writing and also to insure that sufficient disk storage space is available for the data files generated.

A suggested installation method is to create a Subdirectory entitled "MAT" and then to copy all of the MAT Battery program and support files into that Subdirectory. Data files generated will also appear in the Subdirectory.

The second computer with the Voice Card installed and serial link to the Battery computer should have the program MATREMX.EXE, and voice message files MSG57.VOI and MSG58.VOI installed. In addition, files supplied with the Voice Card hardware are required, as well as an addition to the computer boot drive CONFIG.SYS file as specified in the Voice Card manual.

Starting the MAT Battery

If the Voice computer is being used, first start the program "MATREMX" on the voice computer. No other input is required as subsequent control of this program is from the MAT Battery computer. The ESCape key exits from this program.

To start the MAT Battery enter "MAT" or "MATSET4." Either command starts the program MATSET4 which permits (1) setting up the serial port, (2) selecting, reading, checking, and converting the script file, (3) setting of tracking task gain, (4) changing task timeouts and tracking task data recording interval, and (5) selection of MAT task to run (Normal or Alternate version). MATSET4 creates two files required by the MAT Battery main program and then starts the main program. MATSET4 should ALWAYS be used to start the MAT Battery.

Subject Controls and Commands

Table 1 shows the controls and commands that are used by the subject when performing the tasks on the battery. The section entitled "Instructions to Subjects," presented later, provides examples of how each task may be introduced to a subject.

Table 1: MAT Battery Controls and Commands for Subject Use

Key/Device	Task Control Function (Subject Use) *
F1 to F4	Responses to Monitoring Task Moving Pointers
F5, F6	Responses to Monitoring Task Warning Lights
↑ ↓	Communication Task Radio Selection
← →	Communication Task Frequency Change
Enter	Acknowledge Communications Task frequency change completed
1 to 8	Resource Management Pump Controls
Joystick	Used to control position of Tracking Target during "MANUAL" tracking
Mouse	(If implemented) Positioning mouse cursor and "Clicking" on a screen area may be used to control monitoring, communications, and resource management tasks and to select "AUTO" or "MANUAL" tracking when permitted

* *Active keys for the Workload Rating Screen are displayed on the Workload Rating Screen.*

Experimenter Controls and Commands

Table 2 presents the controls and commands that the experimenter may use to control operation of the task while it is running. Table 1 and Table 2 commands may also be entered from the keyboard of the serially linked Voice computer, except for Rating screen inputs and Unpausing the task.

Screen background color may be selected at the initial screen showing the task display. On this screen, where options are RETURN to begin or ESC to exit, the "X" or "C" keys may be used to vary screen background color. Color number is shown in lower right of tracking task window. Background of zero is the default, but color 8 may be used for a dark blue background. The range is 0-63 but obviously some combinations are not so visually pleasing.

Table 2: MAT Battery Commands for Experimenter Use

Key(s)	Functions Available for Experimenter Use *
Alt-F1 to Alt-F4	Begin offset or bias on Scale 1 to 4
Alt-F5 Alt-F6	Present green light trial (Green off) Present red light trial (Red on)
Shift-1 to Shift-8 Alt-1 to Alt-8	Fail Pump 1 to 8 (Red pump area) Fix Pump 1 to 8 (Pump off)
Shift-F1 Shift-F2	Switch Resource Mgmt to AUTO Switch Resource Mgmt to MANUAL
Shift-F3 Shift-F4	Increase Automation (Fewer Tasks, Tasks Easier) Decrease Automation (More Tasks, Tasks Harder) Each Shift-F3 or F4 changes Task Level one step, the range is 1 to 6
Ctrl-F1 Ctrl-F2	Send Ownship Code to Voice Computer Send Other Ship Code to Voice Computer
Ctrl-F3	Present Workload Rating Screen (Task is paused during ratings; automatically exits rating screen after 1 minute)
Ctrl-F4 Ctrl-F5 Ctrl-F6	Switch Tracking Task to LOW difficulty Switch Tracking Task to MEDIUM difficulty Switch Tracking Task to HIGH difficulty Tracking difficulty changes may be made with task in "MANUAL" or "AUTO" mode
Ctrl-F7	Switch Tracking Task to "MANUAL" Mode Also permits F7 and F8 to change tracking gain and display tracking gain value
Ctrl-F8	Switch Tracking Task to "AUTO" Mode Also disables use of F7 and F8 to make tracking gain changes
Ctrl-F9	Pause / Unpause All Tasks
Ctrl-F10	Exit MAT and Save Data

* *The Experimenter can also control the task from the keyboard of the serially linked Voice computer using the Table 1 and Table 2 key commands (Exceptions: Rating screen inputs and Unpausing the task).*

CONTROL OF TASK EVENTS

Script Files

A script file controls the sequence and timing of events in the task. Script files are ASCII files and may be edited or created using wordprocessors in a NON-Document, DOS-Text, or ASCII mode. They may also be created by computer programs designed for that purpose. In order to be displayed in the directory of script files of the task setup program, script files must have a .DTB extension.

The format of the script file is illustrated below. Each line contains 3 sets of numbers corresponding to the time of the event and also the event type in the following order: (1) Hours, (2) Minutes, (3) Seconds, and (4) Event Script code. The event script code can be written by the use of numerical or text code. The acceptable codes and the corresponding event descriptions are presented in Table 3. The numbers and event script codes should be separated by spaces. Event times should be in ascending chronological order from the beginning to the ending of the task, as illustrated below.

```
Hours
| Minutes
| | Seconds (Control Time of Event)
| | |
| | | Event Script Code (Numerical Codes, See Table 3)
| | | |
0  0 32  42
0  0 44  51
0  1  0  82
0  1 19  32
0  1 21  61
.
.
.  (OR)
.
.       Event Script Code (Text Codes, See Table 3)
.       |
0 20 31  RED
0 21  0  MANUAL
0 22 19  SCALE 3 UP
0 22 30  FAIL PUMP 1
0 22 45  RED
0 22 46  SCALE 2 UP
0 23 17  SCALE 3 DOWN
0 23 29  FIX PUMP 1
0 23 40  RED
0 23 59  END
```

Table 3: Event Script Codes and Event Descriptions

Script Code (Text)	(Numerical)	Event Description
RATING	10	Present Rating Scale (NASA-TLX)
SCALE 1 UP	11	Monitoring: Scale 1 (F1) Up
SCALE 1 DOWN	12	Scale 1 (F1) Down
SCALE 2 UP	21	Monitoring: Scale 2 (F2) Up
SCALE 2 DOWN	22	Scale 2 (F2) Down
SCALE 3 UP	31	Monitoring: Scale 3 (F3) Up
SCALE 3 DOWN	32	Scale 3 (F3) Down
SCALE 4 UP	41	Monitoring: Scale 4 (F4) Up
SCALE 4 DOWN	42	Scale 4 (F4) Down
GREEN	51	Monitoring: GREEN Light Off (F5)
COMM TASK START	55	Comm. Scheduling Window: Start Task Indicator
COMM TASK END	56	Comm. Scheduling Window: Stop Task Indicator (Controls Scheduling Window Only, Not Task)
COMM TASK OWN	57	Comm. Task: Own Callsign Message
COMM TASK OTHER	58	Comm. Task: Other Callsign Message
RED	61	Monitoring: RED Light On (F6)
MANUAL	71	Tracking: Start MANUAL Mode
AUTO	72	Tracking: Start AUTO Mode
AUTO END	73	Tracking: Display AUTO END Message (until code 71 or MANUAL)
TRACKING LOW	74	Switch tracking to LOW difficulty
TRACKING MEDIUM	75	Switch tracking to MEDIUM difficulty
TRACKING HIGH	76	Switch tracking to HIGH difficulty Tracking difficulty changes may be made with task in "MANUAL" or "AUTO" mode.
RESOURCE AUTO	78	Switch Resource Mgmt to "AUTO"
RESOURCE MANUAL	79	Switch Resource Mgmt to "MANUAL"
FAIL PUMP 1	81	Fail Pump 1 on Resource Management
FAIL PUMP 2	82	Fail Pump 2
FAIL PUMP 3	83	Fail Pump 3
FAIL PUMP 4	84	Fail Pump 4

(continued)

Table 3: (Continued) Event Script Codes and Event Descriptions

Script Code (Text)	(Numerical)	Event Description
FAIL PUMP 5	85	Fail Pump 5 on Resource Management
FAIL PUMP 6	86	Fail Pump 6
FAIL PUMP 7	87	Fail Pump 7
FAIL PUMP 8	88	Fail Pump 8
FIX PUMP 1	91	Fix Pump 1 / Turn off Pump 1
FIX PUMP 2	92	Fix Pump 2 / Turn off Pump 2
FIX PUMP 3	93	Fix Pump 3 / Turn off Pump 3
FIX PUMP 4	94	Fix Pump 4 / Turn off Pump 4
FIX PUMP 5	95	Fix Pump 5 / Turn off Pump 5
FIX PUMP 6	96	Fix Pump 6 / Turn off Pump 6
FIX PUMP 7	97	Fix Pump 7 / Turn off Pump 7
FIX PUMP 8	98	Fix Pump 8 / Turn off Pump 8
END	99	End of Script, Terminate Task
LEVEL 1	101	Set Level to Lowest Automation, Tasks Hardest
LEVEL 2	102	.
LEVEL 3	103	.
LEVEL 4	104	.
LEVEL 5	105	.
LEVEL 6	106	Set Level to Highest Automation, Tasks Easiest

Note: The script codes written in text are the minimum letter combinations necessary for interpretation. However, additional text may be added beyond the minimum code (Numerical or Text) for annotative purposes. A script file may contain both Numerical and Text event codes. Either UPPERCASE or lowercase text may be used.

Keyboard Control

As noted previously, Table 2 presents the controls and commands that the experimenter may use to control operation of the task while it is running. Table 1 and Table 2 commands may be entered from the keyboard of the serially linked Voice computer, except for Rating screen inputs and Unpausing the task.

Control of task events and pacing usually depends on the type and frequency of events in a script file. Optional "LEVEL" change codes or commands can cause events in the script to be ignored or overruled. For example, at LEVEL 6, several tasks would be set to "AUTO" and the script events for the remaining tasks may ignored. Likewise, scheduling window information would not reflect task LEVEL changes. Although LEVEL changes may be made in the script, normal use of LEVEL changes would be reserved for unique experimental situations in which real-time task loading changes are desired. The combination of tasks active at the six task levels may be modified in the program.

INSTRUCTIONS TO SUBJECTS

General Introduction

The overall purpose of this research is to understand human performance during varying levels of "busyness." The task which is displayed before you is a computerized simulation of some of the kinds of tasks that pilots perform. Each window of the screen represents a different kind of task, as indicated by each heading: system monitoring, tracking, communications and resource management, for example. While eventually you will be asked to perform all of these tasks at the same time, to become familiar with the different tasks you will be introduced to each task one at a time.

System Monitoring

All of the information required to perform the monitoring task is displayed in the upper left window of the screen. This task consists of two parts: lights and dials. You will be monitoring the two lights at the top of this window for any changes. You will also be monitoring the four dials beneath them for any directional changes in the fluctuation of the pointer. Let me demonstrate how this task will appear in the "normal condition."

Let me first explain the changes that can occur with the lights. As you can see, during normal conditions, the left light is on in green. But occasionally, this green light will go out. When this happens you must press the "F5" key as indicated next to that light. With the mouse version, you must move the mouse cursor within the rectangular green light area and press a mouse button. You will receive feedback in that the light will immediately turn back on. Any questions?

The second light is normally off, but occasionally, a red light will turn on in this position. To respond to this, you must press the "F6" key, also indicated next to that light or move the mouse cursor to the red light area and press a mouse button. Again, as soon as you respond correctly, the red light will disappear. To summarize, you will be monitoring the lights for the absence of the green light, and for the presence of the red light on the right. When these events occur, you must respond to them. Now, I'll let you practice for a few seconds. (One occurrence of each light condition is presented to the subject). Any questions?

The second part of this task consists of monitoring the four dials below the lights. Normally, the yellow pointers fluctuate from one unit below to one unit above the center line. Is that obvious to you? Your task is to monitor these four dials and detect any change from the normal fluctuation of the pointer. In other words, if the pointer of one of these dials fluctuates either above or below the normal range, you must respond. The correct response is the key that is indicated below the dial which is out of range. If subject is using the mouse, simply move the mouse cursor within the out-of-range dial area and press either mouse button. (One example each of below and above range fluctuations is presented).

You'll notice that feedback to a correct response is given by the presence of a yellow bar at the bottom of the dial that was out of range and a return to center of that dial pointer. Again, the abnormal fluctuation can occur in either direction - above or below - but there is only one response per dial. Any questions? Now, I'll let you practice with the dials for one minute. (Subject is presented with two signals to respond to).

During the experimental task, you will be monitoring both lights and dials and looking for changes in any of them. Respond as quickly and as accurately as possible.

Tracking

All of the information that you need to perform the tracking task is displayed in the upper middle section of the screen in the section titled "Tracking." Are you right-handed? (Mouse and pad are placed on the side of handedness).

Have you ever used a mouse (or joystick) before?

IF YES...
What have you used the mouse/joystick for? (This is noted in log).
Go to (*** below).

IF NO...

The mouse is one way of controlling your position on the screen. Typically, you would use the arrow keys to move up and down and right and left. The mouse is another way to move around. The mouse pad correlates roughly to the area of the screen so that if you move the mouse up on the pad, your position on the screen is moved up accordingly. The same is true for moving right, left, diagonally, etc. Basically, if you wish to move to a different area of the screen, you must move your mouse in that identical direction on the mouse pad.

One important point to remember about the mouse is that you may have to move the mouse farther with your hand than the distance on the screen indicates (experimenter demonstrates this). Since there is not much room on the pad, you may have to pick up the mouse and set it back down to continue your movement. As long as the mouse is not touching the pad or any other surface, you will not affect your position on the screen. If the joystick is to be used with the tracking task, similar instructions are given.

***

The overall purpose of this task is to keep the airplane symbol, represented by the green circle, within the dotted rectangular area in the center of this task.

If you do not control the plane with the mouse, the plane will drift away from the center. You must control the plane with movements of the mouse/joystick. Basically, you must compensate for this random drifting by pulling the plane back to center with corresponding movements with the mouse/joystick. For example, if the plane is drifting to the right, moving the mouse/joystick to the left will return the ship to center. Most of the time, however, you will be working in two dimensions: horizontal and vertical; so you will be making many diagonal movements. Let me demonstrate (experimenter controls mouse/joystick). Watch both the screen and the mouse. You'll notice that, if the plane is away from the center, you must make rather large movements to return it. If the plane is already in the center, smaller movements will be required. Now, you practice for a few minutes before we start the experiment.

Remember, the overall purpose of this task is to keep the plane in the center rectangular area. Try to maintain this at all times. If the plane leaves the rectangular area, try to return the plane to center as quickly as possible.

Communications

All of the information that you need for the communications task is displayed in the lower left corner of the screen under the title "Communications." The overall purpose of this task is to discriminate between audio signals which will be presented through headphones and to respond as indicated.

The messages that you will hear begin with a six-digit call sign followed by a command. These call signs consist of three letters followed by three numbers. You must respond only when you hear your personal call sign which is NGT504. This number will remain on the screen at all times as a reminder to you. Any other call sign is not meaningful to you. Your call sign will sound like this. Please listen to the first part of the following message: (Ctl F1). The call sign will be presented twice so that you can identify it precisely. Other call signs will be presented in the same way. The following is an example of a different call sign: (Ctl F2). Do you feel comfortable discriminating between your call sign and other call signs? (If no, repeat other examples).

The second part of the message involves a command and you must respond to those that follow your call sign only. Do not respond if the message begins with a different call sign. The second part of the message is a command requiring you to change one of the frequency numbers listed on the screen. Let me explain this part of the task. There are four channels listed in the left column on the screen: NAV1, NAV2, COM1, COM2. These will be referred to in the audio message as: First Navigation, Second Navigation, First Radio and Second Radio. Notice that COM1 and COM2 are referred to as radio channels. In the right column are the frequency numbers that correspond to each channel.

The following is another example of one of these messages: In this message, you were directed to change second navigation to 111.6. That channel is the second channel listed on the screen, or NAV2. You must change the frequency of this channel from what it is now to 111.6. In order to do this, you must use the arrow keys on the right part of the keyboard. The up and down arrow keys change radios for you. Try using these keys.

If using the mouse, move the mouse cursor to the desired radio and press the mouse button to select that radio. After the correct channel has been selected, move the pointer to the left/right arrows located next to the frequency and press the mouse button once for each increment or decrement that you wish to select until the correct frequency is attained.

You can see that you are moving up and down through the different radios. Now go to second navigation.

In order to change the frequency numbers, you must use the left and right arrow keys (click on these areas with the mouse). The right arrow key increases the number by intervals of 0.2 and the left arrow key decreases the number by intervals of 0.2. Now change this frequency to 111.6. Let's try a few more examples. Notice that the range of the navigation frequencies is 108.1 to 117.9. The radio channels range from 118.1 to 135.9. Do you feel ready to begin? Let's first adjust the headphones for you. Put the headphones on and let me know if the volume level is adequate.

When you have completed the requested frequency change, press the "Enter" key to acknowledge that the change is complete (or click the mouse on the "Enter" area).

Remember, the overall goal of this task is to distinguish correctly messages beginning with your call sign and respond to those commands. Please respond as quickly and accurately as you can.

Resource Management

All of the information that you will need to do the resource management task is contained within the two lower right windows with the headings "Resource Management" and "Pump Status."

This task is considered a fuel management task. The rectangles are tanks which hold fuel, the green levels within the tanks that increase and decrease are fuel, and along the lines which connect the tanks are pumps which transfer fuel from one tank to another in the direction that is indicated by the arrow. The numbers underneath four of the tanks represent the amount of fuel in units for each of these tanks. This number will be increasing and decreasing as these levels change. The capacity for the main tanks, A and B, is 4000 units each. The supply tanks, C and D, contain a maximum of 2000 units each. The supply tanks on the right of each three-tank system have an unlimited capacity - they never run out.

Your overall goal with this task is to maintain the levels of fuel in tanks A and B at 2500 units. This critical level is indicated by the tick mark in the shaded area on the side of each of these tanks. This level is also indicated by the numbers underneath each tank. It is acceptable to keep the level of fuel within the shaded area between 2000 and 3000 units. However, optimum performance is obtained when Tanks A and B are at 2500 units.

In order to meet this criterion, you must transfer fuel to tanks A and B in order to meet this criteria because tanks A and B lose fuel at the rate of 800 units per minute (default; select at setup). So you can see that with their present levels of approximately 2400 units each, these tanks would become empty in slightly more than 3 minutes without the transfer of additional fuel. Tanks C and D only lose fuel if they are transferring fuel to another tank.

Let me now demonstrate the process of transferring fuel. Notice that every pump has a number, a square box and an arrow next to it. The arrow indicates the direction through which fuel can be transferred with that pump. Each pump can only transfer fuel in one direction. The pumps are activated by pressing the key corresponding to the pump that you wish to activate. Use the number keys across the top of the keyboard rather than those on the right hand side of the keyboard. If the subject is using the mouse, they are instructed to move the mouse cursor to the square box on the pump that they wish to activate and press the mouse button. A second press of the mouse button in the same area will turn the pump off. I'll demonstrate by turning all of the pumps on.

When I turned the pumps on, two things occurred. First, the square on each pump turned green. That means that the pump is actively transferring fuel. When the pump is off, the square is black. The second change on the screen is the numbers that appeared in the "Pump Status" window. Let's focus on that now.

Under "Pump Status," two columns of numbers are present. The first column, numbers one through eight, indicate the pump numbers and these correspond directly to the pumps in the diagram. The second column of numbers indicates the flow rates in units per minute of each pump when that pump is on. For example, Pump 1 transfers 800 units of fuel per minute from Tank C to Tank A. The flow rate for any given pump is only

presented if that pump is on and actively transferring fuel. Are the flow rates clear to you?

So far, you've seen two conditions for the pumps: on and off. Pressing the pump number key once turns the pump on; pressing the key again turns that pump off, and so on. A third condition is the fault condition, over which you have no control. At various times throughout your task, you will see the status indicator on a pump turn red. This means the pump is inactive as long as that red light is present. You will not be able to use this pump until the red light goes out. However, you must be aware that when the fault is corrected and the red light goes out, that pump will automatically be returned to the "off" status (without any light). Even if you had turned that pump on before the fault occurred, the pump will not be returned to an "on" condition. You will have to turn it on again if that is what you wish. Any questions?

Along the same line, if a tank fills up to its capacity, all incoming pump lines will be turned off automatically. This is because a full tank cannot receive any more fuel. You will have to turn those pumps back on at a later time, if that is what you wish. Conversely, if a tank becomes empty, all outgoing pumps will automatically be turned off. This is because an empty tank can no longer transfer fuel. Again, you will have to turn these pumps on again if that is what you wish to do. Any questions?

Your overall goal is to keep the fuel level in Tanks A and B at 2500 units each. You may use any strategy that you wish to do so. If the fuel level in these tanks should go outside the shaded region, however, please return the fuel level back to the target level as soon as possible.

Scheduling Window

The scheduling window is found in the upper right corner of the display. The purpose of the scheduling window is to present the start and duration of the manual tracking task and the communication task. The scheduling window requires no response on your part but is designed to provide you with information about the scheduling of tasks in the near future. The two indicators are identified by "T" for the tracking task, and "C" for the communication task. The scheduling window allows you to "look ahead" from the present (0 minutes) to 8 minutes into the future. The bold lines, or bars, indicate times during which these two tasks, tracking and communication, will be operating. The thin lines indicate times at which either tracking or communication are not required.

Workload Rating Scale

One of our interests today is in assessing your perceptions of your own performance with the task battery in terms of difficulty level and workload. Now, we would like to describe the technique that will be used to examine your experiences. Our objective is to capture your perceived "workload" level. The concept of workload is hard to define specifically and is composed of many different aspects. Workload may refer, in part, to the physical demands of the task, the time pressure involved, your expended effort, or your resulting stress or frustration levels.

We hope to understand workload on this battery by asking individuals who perform the tasks to describe the various feelings and perceptions that they experienced while operating the battery. Since many factors may be involved, we would like you to tell us about several individual factors rather than one overall workload score. The set of six rating scales that you now see before you was developed at the NASA Ames Research

Center and has been used in a wide variety of tasks.

As you can see, there are six scales on which you will be asked to provide a rating score: mental demand, physical demand, temporal demand, performance, effort, and frustration. Let me take a minute to explain each of these scales. The first, mental demand, refers to the level of mental activity like thinking, deciding and looking that was required by the task. You will rate this scale from low (on the left side) to high (on the right side). The second scale, physical demand, involves the amount of physical activity required of you, such as controlling or activating. Temporal demand refers to the time pressure that you experienced during the task. In other words, was the pace slow and leisurely or rapid and frantic?

The fourth scale involves your perceptions about your own performance level. Your rating here should reflect your satisfaction with your performance in accomplishing the goals of the task. Notice that this scale ranges from good (on the left side) to poor (on the right side). All of the other scales range from low to high. The fifth scale, effort, inquires as to how hard you had to work (both mentally and physically) in order to achieve your level of performance. Finally, the sixth scale, frustration level, is an index of how secure and relaxed (low frustration) versus stressed and discouraged (high frustration) you felt during the task. Do you have any questions?

Now, I will explain the method you will use to rate your experiences with these six scales. When this screen first appears, the pointer on the first scale will be illuminated in yellow. You must select the score that best suits your perceptions about that scale from low to high with either the mouse (left to right movement) or the right and left arrow keys. After you have decided upon your score, use either the left mouse button key, the space bar, or the down arrow key to move to the next scale. You will notice that the pointer from the first scale will turn to a gray color, while the pointer on the active second scale is now yellow. The yellow pointer always indicates the scale which is active or is available for change.

After you have entered the sixth score, the instructions at the bottom portion of the screen will change. You now have the option of returning to the task battery by pressing the right mouse button, or by using the ESCape or Return keys. However, if you wish to change one of your scores, you may do so by pressing the left mouse button, the down arrow key or the space bar. Continue to press that key until the pointer for the scale that you wish to change is yellow. At that point, you may use the mouse or the left and right arrow keys to adjust your rating. When you are finished, press the right mouse button or use the ESCape or Return keys to return to the battery. Please practice this procedure, using your experience on the practice section of this task as the basis for your ratings. Feel free to ask any questions that you would like.

Now that you are familiar with both the task battery and the workload rating system, let me explain the presentation of the rating system. Several times during your performance with the battery, the computer screen will change from the battery to the rating scale. The first time that this happens, I would like you to rate your experiences with the task battery to that point. The second time the rating scale appears, you will rate your experiences with the task since the last rating process and so on. Any questions?

While you are rating your perceptions, the task battery will be paused and time frozen until you finish the rating process. In other words, you will not be missing anything while you complete the rating scale. Once you finish and return to the battery, you will resume where you left off in terms of elapsed time. Therefore, there is no need to rush through the rating scale; however, after 45 seconds, the rating scale screen will automati-

cally end and the task battery screen will reappear. Please give your responses thoughtful consideration, but do not spend too much time deliberating over them. Your first responses will probably accurately reflect your feelings and experiences.

MAT DATA FILES

Each time the Multi-Attribute Task Battery is run, five data files are created. Separate data files exist for each of the four task components: monitoring, tracking, communications and resource management. A fifth data file contains responses to the NASA Task Load Index Rating Scales. If the AFTERMAT and APPLYWT programs are run, these data files are generated as well. Data filenames are automatically assigned and all begin with the letters "MD" followed by the month (2 numerical digits), date (two digits) and hour (two digits) on which the run was begun. The one exception to this is the AFTERMAT weighting data file, which begins with WT. The filename extension consists of the run start time in minutes (two digits) and one of the following letters: M, T, C, R, or X. These letters identify the type of data contained in the file: Monitoring (M), Tracking (T), Communications (C), Resource management (R) or NASA TLX ratings (X). An example of the data file names which are generated from one run of the battery are listed below.

Data files created by the task battery:

MD101413.25M (Monitoring)
MD101413.25T (Tracking)
MD101413.25C (Communications)
MD101413.25R (Resource Management)
MD101413.25X (Subjective Ratings)

Data files created by support programs:

WT101414.10X (AFTERMAT program subscale weightings)
MD101413.25W (APPLYWT program adds the mean and weighted average to the Subjective Ratings file, creating a new file ending in "W")

The data filenames illustrated above would have been created by a run which began October 14, at 1:25 pm.

The following pages contain examples of each of the types of data files with annotation. The first line of each data file is a header containing the date (month-day-year), beginning time (hour:minute: seconds) and the full name of the data file (i.e., MD101413.25M).

```
MONITORING DATA FILE

10-14-1991 13:25:29 MD101413.25M    - header identification line

      Elapsed time at beginning of monitoring event (seconds)
      |
      |  Event Code (1 to 4 = scales, 5 = Green light, 6 = Red light)
      |  |    Negative event codes indicate a change in task Automation /
      |  |      Manual mix (range -1 to -6 reflecting levels 1 to 6)
      |  |
      |  |       Response Time (seconds)
      |  |       |
   57.01  6    1.43
   83.15  2  -20.10   * A negative sign here indicates that the
   81.01  3    3.68     described event timed out before the
  102.05  5    1.97     subject responded (i.e., a miss).
  152.03  5    1.43     The 4 scales time out after 20 seconds;
  163.01  3    5.50     the lights time out after 15 seconds.
   81.01  3    3.68     (These values can be changed in the
  102.05  5    1.97     Setup program)
  152.03  5    1.43
  163.01  3    5.50
  178.01  5    9.39
  194.05  1    0.00   * A response time of 0.00 indicates a
  218.21  3    0.00     false alarm on the subject's part.
  224.15  2  -20.11     The function key that was pressed
  248.04  6    1.43     is recorded in the second column.
  288.90  3    0.00
  336.08  2    0.00
  339.05 -3    0.00   * Negative event codes show changes in
  354.81  3    0.00     task automation / manual mix.
  367.06  1  -20.05
  360.04 -4    0.00
  362.01  6    5.16
  396.06  2  -20.05
  423.03  3    0.00
  425.01  2    3.90
```

TRACKING DATA FILE

10-14-1991 13:25:29 MD101413.25T - header identification line

Seconds into task (Start Time of Data Recording Interval)
| Sum of Squares (SS)
| | Screen updates in RMS Interval (N)
| | | Interval will vary depending on Setup selected
| | | data recording interval (10 * seconds)
| | |
| | | RMS = SQRT(SS/N)
| | | |
| | | |

240.07	51279.30	150	18.49	RMS is deviation from center
255.12	234749.14	150	39.56	of tracking target in pixel units.
270.23	136623.52	150	30.18	(Vertical pixels converted to
285.33	257187.50	150	41.41	Horizontal pixel units before
300.44	22690.17	150	12.30	computing resultant total
315.54	56336.78	150	19.38	deviation)
330.59	121081.50	150	28.41	
345.70	111887.74	150	27.31	
360.80	119243.27	150	28.19	
375.91	196448.38	150	36.19	
390.95	143360.45	150	30.91	
391.70	0.04	0	0.00	Switch to highest difficulty level
406.00	155585.45	150	32.21	tracking
421.05	445533.78	151	54.32	
423.07	0.03	0	0.00	Switch to medium difficulty level
436.21	27608.42	150	13.57	tracking
451.32	97377.30	150	25.48	
460.54	0.01	0	0.00	Switch to lowest difficulty level
466.42	271713.59	149	42.70	tracking (These tracking level changes
.				can be made whether tracking task is
.				on "Manual" or off "Auto")
.				
783.07	306373.41	150	45.19	
798.17	92931.23	150	24.89	
813.28	117154.58	150	27.95	
828.33	139201.88	104	36.59	*Switch to auto at about 10 sec.
840.03	0.00	0	0.00	*Switch from manual to auto.
4800.10	62347.64	150	20.39	*Manual mode resumes.
4815.15	241718.64	150	40.14	
4830.20	30305.10	150	14.21	
.				
.				
.				
5373.41	28163.55	150	13.70	
5388.52	38071.82	102	19.32	
5400.05	0.00	0	0.00	*Switch to auto
5535.00	0.00	0	0.00	*End of run

```
                        COMMUNICATIONS DATA FILE

10-14-1991 13:25:29 MD101413.25C    - header identification line

      Time from beginning of task (in seconds)
      |
      |  Either Event Code (57, 58), Channel (1 to 4), or Enter (13)
      |  |  Event codes 59 and 60 are recodes (+2) of a signal that was
      |  |    in the script but due to task level changes was recoded to
      |  |       prevent it from being presented to the subject
      |  |
      |  |       Change of frequency (0.00 = no change)
      |  |       |
  139.01 57    0.00   * Event (own message) occurs at 139.01 sec.
  147.74  1  108.70   * Subject changes frequency of Channel 1
  147.96  1  108.90     to 108.70, etc. and stops responding at
  148.24  1  109.10     148.79 seconds into the task when Channel
  148.35  1  109.30     1 is at Frequency 109.70.
  148.57  1  109.50
  148.79  1  109.70
  150.35 13    0.00   * "Enter" key pressed by the subject
  211.02 58    0.00   * Other message is presented - No response.
  246.01 57    0.00   * Event (own message) occurs at 246.01 sec.
  254.57  2    0.00   * Subject first changes channel to 2 and
  254.79  3    0.00     then to 3 (Frequency changes = 0.00).
  258.25  3  119.90   * Once channel is set, subject begins to
  258.58  3  120.10     change frequency.
  258.80  3  120.30
  258.97  3  120.50
  259.13  3  120.70
  259.35  3  120.90
  259.46  3  121.10
  263.20 13    0.00
  359.04 58    0.00    *Other message is presented - No response.
  438.03 58    0.00
  513.00 57    0.00    *Event (own message) occurs.
  523.22  2    0.00    *Subject begins to change channels and
  523.33  1    0.00     ends at 525.69 seconds.
  523.49  4    0.00
  524.37  3    0.00
  524.59  2    0.00
  524.81  1    0.00
  525.69  2    0.00
  526.46  2  110.50    *Subject begins to change frequency of
  526.68  2  110.70     Channel 2.
  526.79  2  110.90
  526.90  2  111.10
  527.01  2  111.30
  527.17  2  111.50
  527.28  2  111.70
  528.04 13    0.00
```

RESOURCE MANAGEMENT DATA FILE

10-14-1991 13:25:29 MD101413.25R - header identification line

Time into task from beginning of run (in seconds)
| Pump Activity or Event Codes
| | Amount of fuel present in Tank A
| | | Amount of fuel present in Tank B
| | | | Amount of fuel present in Tank C
| | | | | Amount of fuel present in Tank D
| | | | | |

24.60	1	2158	2170	1005	1010	*Pump 1 turned on by subject
30.04	0	2158	2089	924	1010	*Tank status is automatically updated every 30 seconds, noted by a "0" in Column 2.
30.26	3	2158	2089	924	1010	
43.94	2	2158	2089	762	848	
44.32	4	2178	2089	735	821	
60.03	0	2338	2249	519	605	
90.02	0	2638	2549	114	200	
100.01	10	2738	2649	0	65	*Multiples of 10 in Column 2 show that a pump was turned off by system (10=Pump 1, 20=Pump 2, etc.) and occurs when a tank becomes either empty or full.
106.00	30	2717	2709	0	0	
120.01	0	2668	2660	0	0	
150.00	0	2563	2555	0	0	
180.04	0	2458	2450	0	0	
191.03	85	2423	2415	0	0	*Codes 81, 82, etc. in Column 2 represent pump failures (See Table 1).
203.66	3	2381	2373	0	0	
204.04	30	2374	2393	0	0	
204.48	-4	2374	2393	0	0	*Neg numbers in Column 2 occur whenever the S turns a pump off (-1 = Pump 1 off, -2 = Pump 2 off, etc.)
207.01	4	2367	2366	0	0	
208.44	3	2360	2359	0	0	
210.03	30	2353	2379	0	0	
210.03	0	2353	2379	0	0	
210.03	8	2353	2379	0	0	
227.39	6	2409	2219	0	0	
235.19	-8	2437	2139	0	80	
240.02	0	2416	2118	0	140	
270.01	0	2311	2013	0	440	
270.01	86	2311	2013	0	440	
276.54	-4	2290	1992	0	440	
277.81	4	2290	1992	0	440	
278.03	3	2283	1985	0	440	
300.00	0	2206	2205	0	143	
311.04	95	2171	2305	0	8	*Event Codes 91, 92, etc. in Column 2 represent the repair of pump failures (See Table 1)
312.03	30	2164	2325	0	0	
324.71	5	2122	2283	0	0	
330.04	0	2101	2262	60	0	
360.03	0	1996	2157	360	0	
362.04	78	1990	2151	366	6	*Code 78 indicates switch to "AUTO" mode pumps switch by themselves in this mode
376.12	3	1940	2101	520	24	
378.05	79	1933	2121	540	30	*Code 79 indicates switch to "MANUAL"
380.02	1	1926	2114	560	36	
390.02	0	2026	2079	525	45	
390.02	96	2026	2079	525	54	

WORKLOAD RATING SCALE DATA FILES

Filename: **MDmmddhh.nnX** Where "X" indicates TLX Rating Data File, mm is month, dd is day, hh is hour, and nn is minute of beginning of data run.

```
10-14-1991 13:25:29 MD101413.25X    - header identification line

       Elapsed time (seconds) at beginning of Rating period
       |
       |       Duration of Rating screen presentation (seconds)
       |       |
       |       |   Mental Demand rating        (0 to 100)
       |       |   |   Physical Demand rating    (0 to 100)
       |       |   |   |   Temporal Demand rating  (0 to 100)
       |       |   |   |   |   Performance rating  (0 to 100)
       |       |   |   |   |   |   Effort rating   (0 to 100)
       |       |   |   |   |   |   |   Frustration rating (0 to 100)
       |       |   |   |   |   |   |   |
       |       |   |   |   |   |   |   |
    7.79    7.53  40  74  24  76  31  74
   29.38    4.45  14  81  26  74  25  72
    .
    .
    .
    .
 3548.12    9.37  30  76  24  74  30  74
```

APPLYWT GENERATED RATING DATA FILES

Filename: **MDmmddhh.nnW** Where "W" suffix indicates Weighted Rating Data File

APPLYWT applies the weights established during AFTERMAT to the ratings given by the subject that are contained in the rating data file or files. The data file created by APPLWT is a new data file with a "W" ending.

```
Example showing two new data                Mean of Subscales
columns created after AFTERMAT                |
and APPLYWT:                                  |  Weighted Rating Score
                                              |       |
                                              |       |
    header identification line                |       |
              |                               |       |
10-14-1991 13:25:29 MD101413.25X              |       |
    7.79    7.53  40  74  24  76  31  74  53.167  55.250
   29.38    4.45  14  81  26  74  25  72  48.667  46.430
    .
    .
    .
    .
 3548.12    9.37  30  76  24  74  30  74  51.333  52.150
```

AFTERMAT DATA FILE

```
10-14-1991 14:10:20 WT101414.10X RC   - header identification line
 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1        - subscale chosen of the pair
 3 1 2 3 4 2         - Total number of times each scale was chosen
 | | | | | |             as largest contributor to workload
 | | | | | |
 | | | | | Frustration
 | | | | Effort
 | | | Performance
 | | Temporal Demand
 | Physical Demand
 Mental Demand
```

SUMMARY

The Multi-Attribute Task (MAT) Battery provides a benchmark set of tasks for use in a wide range of laboratory studies of operator performance and workload. The battery incorporates tasks analogous to activities that aircraft crewmembers perform in flight, while providing a high degree of experimenter control, performance data on each subtask, and freedom to use non-pilot test subjects. Features not found in existing computer-based tasks include an auditory communications task (to simulate Air Traffic Control communication), a resource management task permitting many avenues or strategies of maintaining target performance, a scheduling window which gives the operator information about future task demands, and the option of manual or automated control of tasks. Performance data are generated for each subtask. In addition, the task battery may be paused and onscreen workload rating scales presented to the subject. The MAT Battery requires a desktop computer (80286/386/486 processor) with color graphics (at least 640 x 350 pixel). The communications task requires a serial link to a second desktop computer with a voice synthesizer or digitizer card.

The preceding pages include sections describing various aspects of the MAT Battery. A section entitled "Task Battery Description" described each task. Details concerning setup, operation, and control of the MAT Battery are found in the sections "Running the Task Battery" and "Control of Task Events." The section entitled "Instructions to Subjects" provides descriptions of how each task may be introduced to test subjects. Examples of performance and rating data files generated by the task battery are presented in the section entitled "MAT Data Files." Research conducted using the task battery may be found in Arnegard (1991) and Arnegard and Comstock (1991).

REFERENCES

Arnegard, R. J. (1991) *Operator Strategies Under Varying Conditions of Workload.* National Aeronautics and Space Administration, Langley Research Center, NASA CR-4385.

Arnegard, R. J., and Comstock, J. R., Jr. (1991) Multi-Attribute Task Battery: Applications in Pilot Workload and Strategic Behavior Research. *Proceedings of the Sixth International Symposium on Aviation Psychology,* Columbus, Ohio, April 29 - May 2, 1118-1123.

Hart S. G., and Staveland, L. E. (1988) Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In P. A. Hancock and N. Meshkati (Eds.) *Human Mental Workload.* Amsterdam: North Holland Press.

MATSET4 Program Listing

```
'Comstock:  Multi-Attribute Task Battery Setup Program  MATSET4 (Ver 4.0)
' Permits setting or modifying parameters used by the MAT.  Reads old
' parameters from file MATFILE4.NAM and/or generates new file.  Reads Alpha-
' numeric script and translates script to all numeric MATNUM.SCR for MAT use.
' Sets up Serial port parameters for MAT connection to Voice task computer.
' Developed using Microsoft QuickBasic 4.5.
'
' J. R. Comstock, Jr.  NASA Langley Research Center  804-864-6643
' 03-27-91 Version 3.5
' 09-30-91 Version 4.0 w/ res mgmt auto/man & task level control
'
DEFINT I-N
DECLARE SUB keepit (ltype)
DECLARE SUB mainmenu ()
DECLARE SUB mainselect (lsel)
DECLARE SUB menubox (ix1, iy1, ix2, iy2, itxtcol, ibckgnd)
DECLARE SUB parambox ()
DECLARE SUB readscript ()
DECLARE SUB ScriptGen ()
DECLARE SUB taskparam ()
DECLARE SUB trackgain ()
DECLARE FUNCTION Parse$ (i$, j%)

DIM SHARED ipmpnew(8), ipmpold(8), ipmpdef(8)

COMMON SHARED filscr$, newscr$, oldscr$, mat$, matvj$, matvm$
COMMON SHARED ltrkgain, lprobscr, ltrkgainold, lerrsect
COMMON SHARED tmgrnnew, tmgrnold, tmgrndef
COMMON SHARED tmrednew, tmredold, tmreddef
COMMON SHARED tmdianew, tmdiaold, tmdiadef
COMMON SHARED trkincnew, trkincold, trkincdef
COMMON SHARED idropanew, idropaold, idropadef
COMMON SHARED idropbnew, idropbold, idropbdef

ON ERROR GOTO errorhandler

start:  ' assign default values
  CLS
  mat$ = "MATFILE4.NAM"
  matvj$ = "MATV40J"
  matvm$ = "MATV40M"

  tmgrndef = 15!
  tmreddef = 15!
  tmdiadef = 20!
  trkincdef = 5!

  idropadef = 800
  idropbdef = 800
  ipmpdef(1) = 800
  ipmpdef(2) = 600
  ipmpdef(3) = 800
  ipmpdef(4) = 600
  ipmpdef(5) = 600
  ipmpdef(6) = 600
  ipmpdef(7) = 400
  ipmpdef(8) = 400

' read existing matfiles file

  lerrsect = 1  'index this section for errorhandler
OPEN "I", #1, mat$
  LINE INPUT #1, oldscr$
  FOR i = 1 TO 5
    LINE INPUT #1, du1$
  NEXT i
  INPUT #1, ltrkgainold, tmgrnold, tmredold, tmdiaold, trkincold
  INPUT #1, idropaold, idropbold
  FOR i = 1 TO 8
    INPUT #1, ipmpold(i)
  NEXT i
  LINE INPUT #1, matvj$
```

```
  LINE INPUT #1, matvm$
CLOSE #1

begin:
  tmgrnnew = tmgrnold
  tmrednew = tmredold
  tmdianew = tmdiaold
  trkincnew = trkincold

  idropanew = idropaold
  idropbnew = idropbold

  FOR i = 1 TO 8
    ipmpnew(i) = ipmpold(i)
  NEXT i

  filscr$ = UCASE$(oldscr$): ltrkgain = ltrkgainold
  lsel = 1
  mainmenu

menu1:
  mainselect lsel

  SELECT CASE lsel
    CASE 1 'Initialize Serial Port
      lerrsect = 2
      OPEN "COM1:19200,N,8,1" FOR RANDOM AS #5 LEN = 256: CLOSE #5
      COLOR 15, 3: LOCATE 5, 4: PRINT CHR$(251);
      lsel = 2

    CASE 2 'Select Script File
      lerrsect = 3
      readscript
      COLOR 15, 3: LOCATE 6, 4: PRINT CHR$(251);
      lsel = 4

    CASE 4 'Begin Task (Normal)
      ltype = 1
      COLOR 15, 3: LOCATE 8, 4: PRINT CHR$(251);
      keepit ltype
      IF ltype = 27 THEN GOTO begin
      RUN matvj$
      END

    CASE 5 'Begin Task (Alternate)
      ltype = 1
      COLOR 15, 3: LOCATE 9, 4: PRINT CHR$(251);
      keepit ltype
      IF ltype = 27 THEN GOTO begin
      RUN matvm$
      END

    CASE 7 'Test Run Normal
      ltype = 2
      COLOR 15, 3: LOCATE 11, 4: PRINT CHR$(251);
      keepit ltype
      IF ltype = 27 THEN GOTO begin
      RUN matvj$
      END

    CASE 8 'Test Run Alternate
      ltype = 2
      COLOR 15, 3: LOCATE 12, 4: PRINT CHR$(251);
      keepit ltype
      IF ltype = 27 THEN GOTO begin
      RUN matvm$
      END

    CASE 11 'Change Tracking Gain
      trackgain
      SCREEN 0, , 0
      COLOR 0, 3
```

```
      LOCATE 15, 28: PRINT USING "(##)"; ltrkgain;
      COLOR 15, 3: LOCATE 15, 4: PRINT CHR$(251);

    CASE 12 'Change Task Parameters
      taskparam
      SCREEN 0, , 0
      COLOR 15, 3: LOCATE 16, 4: PRINT CHR$(251);

    CASE 14 'Exit, Saving Setup
      ltype = 1
      COLOR 15, 3: LOCATE 18, 4: PRINT CHR$(251);
      keepit ltype
      IF ltype = 27 THEN GOTO begin
      SCREEN 0, , 0: CLS : END

    CASE 15 'Exit, Cancel Setup
      SCREEN 0, , 0: CLS : END

    CASE ELSE
  END SELECT
GOTO menu1
END

' E R R O R H A N D L E R ------------------------------------------

errorhandler:
lerrnum = ERR
COLOR 7, 0: CLS
PRINT "Multi-Attribute Task Setup Program": PRINT
PRINT "ERROR NUMBER: "; lerrnum: PRINT

SOUND 1000, 2
COLOR 15, 4

IF lerrsect = 1 THEN
  CLOSE #1
  PRINT " Creating DEFAULT "; mat$
  OPEN "O", #1, mat$
    PRINT #1, "SCR24.DTB"
    PRINT #1, "MDTEST00.00M"
    PRINT #1, "MDTEST00.00T"
    PRINT #1, "MDTEST00.00R"
    PRINT #1, "MDTEST00.00C"
    PRINT #1, "MDTEST00.00X"
    PRINT #1, "34  15.0  15.0  20.0   5.0"
    PRINT #1, "  800  800  800  600  800  600  600  600  400  400"
    PRINT #1, matvj$
    PRINT #1, matvm$
  CLOSE #1
END IF

IF lerrsect = 2 THEN
  CLOSE #5
  PRINT " PROBLEM INITIALIZING SERIAL PORT "
END IF

IF lerrsect = 3 THEN
  CLOSE #3
  PRINT " PROBLEM WITH SCRIPT FILE or Directory (*.DTB) "
END IF

IF lerrsect = 4 THEN
  PRINT " PROBLEM RUNNING MAT TASK Executable file "
END IF

IF lerrsect = 5 THEN
  PRINT " PROBLEM Writing Setup File "
END IF

  COLOR 7, 0
  PRINT : PRINT " Enter RETURN to Restart SETUP, ESC to Quit"
  DO
```

```
    a$ = INKEY$
    IF a$ = CHR$(27) THEN END
    IF a$ = CHR$(13) THEN EXIT DO
  LOOP
  RESUME start

END
```

```
'
' K E E P I T  Sub ------------------------------------------------
'   Comstock
SUB keepit (ltype) STATIC

lerrsect = 5
IF ltype = 1 THEN
  d$ = DATE$: T$ = TIME$
  fildat$ = "MD" + LEFT$(d$, 2) + MID$(d$, 4, 2) + LEFT$(T$, 2) + "." + MID$(T$, 4, 2)
  fildat1$ = fildat$ + "M"
  fildat2$ = fildat$ + "T"
  fildat3$ = fildat$ + "R"
  fildat4$ = fildat$ + "C"
  fildat5$ = fildat$ + "X"
ELSE
  fildat1$ = "MDTEST00.00M"
  fildat2$ = "MDTEST00.00T"
  fildat3$ = "MDTEST00.00R"
  fildat4$ = "MDTEST00.00C"
  fildat5$ = "MDTEST00.00X"
END IF

' display summary box

menubox 40, 5, 66, 18, 0, 7

  COLOR 0, 7
  LOCATE 6, 43: PRINT "SUMMARY INFORMATION:";
  LOCATE 8, 43: PRINT "Script: "; filscr$;
  LOCATE 10, 43: PRINT "DATA filenames:";
  LOCATE 12, 45: PRINT fildat1$;
  LOCATE 13, 45: PRINT fildat2$;
  LOCATE 14, 45: PRINT fildat3$;
  LOCATE 15, 45: PRINT fildat4$;
  LOCATE 16, 45: PRINT fildat5$;

  COLOR 0, 7: LOCATE 21, 40: PRINT SPC(5); "Press RETURN to Continue,  ";
  LOCATE 22, 40: PRINT SPC(5); "ESC to RESTART Setup"; SPC(7);
  DO
    a$ = INKEY$
    IF a$ = CHR$(27) THEN ltype = 27: EXIT SUB
    IF a$ = CHR$(13) THEN EXIT DO
  LOOP

' save new MATFILE4.NAM

OPEN "O", #2, mat$
  PRINT #2, filscr$
  PRINT #2, fildat1$
  PRINT #2, fildat2$
  PRINT #2, fildat3$
  PRINT #2, fildat4$
  PRINT #2, fildat5$
  PRINT #2, USING "##"; ltrkgain;
  PRINT #2, USING "####.#"; tmgrnnew; tmrednew; tmdianew; trkincnew
  PRINT #2, USING "#####"; idropanew; idropbnew;
  FOR i = 1 TO 8
    PRINT #2, USING "#####"; ipmpnew(i);
  NEXT i
  PRINT #2,
  PRINT #2, matvj$
  PRINT #2, matvm$
CLOSE #2

  SCREEN 0, , 0
  lerrsect = 4
END SUB
```

```
'
'   M A I N M E N U  Sub ------------------------------------------
'     Comstock
SUB mainmenu
  COLOR 15, 1: CLS
  LOCATE 2, 20: PRINT "Multi-Attribute Task Battery Setup Program"
  PRINT STRING$(80, 205);
  PRINT : COLOR 7, 1
  LOCATE 24, 6: PRINT "J. R. Comstock, Jr.,  NASA Langley Research Center,  ";
  PRINT "Hampton, Virginia";
  LOCATE 1, 65: PRINT "(SETUP Ver 4.0)"; : LOCATE 5, 1
  menubox 2, 4, 35, 21, 0, 3

  COLOR 0, 3
  LOCATE 5, 6: PRINT "Initialize Serial Port";
  LOCATE 6, 6: PRINT "Select Script File";
  LOCATE 8, 6: PRINT "Begin Task (Normal Version)";
  LOCATE 9, 6: PRINT "Begin Task (Alternate Ver.)";
  LOCATE 11, 6: PRINT "Test Run (Normal Version)";
  LOCATE 12, 6: PRINT "Test Run (Alternate Ver.)";
  LOCATE 15, 6: PRINT "Change Tracking Gain";
  PRINT USING "  (##)"; ltrkgain;
  LOCATE 16, 6: PRINT "Change Task Parameters";
  LOCATE 18, 6: PRINT "Exit, Saving Setup";
  LOCATE 19, 6: PRINT "Exit, Cancel Setup";

  LOCATE 7, 2: PRINT CHR$(195); : PRINT STRING$(32, 196); : PRINT CHR$(180);
  LOCATE 14, 2: PRINT CHR$(195); : PRINT STRING$(32, 196); : PRINT CHR$(180);
  LOCATE 17, 2: PRINT CHR$(195); : PRINT STRING$(32, 196); : PRINT CHR$(180);
  COLOR 7, 1
  PCOPY 1, 0
  SCREEN 0, , 0
END SUB
```

```
'
' M A I N S E L E C T  Sub ------------------------------------
'   Comstock
SUB mainselect (item) STATIC
PCOPY 0, 1
SCREEN 0, , 1

 itemold = item
 ix1 = 2: ix2 = 35
 iy1 = 4: iy2 = 21
 itxtcol = 0: ibckgnd = 3

newitem:

 FOR icol = ix1 + 1 TO ix2 - 1
   ipl = SCREEN(iy1 + itemold, icol, 0)
   COLOR itxtcol, ibckgnd
   LOCATE iy1 + itemold, icol: PRINT CHR$(ipl);
   ipl = SCREEN(iy1 + item, icol, 0)
   COLOR 15, 0
   LOCATE iy1 + item, icol: PRINT CHR$(ipl);
 NEXT icol
 itemold = item

COLOR 7, 1
FOR i = 5 TO 20
  LOCATE i, 42: PRINT SPC(35);
NEXT i

SELECT CASE item
  CASE 1
    LOCATE 5, 43: PRINT "Sets COM1 serial port parameters";
    LOCATE 6, 43: PRINT "for link to voice computer.";
    LOCATE 7, 43: PRINT "If second computer is not used,";
    LOCATE 8, 43: PRINT "then skip this selection.";
  CASE 2
    LOCATE 6, 43: PRINT "Displays the Directory of ASCII";
    LOCATE 7, 43: PRINT "script files (with .DTB extension)";
    LOCATE 8, 43: PRINT "and permits selection of desired";
    LOCATE 9, 43: PRINT "script.";
  CASE 4, 5
    LOCATE 8, 43: PRINT "Begin task with DATA filenames";
    LOCATE 9, 43: PRINT "based on present date and time.";
    LOCATE 10, 43: PRINT "(Normal Task Start)";
  CASE 7, 8
    LOCATE 11, 43: PRINT "Begin test or demo run with DATA";
    LOCATE 12, 43: PRINT "filenames assigned MDTEST00.00*,";
    LOCATE 13, 43: PRINT "Overwrite existing MDTEST files.";
  CASE 11
    LOCATE 15, 43: PRINT "Permits adjusting Tracking task"
    LOCATE 16, 43: PRINT "control sensitivity.  Values";
    LOCATE 17, 43: PRINT "range from 1 to 48, higher values";
    LOCATE 18, 43: PRINT "mean higher gain.  Typical values";
    LOCATE 19, 43: PRINT "Joystick: 34,  Mouse: 25";
  CASE 12
    LOCATE 16, 43: PRINT "Permits modifying task timeouts,";
    LOCATE 17, 43: PRINT "Tracking RMS data interval, and";
    LOCATE 18, 43: PRINT "Resource Mgmt task flow rates.";
  CASE 14
    LOCATE 18, 43: PRINT "Exit Setup, Save present values";
    LOCATE 19, 43: PRINT "for use in next run of Setup.";
  CASE 15
    LOCATE 19, 43: PRINT "Exit Setup, Ignore any values";
    LOCATE 20, 43: PRINT "changed during this setup.";
  CASE ELSE
END SELECT

DO
  a$ = INKEY$
  IF a$ <> "" THEN
    keyasc = ASC(a$)
    IF keyasc = 0 THEN keyfunct = ASC(RIGHT$(a$, 1)) ELSE keyfunct = 0
```

```
    IF keyfunct = 72 THEN
      item = item - 1
      SELECT CASE item
        CASE IS < 1
          item = 15
        CASE 3, 6, 13
          item = item - 1
        CASE 10
          item = item - 2
        CASE ELSE
      END SELECT
      GOTO newitem
    END IF
    IF keyfunct = 80 OR keyasc = 32 THEN
      item = item + 1
      SELECT CASE item
        CASE IS > 15
          item = 1
        CASE 3, 6, 13
          item = item + 1
        CASE 9
          item = item + 2
        CASE ELSE
      END SELECT
      GOTO newitem
    END IF
    IF keyasc = 13 THEN SCREEN 0, , 0: EXIT SUB
    IF keyasc = 27 THEN CLS : SCREEN 0, , 0: CLS : END
  END IF
LOOP

END SUB
```

```
'
' M E N U B O X  Sub ---------------------------------------------
'    Comstock
SUB menubox (ix1, iy1, ix2, iy2, itxtcol, ibckgnd) STATIC

PCOPY 0, 1
SCREEN 0, , 1
COLOR itxtcol, ibckgnd
iwidthm1 = (ix2 - ix1) - 1

  FOR irow = iy1 + 1 TO iy2 - 1            'fill box with background
    LOCATE irow, ix1: PRINT CHR$(179); SPC(iwidthm1); CHR$(179);
  NEXT irow

  LOCATE iy1, ix1 + 1: PRINT STRING$(iwidthm1, 196);
  LOCATE iy2, ix1 + 1: PRINT STRING$(iwidthm1, 196);
  LOCATE iy1, ix1: PRINT CHR$(218);
  LOCATE iy1, ix2: PRINT CHR$(191);
  LOCATE iy2, ix1: PRINT CHR$(192);
  LOCATE iy2, ix2: PRINT CHR$(217);

  COLOR 8, 0
  FOR irow = iy1 + 1 TO iy2 + 1
    ip1 = SCREEN(irow, ix2 + 1, 0)
    ip2 = SCREEN(irow, ix2 + 2, 0)
    LOCATE irow, ix2 + 1: PRINT CHR$(ip1); CHR$(ip2);
  NEXT irow

  FOR icol = ix1 + 2 TO ix2 + 1
    ip1 = SCREEN(iy2 + 1, icol, 0)
    LOCATE iy2 + 1, icol: PRINT CHR$(ip1);
  NEXT icol

END SUB
```

```
'
' P A R A M B O X Sub ----------------------------------------
'   Comstock
SUB parambox STATIC

COLOR 0, 7
  iy1 = 19: iy2 = 24: ix1 = 43: ix2 = 78
  iwidthm1 = (ix2 - ix1) - 1
  FOR irow = iy1 + 1 TO iy2 - 1
    LOCATE irow, ix1: PRINT CHR$(179); SPC(iwidthm1); CHR$(179);
  NEXT irow

  LOCATE iy1, ix1 + 1: PRINT STRING$(iwidthm1, 196);
  LOCATE iy2, ix1 + 1: PRINT STRING$(iwidthm1, 196);
  LOCATE iy1, ix1: PRINT CHR$(218);
  LOCATE iy1, ix2: PRINT CHR$(191);
  LOCATE iy2, ix1: PRINT CHR$(192);
  LOCATE iy2, ix2: PRINT CHR$(217);

END SUB
```

```
'
'  P A R S E  Sub ------------------------------------------
'     R. L. Harris, Sr.
FUNCTION Parse$ (i$, j)

c$ = ""

FOR k = j TO LEN(i$)
    IF ASC(MID$(i$, k, 1)) > 32 THEN EXIT FOR
NEXT k

FOR l = k TO LEN(i$)
    IF ASC(MID$(i$, l, 1)) < 33 THEN EXIT FOR
    c$ = c$ + MID$(i$, l, 1)
NEXT l

j = l
Parse$ = c$

END FUNCTION
```

```
'
' R E A D S C R I P T  Sub ---------------------------------------
'   Comstock
SUB readscript
  SCREEN 0, , 1
  COLOR 7, 1: CLS

  lprobscr = 1
  PRINT "  Directory of Script Files:"
  PRINT : FILES "*.DTB": PRINT

' obtain script file name

  PRINT "  Enter name of Script File <RETURN for: "; filscr$; " > ";
  LINE INPUT newscr$

  IF LEN(newscr$) > 0 THEN filscr$ = UCASE$(newscr$)

  lprobscr = 2
  OPEN "I", #3, filscr$   'read the script file
  CALL ScriptGen
  CLOSE #3

  SCREEN 0, , 0
END SUB
```

```
'
'  S C R I P T G E N  Sub ---------------------------------------
'     R. L. Harris, Sr.
SUB ScriptGen
OPEN "MATNUM.SCR" FOR OUTPUT AS #4

N = 0
Told = 0

PRINT "    Reading Line Number";
VLine = CSRLIN
DO UNTIL EOF(3)
    LINE INPUT #3, In$
    N = N + 1
    LOCATE VLine, 25
    PRINT USING "####"; N
    i$ = UCASE$(In$)
    IF i$ = "" THEN i$ = " "
    i = ASC(MID$(i$, 1, 1))

' Check for tab, space, or number
    IF i = 9 OR i = 32 OR (i > 47 AND i < 58) THEN    'Not a comment line
        j = 1
        h = VAL(Parse$(i$, j))
        m = VAL(Parse$(i$, j))
        s = VAL(Parse$(i$, j))
        c$ = Parse$(i$, j)
        code = VAL(c$)

        IF code = 0 THEN

            SELECT CASE c$

              CASE "RATING", "RATINGS", "WORKLOAD", "TLX"
                code = 10

              CASE "SCALE"
                code = VAL(Parse$(i$, j)) * 10   'Codes 10, 20, 30, or 40
                IF Parse$(i$, j) = "UP" THEN
                    code = code + 1
                ELSE
                    code = code + 2
                END IF

              CASE "GREEN"
                code = 51

              CASE "COMM", "COMMUNICATION", "COMMUNICATIONS"
                c$ = Parse$(i$, j)
                IF c$ = "TASK" THEN c$ = Parse$(i$, j)

                SELECT CASE c$
                  CASE "BEGIN", "BEGINS", "START", "STARTS"
                    code = 55
                  CASE "END", "FINISH"
                    code = 56
                  CASE "OWN"
                    code = 57
                  CASE "OTHER"
                    code = 58
                  CASE ELSE
                    code = -9
                END SELECT

              CASE "RED"
                code = 61

              CASE "MANUAL"
                code = 71

              CASE "AUTO"
                c$ = Parse$(i$, j)
```

```
            IF c$ = "END" THEN
                code = 73
            ELSE
                code = 72
            END IF

          CASE "TRACK", "TRACKING"
            c$ = Parse$(i$, j)
            SELECT CASE c$
              CASE "LO", "LOW"
                code = 74
              CASE "MED", "MEDIUM"
                code = 75
              CASE "HI", "HIGH"
                code = 76
              CASE ELSE
                code = -9
            END SELECT

          CASE "FAIL"
            c$ = Parse$(i$, j)
            IF c$ = "PUMP" THEN c$ = Parse$(i$, j)

            code = VAL(c$)
            IF code > 0 AND code < 9 THEN
                code = code + 80
            ELSE
                code = -9
            END IF

          CASE "FIX"
            c$ = Parse$(i$, j)
            IF c$ = "PUMP" THEN c$ = Parse$(i$, j)

            code = VAL(c$)
            IF code > 0 AND code < 9 THEN
                code = code + 90
            ELSE
                code = -9
            END IF

          CASE "RESOURCE", "RES"
            c$ = Parse$(i$, j)
            SELECT CASE c$
              CASE "AUTO"
                code = 78
              CASE "MANUAL", "MAN"
                code = 79
              CASE ELSE
                code = -9
            END SELECT

          CASE "LEVEL", "LEV"
            c$ = Parse$(i$, j)
            code = VAL(c$)
            IF code > 0 AND code < 7 THEN
                code = code + 100
            ELSE
                code = -9
            END IF

          CASE "END"
            code = 99

          CASE ELSE
            code = -9

        END SELECT
    END IF
    IF code > 0 THEN
        T = h * 3600 + m * 60 + s
        IF T >= Told THEN
```

```
                PRINT #4, USING "###"; h; m; s;
                PRINT #4, USING "####"; code
                Told = T
            ELSE
                SOUND 1000, 3
                PRINT "Timing sequence ERROR in line"; N
                PRINT In$
                SLEEP 3
            END IF
        ELSEIF code < 0 THEN
            SOUND 1000, 3
            PRINT "Syntax ERROR in line"; N
            PRINT In$
            SLEEP 3
        END IF
    END IF
LOOP
CLOSE #4

END SUB
```

```
'
' T A S K P A R A M  Sub ---------------------------------------------
'   Comstock
SUB taskparam STATIC

param1:
SCREEN 0, , 1
COLOR 7, 1: CLS
LOCATE 2, 1: PRINT STRING$(80, 205);
LOCATE 2, 27: PRINT " Task Parameter Modification ";
LOCATE 4, 4: PRINT "F1 - Select All Timing Defaults";
LOCATE 6, 4: PRINT "F2 - Select All Flow Defaults";

LOCATE 8, 2
  PRINT STRING$(7, 196); " TASK TIMING  (default)  now "; STRING$(2, 196);
LOCATE 10, 4: PRINT USING "F3 - Timeout Green  (###.#) "; tmgrndef;
  PRINT USING "####.#"; tmgrnnew;
LOCATE 12, 4: PRINT USING "F4 - Timeout Red    (###.#) "; tmreddef;
  PRINT USING "####.#"; tmrednew;
LOCATE 14, 4: PRINT USING "F5 - Timeout Dials  (###.#) "; tmdiadef;
  PRINT USING "####.#"; tmdianew;
LOCATE 16, 4: PRINT USING "F6 - RMS Interval   (###.#) "; trkincdef;
  PRINT USING "####.#"; trkincnew;

LOCATE 18, 2: PRINT STRING$(7, 196); " EXECUTABLE PROGRAM "; STRING$(11, 196);
LOCATE 20, 4: PRINT "F7 - Normal Program: "; matvj$;
LOCATE 22, 4: PRINT "F8 - Alternate Pgrm: "; matvm$;

FOR j = 4 TO 24
  LOCATE j, 41: PRINT CHR$(179);
NEXT j

LOCATE 4, 43
  PRINT STRING$(5, 196); " RESOURCE MGMT RATES /min "; STRING$(5, 196);
LOCATE 5, 59: PRINT "(default)  now";
LOCATE 6, 45: PRINT USING "A - Tank A Drop  (####)"; idropadef;
  PRINT USING "#####"; idropanew;
LOCATE 8, 45: PRINT USING "B - Tank B Drop  (####)"; idropbdef;
  PRINT USING "#####"; idropbnew;

FOR i = 1 TO 8
  LOCATE 9 + i, 44: PRINT i; "- Pump"; i; "Rate ";
  PRINT USING " (####)"; ipmpdef(i);
  PRINT USING "#####"; ipmpnew(i);
NEXT i

parambox
COLOR 0, 7
LOCATE 20, 46: PRINT "Select changes by Function key,";
LOCATE 21, 46: PRINT "A, B, or Number key, or press";
LOCATE 23, 46: PRINT "F10 (or Esc) to Exit";

DO
  a$ = INKEY$
  IF a$ <> "" THEN
    keyasc = ASC(a$)
    IF keyasc = 0 THEN keyfunct = ASC(RIGHT$(a$, 1)) ELSE keyfunct = 0

    SELECT CASE keyasc

      CASE 27 'esc
        EXIT SUB

      CASE 49 TO 56 'numbers
        parambox
        ipu = keyasc - 48
        LOCATE 20, 46: PRINT "Enter Pump"; ipu; "Flow Rate";
        LOCATE 21, 46: PRINT "Range of 0 to 2000";
        LOCATE 22, 48: PRINT "--> ";

        LINE INPUT v$
        IF v$ = "" THEN EXIT DO
```

```
    v$ = LEFT$(v$, 4)
    iv = VAL(v$)
    IF iv >= 0 AND iv <= 2000 THEN
      ipmpnew(ipu) = iv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO

  CASE 65, 97 'A or a
    parambox
    LOCATE 20, 46: PRINT "Enter Tank A Drop Rate";
    LOCATE 21, 46: PRINT "Range of 0 to 2000";
    LOCATE 22, 48: PRINT "--> ";

    LINE INPUT v$
    IF v$ = "" THEN EXIT DO
    v$ = LEFT$(v$, 4)
    iv = VAL(v$)
    IF iv >= 0 AND iv <= 2000 THEN
      idropanew = iv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO

  CASE 66, 98 'B or b
    parambox
    LOCATE 20, 46: PRINT "Enter Tank B Drop Rate";
    LOCATE 21, 46: PRINT "Range of 0 to 2000";
    LOCATE 22, 48: PRINT "--> ";

    LINE INPUT v$
    IF v$ = "" THEN EXIT DO
    v$ = LEFT$(v$, 4)
    iv = VAL(v$)
    IF iv >= 0 AND iv <= 2000 THEN
      idropbnew = iv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO

  CASE ELSE
END SELECT

SELECT CASE keyfunct

  CASE 59 'f1
    tmgrnnew = tmgrndef
    tmrednew = tmreddef
    tmdianew = tmdiadef
    trkincnew = trkincdef
    EXIT DO

  CASE 60 'f2
    idropanew = idropadef
    idropbnew = idropbdef
    FOR i = 1 TO 8
      ipmpnew(i) = ipmpdef(i)
    NEXT i
    EXIT DO

  CASE 61 'f3
    parambox
    LOCATE 20, 46: PRINT "Enter Green Light Timeout";
```

```
    LOCATE 21, 46: PRINT "Range of 2.0 to 30.0 sec";
    LOCATE 22, 48: PRINT "--> ";

    LINE INPUT v$
    IF v$ = "" THEN EXIT DO
    v$ = LEFT$(v$, 4)
    tv = VAL(v$)
    IF tv >= 2! AND tv <= 30! THEN
      tmgrnnew = tv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO

  CASE 62 'f4
    parambox
    LOCATE 20, 46: PRINT "Enter Red Light Timeout";
    LOCATE 21, 46: PRINT "Range of 2.0 to 30.0 sec";
    LOCATE 22, 48: PRINT "--> ";

    LINE INPUT v$
    IF v$ = "" THEN EXIT DO
    v$ = LEFT$(v$, 4)
    tv = VAL(v$)
    IF tv >= 2! AND tv <= 30! THEN
      tmrednew = tv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO

  CASE 63 'f5
    parambox
    LOCATE 20, 46: PRINT "Enter Dial Signal Timeout";
    LOCATE 21, 46: PRINT "Range of 5.0 to 60.0 sec";
    LOCATE 22, 48: PRINT "--> ";

    LINE INPUT v$
    IF v$ = "" THEN EXIT DO
    v$ = LEFT$(v$, 4)
    tv = VAL(v$)
    IF tv >= 5! AND tv <= 60! THEN
      tmdianew = tv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO

  CASE 64 'f6
    parambox
    LOCATE 20, 46: PRINT "Enter RMS Calculation Interval";
    LOCATE 21, 46: PRINT "Range of 1.0 to 15.0 sec";
    LOCATE 22, 48: PRINT "--> ";

    LINE INPUT v$
    IF v$ = "" THEN EXIT DO
    v$ = LEFT$(v$, 4)
    tv = VAL(v$)
    IF tv >= 1! AND tv <= 15! THEN
      trkincnew = tv
    ELSE
      SOUND 1000, 2
      LOCATE 23, 46: PRINT "Re-Enter Value";
      SLEEP 1
    END IF
    EXIT DO
```

```
      CASE 65 'f7
        parambox
        LOCATE 20, 46: PRINT "Enter Normal Program Name";
        LOCATE 21, 46: PRINT "(without (.) and extension)";
        LOCATE 22, 48: PRINT "--> ";

        LINE INPUT v$
        IF v$ = "" THEN EXIT DO
        v$ = UCASE$(v$)
        matvj$ = LEFT$(v$, 8)
        EXIT DO

      CASE 66 'f8
        parambox
        LOCATE 20, 46: PRINT "Enter Alternate Program Name";
        LOCATE 21, 46: PRINT "(without (.) and extension)";
        LOCATE 22, 48: PRINT "--> ";

        LINE INPUT v$
        IF v$ = "" THEN EXIT DO
        v$ = UCASE$(v$)
        matvm$ = LEFT$(v$, 8)
        EXIT DO

      CASE 68 'f10 exit
        EXIT SUB

      CASE ELSE
    END SELECT
  END IF

LOOP
GOTO param1

END SUB
```

```
'
' T R A C K G A I N  Sub ----------------------------------------
'    Comstock:  Select tracking gain value
SUB trackgain

startgain:
  menubox 45, 15, 75, 20, 0, 7
  COLOR 0, 7
  LOCATE 16, 48: PRINT "Enter Tracking Gain";
  LOCATE 17, 48: PRINT "Range of 1 to 48";
  LOCATE 18, 48: PRINT "  --> ";

  LINE INPUT m$
  IF m$ = "" THEN EXIT SUB
  m$ = LEFT$(m$, 2)
  newtrack = VAL(m$)
  IF newtrack > 0 AND newtrack < 49 THEN
    ltrkgain = newtrack
  ELSE
    GOTO startgain
  END IF

END SUB
```

MATV40J Program Listing

```
' Comstock:  Multi-Attribute Task  MATV40J.BAS
' Version 4.0  09-30-91
'
' J. R. Comstock, Jr., Ph.D. (804) 864-6643; FTS 928-6643
' MS-152 NASA LaRC, Hampton, VA  23665
'
' Written using Microsoft QuickBASIC 4.5, using Screen mode 9 (EGA)
' Hardware Requirements: PC or compatible with EGA graphics;
'   80286/80386/80486 machine recommended.  Has been tested on
'   IBM PC AT, IBM PS-2, & Compaq 386/20
'
' Contains CALLs to Library routines found in SJ.QLB and/or SJ.LIB
' for handling joystick input and serial port I/O.  May be altered
' to use mouse input for task control (see comments in MOUSINPUT sub).
'
DEFINT I-N
DECLARE SUB automessage (msgleft, msgrt)
DECLARE SUB blend ()
DECLARE SUB communicate (dfreq, nrad)
DECLARE SUB keypressed (keyasc, keyfunct)
DECLARE SUB monitgrid (igridcol)
DECLARE SUB monitoring (lmonoff)
DECLARE SUB mousinput ()
DECLARE SUB progexit (lerror)
DECLARE SUB ratesetup ()
DECLARE SUB ratings ()
DECLARE SUB resource (keyhit, ifault)
DECLARE SUB resourceauto (lrmode)
DECLARE SUB resourcegrid (irescolor)
DECLARE SUB savecomm (nactiv, freq)
DECLARE SUB savedata (istim, rt1, rt2)
DECLARE SUB savefuel (npump)
DECLARE SUB savetrack (trstart, ssqtr, xntr)
DECLARE SUB schedulcom (i)
DECLARE SUB schedultrk (i)
DECLARE SUB scriptcode (levent)
DECLARE SUB scrngets ()
DECLARE SUB scrnstuff ()
DECLARE SUB tracking (jetcode)
DECLARE SUB trackgrid (igridcol)
DECLARE SUB warnlights (lcode)

DIM SHARED lab1(290), lab2(146), lab3(178), lab4(242), lab5(322), lab6(146)
DIM SHARED lab7(210), lwedg(80), jet1(500), levtype(900), etime(900)
DIM SHARED mon1vec(63), mon2vec(42), mon3vec(79), mon4vec(35)
DIM SHARED mauto(50), mcomtsk(50), cursor(15, 1), mcursor(60)
DIM SHARED tmlog(900), ltyp(900), rt(900), tmtrv(900), ssqv(900)
DIM SHARED ndpump(900), tmfuel(900), xntrv(900), tmcom(900)
DIM SHARED icomact(900), freqv(900), lfa(900), lfb(900), lfc(900), lfd(900)
DIM SHARED lrate1(120), lrate2(100), lxpos(6), lxold(6), lypos(6)

COMMON SHARED maxyscr, tnow, pi2, iqa, iqb, iqaff, iqbff, joyxinit, joyyinit
COMMON SHARED maxmauto, maxmcom, indcom, indcomax, itrktog
COMMON SHARED mctl, mb, mx, my, initx, inity, lfx, indrt, mantrksel, ibckgnd
COMMON SHARED indrtmax, ixtr, ixtrmax, indfuel, indfuelmax, itrksens, itrcode
COMMON SHARED clktics, ticpause, totlpause, trkincsav, trkxinc, trkyinc
COMMON SHARED fildat1$, fildat2$, fildat3$, fildat4$, fildat5$
COMMON SHARED idropa, idropb, ipflow1, ipflow2, ipflow3, ipflow4, ipflow5
COMMON SHARED ipflow6, ipflow7, ipflow8, msgleft, msgrt, ictlf7
COMMON SHARED levtask, levtaskold, lresdisp, lpumpfail, lcomtask, lighttask
COMMON SHARED k1, k2, k3, k4, k5, k6, k7, k8, lmontask, moncode, mwrking
COMMON SHARED ltogred, ltoggrn, ltogm1, ltogm2, ltogm3, ltogm4
COMMON SHARED rtred, rtgrn, rtmon1, rtmon2, rtmon3, rtmon4
COMMON SHARED lyello1, lyello2, lyello3, lyello4
COMMON SHARED lmonp1, lmonp2, lmonp3, lmonp4
COMMON SHARED rtyel1, rtyel2, rtyel3, rtyel4

RANDOMIZE TIMER
pi2 = 6.28318
trkxinc = .03: trkyinc = .0225   'medium tracking difficulty default
ticpause = 0!: totlpause = 0!    'keep pause time
initx = 340: inity = 80          'initial trk position
```

```
imx = 210: imy = 188              'initial mouse cursor position
indrt = 0: ixtr = 0: indcom = 0: indfuel = 0  'indices for data arrays
tmoutyello = 1.5     'timeout for monitoring response confirmation
ibckgnd = 0          'screen bckgnd color default

'max length of data arrays before disk write

indrtmax = 900     'monitoring data array
ixtrmax = 900      'tracking
indcomax = 900     'communications
indfuelmax = 900   'resource mgmt

maxscript = 900    'max length of script
maxyscr = 349       'max Y screen in mode 9

CLS : SCREEN 9: WIDTH 80, 43
ratesetup           'setup rating form
scrngets            'setup screen gets
scrnstuff           'draw static screen stuff
monitgrid 11        'draw mon grid
resourcegrid 11     'draw res grid
communicate 0!, 99  'init communications task
warnlights 2        'turn on green light
monitoring 99       'setup mon wedges

COLOR 14, ibckgnd: LOCATE 21, 14: PRINT DATE$;

' obtain script and output filenames from setup generated MATFILE4.NAM

filsup$ = "MATFILE4.NAM"
OPEN "I", #1, filsup$
  LINE INPUT #1, filscr$
  LINE INPUT #1, fildat1$
  LINE INPUT #1, fildat2$
  LINE INPUT #1, fildat3$
  LINE INPUT #1, fildat4$
  LINE INPUT #1, fildat5$
  INPUT #1, itrksens, tmoutgrn, tmoutred, tmoutmon, trkincsav
  INPUT #1, idropa, idropb, ipflow1, ipflow2, ipflow3, ipflow4
  INPUT #1, ipflow5, ipflow6, ipflow7, ipflow8
CLOSE #1

itrksens = 51 - itrksens

' display version, script, and data filenames

COLOR 15: LOCATE 4, 37: PRINT "MAT Version 4.0J";
LOCATE 6, 28: PRINT "ASCII file "; filsup$; " indicates:";
LOCATE 8, 28: PRINT "SCRIPT file is:"
LOCATE 9, 30: PRINT filscr$
LOCATE 11, 28: PRINT "Data files:"
LOCATE 12, 30: PRINT fildat1$
LOCATE 13, 30: PRINT fildat2$
LOCATE 14, 30: PRINT fildat3$
LOCATE 15, 30: PRINT fildat4$
LOCATE 16, 30: PRINT fildat5$
LOCATE 19, 28: PRINT "Enter RETURN to begin, ESC to exit:";

DO      'wait for CR or ESC
  a$ = UCASE$(INKEY$)
  IF a$ = "C" THEN        'permit background color adjust via X & C keys
    ibckgnd = ibckgnd + 1
    IF ibckgnd > 63 THEN ibckgnd = 0
    COLOR 15, ibckgnd
    LOCATE 21, 58: PRINT USING "##"; ibckgnd;
  END IF
  IF a$ = "X" THEN
    ibckgnd = ibckgnd - 1
    IF ibckgnd < 0 THEN ibckgnd = 63
    COLOR 15, ibckgnd
    LOCATE 21, 58: PRINT USING "##"; ibckgnd;
  END IF
```

```
  IF a$ = CHR$(27) THEN CLS : SCREEN 0: END
LOOP UNTIL a$ = CHR$(13)

CALL qjoy(joyxinit, joyyinit)       'obtain initial joystick values
LINE (201, 12)-(519, 151), 0, BF    'erase text area in tracking window

' set up mouse

mwrking = 1: mx = 0
'mctl = 0: CALL mouse(mctl, mb, mx, my)
'mctl = 4: CALL mouse(mctl, mb, imx, imy)
'mctl = 3: CALL mouse(mctl, mb, mx, my)
IF mx = 0 THEN mwrking = 0

' read script file

OPEN "I", #1, "MATNUM.SCR"
i = 0: j = 0: k = 0
DO UNTIL EOF(1)
  i = i + 1: IF i > maxscript THEN progexit 2
  INPUT #1, nhrs, nmins, nsec, levtype(i)
  etime(i) = nhrs * 3600 + nmins * 60 + nsec
  IF levtype(i) = 71 OR levtype(i) = 72 THEN j = j + 1: mauto(j) = i
  IF levtype(i) = 55 OR levtype(i) = 56 THEN k = k + 1: mcomtsk(k) = i
  IF j > 49 OR k > 49 THEN progexit 2
LOOP
CLOSE #1
maxevents = i
maxmauto = j
maxmcom = k

' date and time stamp output data files

OPEN fildat1$ FOR OUTPUT AS #1
  PRINT #1, DATE$; " "; TIME$; " "; fildat1$
CLOSE #1

OPEN fildat2$ FOR OUTPUT AS #2
  PRINT #2, DATE$; " "; TIME$; " "; fildat2$
CLOSE #2

OPEN fildat3$ FOR OUTPUT AS #3
  PRINT #3, DATE$; " "; TIME$; " "; fildat3$
CLOSE #3

OPEN fildat4$ FOR OUTPUT AS #4
  PRINT #4, DATE$; " "; TIME$; " "; fildat4$
CLOSE #4

OPEN fildat5$ FOR OUTPUT AS #5
  PRINT #5, DATE$; " "; TIME$; " "; fildat5$
CLOSE #5

CALL rsout(255)     'start signal to RS-232 port
clktics = TIMER     'hold time of day tics til exit
TIME$ = "00:00:00"  'set clock to zero
tbegin = TIMER
oldt = tbegin: tnow = tbegin

' initialize prior to realtime

trackgrid 8                 'put up gray grid
tracking 99                 'setup tracking
itrcode = 2: itrcodeb = 2   'init tracking to auto
msgleft = 1: msgrt = 2      'select msg
automessage msgleft, msgrt  'display tracking msg

lscrpindx = 1               'init script index
mantrksel = 1: itrktog = 0  'subj control trking, trk toggle
ltoggrn = 0: ltogred = 0    'toggles for light task (off)
ictlf7 = 0                  'ctl-f7 toggle (off)
```

```
ltogm1 = 0: ltogm2 = 0: ltogm3 = 0: ltog4 = 0   'dial toggles
secm2 = -2!: seclast = -1!      'timing counters

lyello1 = 0: lyello2 = 0: lyello3 = 0: lyello4 = 0 'resp confirm tog
rtyel1 = 0!: rtyel2 = 0!: rtyel3 = 0!: rtyel4 = 0! 'resp confirm timer

levtask = 2: levtaskold = 2     'init task level
nby2 = 1: nresby2 = 1           'alternation toggles

' task toggles 0=Off 1=ON
lcomtask = 1: lmontask = 1: lighttask = 1: lpumpfail = 1: lresdisp = 1

schedultrk 99: schedulcom 99    'init scheduling
resource 99, 0                  'init resource mgmt
'PUT (mx, my), mcursor           'put mouse cursor on screen

' R E A L T I M E   L O O P ---------------------------------------

'   ictr = 0    'counter to check for spare time in loop
realtime:
'   ictr = ictr + 1  'incr counter to check for spare time in loop

'mousinput       'check for mouse inputs and update mouse cursor

' detect remote inputs via Serial link

CALL rsin(ircv) 'get char from RS-232 port
  IF ircv > 100 THEN keyfunct = ircv - 100: keypressed 0, keyfunct
  IF ircv > 0 THEN keyasc = ircv: keypressed keyasc, 0

' detect keypresses

a$ = INKEY$
IF a$ <> "" THEN
    keyasc = ASC(a$)
    IF keyasc = 0 THEN keyfunct = ASC(RIGHT$(a$, 1)) ELSE keyfunct = 0
'     LOCATE 41, 5: PRINT keyasc; keyfunct;  'display codes for key pressed
    keypressed keyasc, keyfunct
END IF

' obtain present time & do 1 & 2 sec updates

tnow = TIMER
secnow = FIX(tnow)
IF secnow <> seclast THEN
  LOCATE 21, 3: COLOR 14: PRINT TIME$; 'show elapsed time each second
  seclast = secnow
  IF nresby2 = 1 THEN
    nresby2 = 0
    IF lresdisp = 0 THEN resourceauto 1  'pumps on AUTO
    resource 0, 1
  ELSE
    nresby2 = 1
  END IF
END IF

' check elapsed time for new script events

IF tnow >= etime(lscrpindx) THEN
    levent = levtype(lscrpindx)
    scriptcode levent             'act on script item
    lscrpindx = lscrpindx + 1     'increment index
    IF lscrpindx > maxevents THEN progexit 1
END IF

' begin approx 1/10 sec updates

IF tnow - oldt < .1 THEN GOTO skiptenth
oldt = oldt + .1

IF levtask <> levtaskold THEN blend  'change task mix based on new task level
```

```
monitoring moncode  'update monitoring task

IF itrktog = 1 THEN
  itrktog = 0
  IF mantrksel > 0 AND itrcode = 1 THEN
    itrcode = 2: msgleft = 1: msgrt = 2: automessage msgleft, msgrt
  ELSEIF mantrksel > 0 AND itrcode = 2 THEN
    itrcode = 1: msgleft = 2: msgrt = 1: automessage msgleft, msgrt
  END IF
END IF

IF itrcode <> itrcodeb THEN
  IF itrcode = 1 THEN tracking 3 ELSE tracking 4
  itrcodeb = itrcode
END IF
tracking itrcode    'update tracking

'  LOCATE 42, 3: PRINT ictr; : ictr = 0  'diag chk of prog spare time

' check for timeout on monitoring tasks

IF ltoggrn > 0 AND tnow - rtgrn > tmoutgrn THEN
  warnlights 2: ltoggrn = 0
  savedata 5, tnow, rtgrn
END IF

IF ltogred > 0 AND tnow - rtred > tmoutred THEN
  warnlights 3: ltogred = 0
  savedata 6, tnow, rtred
END IF

IF ltogm1 > 0 AND tnow - rtmon1 > tmoutmon THEN
  moncode = 1: ltogm1 = 0
  savedata 1, tnow, rtmon1
END IF

IF ltogm2 > 0 AND tnow - rtmon2 > tmoutmon THEN
  moncode = 2: ltogm2 = 0
  savedata 2, tnow, rtmon2
END IF

IF ltogm3 > 0 AND tnow - rtmon3 > tmoutmon THEN
  moncode = 3: ltogm3 = 0
  savedata 3, tnow, rtmon3
END IF

IF ltogm4 > 0 AND tnow - rtmon4 > tmoutmon THEN
  moncode = 4: ltogm4 = 0
  savedata 4, tnow, rtmon4
END IF

' check for timeout of prob yellow

lyeltot = lyello1 + lyello2 + lyello3 + lyello4
IF lyeltot = 0 THEN GOTO skiptenth  'see if we can skip all checks

IF lyello1 > 0 AND tnow - rtyel1 > tmoutyello THEN
  'PUT (mx, my), mcursor
  lyello1 = 0: LINE (28, 135)-(47, 138), 0, BF
  'PUT (mx, my), mcursor
ELSEIF lyello1 > 0 THEN
  lmonp1 = 0
END IF

IF lyello2 > 0 AND tnow - rtyel2 > tmoutyello THEN
  'PUT (mx, my), mcursor
  lyello2 = 0: LINE (68, 135)-(87, 138), 0, BF
  'PUT (mx, my), mcursor
ELSEIF lyello2 > 0 THEN
  lmonp2 = 0
END IF
```

```
IF lyello3 > 0 AND tnow - rtyel3 > tmoutyello THEN
  'PUT (mx, my), mcursor
  lyello3 = 0: LINE (108, 135)-(127, 138), 0, BF
  'PUT (mx, my), mcursor
ELSEIF lyello3 > 0 THEN
  lmonp3 = 0
END IF

IF lyello4 > 0 AND tnow - rtyel4 > tmoutyello THEN
  'PUT (mx, my), mcursor
  lyello4 = 0: LINE (148, 135)-(167, 138), 0, BF
  'PUT (mx, my), mcursor
ELSEIF lyello4 > 0 THEN
  lmonp4 = 0
END IF

skiptenth: 'S K I P T E N T H  end of approx 1/10th interval section

keyasc = 0: keyfunct = 0

GOTO realtime
END
```

```
'
' A U T O M E S S A G E sub ----------------------------------------
'   J. R. Comstock, Jr.
SUB automessage (msgleft, msgrt)

'PUT (mx, my), mcursor
  LINE (202, 158)-(290, 168), 0, BF

  SELECT CASE msgleft '1 blank, 2 AUTO AVAIL, 3 [AUTO END]

    CASE 2
    COLOR 10: LOCATE 21, 27: PRINT "AUTO AVAIL";

    CASE 3
    LINE (202, 158)-(290, 168), 14, BF
    PUT (206, 160), lab6

    CASE ELSE
  END SELECT

COLOR 10
  SELECT CASE msgrt

    CASE 1
    LOCATE 21, 57: PRINT " MANUAL";

    CASE 2
    LOCATE 21, 57: PRINT "AUTO ON";

    CASE ELSE
  END SELECT
'PUT (mx, my), mcursor
END SUB
```

```
'
' B L E N D  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB blend STATIC

COLOR 8: LOCATE 43, 76: PRINT levtask; 'show level lower rt corner

  i = 0 - levtask: savedata i, tnow, tnow   'save level in mon file (neg vals)

  IF levtask <= 5 THEN  'monitoring tasks
    lighttask = 1   'new problems TRUE
    lmontask = 1
  ELSE
    lighttask = 0   'new problems FALSE
    lmontask = 0
  END IF

  SELECT CASE levtask
    CASE 1
      trkxinc = .04: trkyinc = .03  'high trk difficulty
      savetrack tnow, trkxinc, 0!
      itrcode = 1       'tracking MANUAL
      lcomtask = 1      'comm task ON
      lpumpfail = 1     'possible pump fails TRUE
      lresdisp = 1      'res mgmt MANUAL

    CASE 2
      trkxinc = .01: trkyinc = .0075 'low trk dif
      savetrack tnow, trkxinc, 0!
      itrcode = 1       'tracking MANUAL
      lcomtask = 1      'comm task ON
      lpumpfail = 1     'possible pump fails TRUE
      lresdisp = 1      'res mgmt MANUAL

    CASE 3
      itrcode = 2       'tracking AUTO
      lcomtask = 0      'comm task OFF
      lpumpfail = 1     'possible pump fails TRUE
      lresdisp = 1      'res mgmt MANUAL

    CASE 4
      itrcode = 2       'tracking AUTO
      lcomtask = 0      'comm task OFF
      lpumpfail = 0     'possible pump fails FALSE
      lresdisp = 1      'res mgmt MANUAL

    CASE 5
      itrcode = 2       'tracking AUTO
      lcomtask = 0      'comm task OFF
      lpumpfail = 0     'possible pump fails FALSE
      lresdisp = 0      'res mgmt AUTO

    CASE 6
      itrcode = 2       'tracking AUTO
      lcomtask = 0      'comm task OFF
      lpumpfail = 0     'possible pump fails FALSE
      lresdisp = 0      'res mgmt AUTO

    CASE ELSE
  END SELECT

  IF levtask >= 5 AND levtaskold <= 4 THEN  'show auto display
      resourceauto 2
  END IF

  IF levtask <= 4 AND levtaskold >= 5 THEN  'switch to MANUAL
    resource 0, 1
    savefuel 79
  END IF
  msgleft = 1: automessage msgleft, itrcode
  levtaskold = levtask
END SUB
```

```
'
' C O M M U N I C A T E   sub ----------------------------------------
'   J. R. Comstock, Jr.
SUB communicate (dfreq, nrad) STATIC

' setup

IF nrad = 99 THEN
    freq1 = 108.5: freq2 = 110.3: freq3 = 119.7: freq4 = 120.9
    COLOR 14
    LOCATE 29, 12: PRINT USING "####.#"; freq1;
    LOCATE 31, 12: PRINT USING "####.#"; freq2;
    LOCATE 33, 12: PRINT USING "####.#"; freq3;
    LOCATE 35, 12: PRINT USING "####.#"; freq4;
    COLOR 11: LOCATE 29, 19: PRINT "("; CHR$(27); ")("; CHR$(26); ")";
    LOCATE 29, 4: PRINT CHR$(25); CHR$(24);
    iradio = 1

    COLOR 9: LOCATE 25, 5: PRINT "Callsign "; : COLOR 14: PRINT "NGT504";
    COLOR 3
    LOCATE 29, 7: PRINT "NAV1";
    LOCATE 31, 7: PRINT "NAV2";
    LOCATE 33, 7: PRINT "COM1";
    LOCATE 35, 7: PRINT "COM2";
    LINE (63, 292)-(120, 306), 11, B
    LOCATE 38, 10: PRINT "ENTER";
    EXIT SUB
END IF

' change radio & up/down arrows

IF nrad <> 0 THEN
    'PUT (mx, my), mcursor
    LOCATE 27 + iradio * 2, 19: PRINT "      ";
    LOCATE 27 + iradio * 2, 4: PRINT "  ";
    IF nrad > 10 THEN
      iradio = nrad - 10
    ELSE
      iradio = iradio + nrad
      IF iradio > 4 THEN iradio = 1
      IF iradio < 1 THEN iradio = 4
    END IF
    COLOR 11: LOCATE 27 + iradio * 2, 19: PRINT "("; CHR$(27); ")("; CHR$(26); ")";
    LOCATE 27 + iradio * 2, 4: PRINT CHR$(25); CHR$(24);
    savecomm iradio, 0!
    'PUT (mx, my), mcursor
    EXIT SUB
END IF
COLOR 14

  SELECT CASE iradio

    CASE 1 'nav1
    freq1 = freq1 + dfreq
    IF freq1 > 118! THEN freq1 = 108.1
    IF freq1 < 108! THEN freq1 = 117.9
    LOCATE 29, 12: PRINT USING "####.#"; freq1;
    savecomm iradio, freq1

    CASE 2 'nav2
    freq2 = freq2 + dfreq
    IF freq2 > 118! THEN freq2 = 108.1
    IF freq2 < 108! THEN freq2 = 117.9
    LOCATE 31, 12: PRINT USING "####.#"; freq2;
    savecomm iradio, freq2

    CASE 3 'com1
    freq3 = freq3 + dfreq
    IF freq3 > 136! THEN freq3 = 118.1
    IF freq3 < 118! THEN freq3 = 135.9
    LOCATE 33, 12: PRINT USING "####.#"; freq3;
    savecomm iradio, freq3
```

```
    CASE 4 'com2
    freq4 = freq4 + dfreq
    IF freq4 > 136! THEN freq4 = 118.1
    IF freq4 < 118! THEN freq4 = 135.9
    LOCATE 35, 12: PRINT USING "####.#"; freq4;
    savecomm iradio, freq4

    CASE ELSE
  END SELECT
END SUB
```

```
'
' K E Y P R E S S E D  sub ---------------------------------
'   J. R. Comstock, Jr.
SUB keypressed (keyasc, keyfunct) STATIC

  SELECT CASE keyasc       'non-function keys
    CASE 0

    CASE 13
      savecomm 13, 0!

    CASE 33, 35, 36, 37 'shift 1 3 4 5 to fail pumps
      jj = keyasc + 48: resource 1, jj: EXIT SUB

    CASE 64 'shift 2 fail p2
      resource 1, 82: EXIT SUB

    CASE 94 'shift 6 fail p6
      resource 1, 86: EXIT SUB

    CASE 38 'shift 7 fail p7
      resource 1, 87: EXIT SUB

    CASE 42 'shift 8 fail p8
      resource 1, 88: EXIT SUB

    CASE 49 TO 56 '1 to 8 toggle pumps
      resource keyasc, 0: EXIT SUB

    CASE ELSE
  END SELECT

  SELECT CASE keyfunct       'function keys
    CASE 0
      EXIT SUB

    CASE 59 'f1
      moncode = 1
      IF ltogm1 = 0 THEN
        savedata 1, tnow, tnow!
      ELSE
        ltogm1 = 0: savedata 1, rtmon1, tnow
        'PUT (mx, my), mcursor
        LINE (28, 135)-(47, 138), 14, BF
        'PUT (mx, my), mcursor
        lyello1 = 1: rtyel1 = tnow: lmonp1 = 0
      END IF

    CASE 60 'f2
      moncode = 2
      IF ltogm2 = 0 THEN
        savedata 2, tnow, tnow!
      ELSE
        ltogm2 = 0: savedata 2, rtmon2, tnow
        'PUT (mx, my), mcursor
        LINE (68, 135)-(87, 138), 14, BF
        'PUT (mx, my), mcursor
        lyello2 = 1: rtyel2 = tnow: lmonp2 = 0
      END IF

    CASE 61 'f3
      moncode = 3
      IF ltogm3 = 0 THEN
        savedata 3, tnow, tnow!
      ELSE
        ltogm3 = 0: savedata 3, rtmon3, tnow
        'PUT (mx, my), mcursor
        LINE (108, 135)-(127, 138), 14, BF
        'PUT (mx, my), mcursor
        lyello3 = 1: rtyel3 = tnow: lmonp3 = 0
      END IF
```

```
CASE 62 'f4
  moncode = 4
  IF ltogm4 = 0 THEN
    savedata 4, tnow, tnow!
  ELSE
    ltogm4 = 0: savedata 4, rtmon4, tnow
    'PUT (mx, my), mcursor
    LINE (148, 135)-(167, 138), 14, BF
    'PUT (mx, my), mcursor
    lyello4 = 1: rtyel4 = tnow: lmonp4 = 0
  END IF

CASE 63 'f5
  IF ltoggrn = 0 THEN
    savedata 5, tnow, tnow
  ELSE
    warnlights 2: ltoggrn = 0: savedata 5, rtgrn, tnow
  END IF

CASE 64 'f6
  IF ltogred = 0 THEN
    savedata 6, tnow, tnow
  ELSE
    warnlights 3: ltogred = 0: savedata 6, rtred, tnow
  END IF

CASE 65 'f7 decr tracking sens if ^f7 invoked first
  IF ictlf7 > 0 THEN
    itrksens = itrksens + 1
    IF itrksens > 50 THEN itrksens = 50
    LOCATE 42, 2: PRINT USING "###"; 51 - itrksens;
  END IF

CASE 66 'f8 incr tracking sens if ^f7 invoked first
  IF ictlf7 > 0 THEN
    itrksens = itrksens - 1
    IF itrksens < 3 THEN itrksens = 3
    LOCATE 42, 2: PRINT USING "###"; 51 - itrksens;
  END IF

CASE 72 'up arrow Comm task
  communicate 0!, -1

CASE 75 'left arrow
  communicate -.2, 0

CASE 77 'right arrow
  communicate .2, 0

CASE 80 'down arrow
  communicate 0!, 1

CASE 84 'shift-f1 res mgmt AUTO
  IF lresdisp = 1 THEN
    lresdisp = 0
    resourceauto 2
  END IF

CASE 85 'shift-f2 res mgmt MANUAL
  IF lresdisp = 0 THEN
    lresdisp = 1
    resource 0, 1
    savefuel 79
  END IF

CASE 86 'shift-f3
  levtask = levtask + 1
  IF levtask > 6 THEN levtask = 6

CASE 87 'shift-f4
  levtask = levtask - 1
  IF levtask < 1 THEN levtask = 1
```

```
    CASE 94 'Ctl-f1
      CALL rsout(57)   'send ownship code to voice computer

    CASE 95 'Ctl-f2
      CALL rsout(58)   'send other ship msg

    CASE 96 'Ctl-f3 present rating screen
      CALL rsout(10)
      ratings
      CALL rsout(250)

    CASE 97 'Ctl-f4  low trk difficulty
'     trkxinc = .02: trkyinc = .015
      trkxinc = .01: trkyinc = .0075
      savetrack tnow, trkxinc, 0!

    CASE 98 'Ctl-f5  med trk difficulty
      trkxinc = .03: trkyinc = .0225
      savetrack tnow, trkxinc, 0!

    CASE 99 'Ctl-f6  hi trk difficulty
      trkxinc = .04: trkyinc = .03
      savetrack tnow, trkxinc, 0!

    CASE 100 'Ctl-f7
'     IF itrcode = 2 THEN
        itrcode = 1
        mantrksel = 0: ictlf7 = 1
        msgleft = 1: msgrt = 1
        automessage msgleft, msgrt
'       LOCATE 42, 2: PRINT USING "###"; 51 - itrksens;
'     END IF

    CASE 101 'Ctl-f8
'     IF ictlf7 > 0 THEN
        ictlf7 = 0: itrcode = 2: mantrksel = 1
        msgleft = 1: msgrt = 2
        automessage msgleft, msgrt
        LOCATE 42, 2: PRINT "   ";
'     END IF

    CASE 102 'Ctl-f9 to pause
      ticpause = TIMER: tpause$ = TIME$
      LOCATE 42, 6: PRINT "**** PAUSED ****";
      DO
        B$ = INKEY$
        IF B$ <> "" THEN
          IF ASC(B$) = 0 AND ASC(RIGHT$(B$, 1)) = 102 THEN EXIT DO
        END IF
      LOOP
      LOCATE 42, 6: PRINT SPC(15);
      ticpause = TIMER - ticpause
      TIME$ = tpause$
      totlpause = totlpause + ticpause

    CASE 103 'Ctl-f10 program exit
      CALL rsout(254)  'send end signal to RS-232 port
      progexit 4

    CASE 104 'Alt-f1 scale 1 bias
      IF ltogm1 = 0 THEN
        IF INT(RND * 2!) = 0 THEN moncode = 5 ELSE moncode = 9
        rtmon1 = tnow
        ltogm1 = 1
      END IF

    CASE 105 'Alt-f2 scale 2 bias
      IF ltogm2 = 0 THEN
        IF INT(RND * 2!) = 0 THEN moncode = 6 ELSE moncode = 10
        rtmon2 = tnow
        ltogm2 = 1
```

```
      END IF

    CASE 106 'Alt-f3 scale 3 bias
      IF ltogm3 = 0 THEN
        IF INT(RND * 2!) = 0 THEN moncode = 7 ELSE moncode = 11
        rtmon3 = tnow
        ltogm3 = 1
      END IF

    CASE 107 'Alt-f4 scale 4 bias
      IF ltogm4 = 0 THEN
        IF INT(RND * 2!) = 0 THEN moncode = 8 ELSE moncode = 12
        rtmon4 = tnow
        ltogm4 = 1
      END IF

    CASE 108 'Alt-f5 green off
      IF ltoggrn = 0 THEN
        warnlights 1
        rtgrn = tnow
        ltoggrn = 1
      END IF

    CASE 109 'Alt-f6 red on
      IF ltogred = 0 THEN
        warnlights 4
        rtred = tnow
        ltogred = 1
      END IF

    CASE 120 TO 127 'Alt 1 to Alt 8 fix pump failures
      jj = keyfunct - 29: resource 1, jj

    CASE ELSE
      savedata 0, tnow, tnow

  END SELECT
END SUB
```

```
'
' M O N I T G R I D  sub ---------------------------------
'   J. R. Comstock, Jr.
SUB monitgrid (igridcol) STATIC

  LINE (25, 22)-(75, 44), 11, B
  LINE (27, 24)-(73, 42), 11, B
  LINE (115, 22)-(165, 44), 11, B
  LINE (117, 24)-(163, 42), 11, B

'probability

FOR lx = 40 TO 160 STEP 40
  LINE (lx - 15, 64)-(lx + 10, 140), igridcol, B
  LINE (lx, 102)-(lx + 9, 102), igridcol
  LINE (lx + 5, 110)-(lx + 9, 110), igridcol
  LINE (lx + 5, 118)-(lx + 9, 118), igridcol
  LINE (lx + 5, 94)-(lx + 9, 94), igridcol
  LINE (lx + 5, 86)-(lx + 9, 86), igridcol
  LINE (lx + 5, 126)-(lx + 9, 126), igridcol
  LINE (lx + 5, 78)-(lx + 9, 78), igridcol
NEXT lx

  COLOR igridcol
  IF igridcol = 11 THEN COLOR 3
  LOCATE 5, 11: PRINT "F5";
  LOCATE 5, 22: PRINT "F6";
  LOCATE 19, 5: PRINT "F1   F2   F3   F4";

END SUB
```

```
'
' M O N I T O R I N G  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB monitoring (monoffset) STATIC

  SELECT CASE monoffset

    CASE 99 ' setup initially
      FOR lx = 28 TO 148 STEP 40    'put wedges in initial position
        PUT (lx, 96), lwedg
      NEXT lx

      lmid = 96: lup = 85: ldown = 107  'position for offsets
      loff1 = 96: loff2 = 96: loff3 = 96: loff4 = 96  'current positions
      mon1b = 96: mon2b = 96: mon3b = 96: mon4b = 96  'previous positions
      xsin1 = 0: xsin2 = 0: xsin3 = 0: xsin4 = 0
      lmonp1 = 0: lmonp2 = 0: lmonp3 = 0: lmonp4 = 0

' calculate sine wave arrays of different frequencies for each gauge

      DO UNTIL xsin1 > pi2
        lmonp1 = lmonp1 + 1       'increment array index
        xsin1 = xsin1 + .1  'increment for sine function
        mon1vec(lmonp1) = INT(SIN(xsin1) * 8!)
      LOOP
      lmonp1max = lmonp1

      DO UNTIL xsin2 > pi2
        lmonp2 = lmonp2 + 1       'increment array index
        xsin2 = xsin2 + .15  'increment for sine function
        mon2vec(lmonp2) = INT(SIN(xsin2) * 8!)
      LOOP
      lmonp2max = lmonp2

      DO UNTIL xsin3 > pi2
        lmonp3 = lmonp3 + 1       'increment array index
        xsin3 = xsin3 + .08  'increment for sine function
        mon3vec(lmonp3) = INT(SIN(xsin3) * 8!)
      LOOP
      lmonp3max = lmonp3

      DO UNTIL xsin4 > pi2
        lmonp4 = lmonp4 + 1       'increment array index
        xsin4 = xsin4 + .18  'increment for sine function
        mon4vec(lmonp4) = INT(SIN(xsin4) * 8!)
      LOOP
      lmonp4max = lmonp4

      lmonp1 = 0: lmonp2 = 0: lmonp3 = 0: lmonp4 = 0

'length of sine wave arrays
'LOCATE 41, 2: PRINT lmonp1max; lmonp2max; lmonp3max; lmonp4max

' monoffset indicates required offset

    CASE 1 'normal case
    loff1 = lmid
    CASE 2
    loff2 = lmid
    CASE 3
    loff3 = lmid
    CASE 4
    loff4 = lmid

    CASE 5 'up case
    loff1 = lup
    CASE 6
    loff2 = lup
    CASE 7
    loff3 = lup
    CASE 8
    loff4 = lup
```

```
    CASE 9 'down case
    loff1 = ldown
    CASE 10
    loff2 = ldown
    CASE 11
    loff3 = ldown
    CASE 12
    loff4 = ldown

    CASE ELSE
  END SELECT

' increment indices for sine wave arrays and start over if at end of array

lmonp1 = lmonp1 + 1: IF lmonp1 > lmonp1max THEN lmonp1 = 1
lmonp2 = lmonp2 + 1: IF lmonp2 > lmonp2max THEN lmonp2 = 1
lmonp3 = lmonp3 + 1: IF lmonp3 > lmonp3max THEN lmonp3 = 1
lmonp4 = lmonp4 + 1: IF lmonp4 > lmonp4max THEN lmonp4 = 1

' graphic position is sine wave - offset

mon1 = loff1 - mon1vec(lmonp1)
mon2 = loff2 - mon2vec(lmonp2)
mon3 = loff3 - mon3vec(lmonp3)
mon4 = loff4 - mon4vec(lmonp4)

' erase old position, put up new position

IF mon1b <> mon1 THEN
  PUT (28, mon1b), lwedg: PUT (28, mon1), lwedg
  mon1b = mon1
END IF

IF mon2b <> mon2 THEN
  PUT (68, mon2b), lwedg: PUT (68, mon2), lwedg
  mon2b = mon2
END IF

IF mon3b <> mon3 THEN
  PUT (108, mon3b), lwedg: PUT (108, mon3), lwedg
  mon3b = mon3
END IF

IF mon4b <> mon4 THEN
  PUT (148, mon4b), lwedg: PUT (148, mon4), lwedg
  mon4b = mon4
END IF

END SUB
```

```
'
' M O U S I N P U T  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB mousinput STATIC

' To use mouse input for task control and rating scales, make the following
' changes:  1. Uncomment all lines containing CALL mouse.  2. Uncomment all
' lines containing PUT (mx, my), mcursor.  3. Uncomment mousinput statement
' near beginning of realtime loop in main program.  4. Switch to alternate
' locate & print statements in ratings sub (two places, both commented).
' 5. Use QuickBasic Libraries incorporating Mouse calls (see SJ.DOC).

mbold = mb: mxold = mx: myold = my
'CALL mouse(mctl, mb, mx, my)

'   LOCATE 39, 2: PRINT mctl; mb; mx; my  'diag chk for mouse values

IF mx > 632 THEN mx = 632   'keep PUT image on screen
IF my > 338 THEN my = 338
IF mxold <> mx OR myold <> my THEN
  PUT (mxold, myold), mcursor: PUT (mx, my), mcursor
END IF

' detect mouse button and location, mx=x position, my=y position

IF mb - mbold > 0 THEN
  IF my < 154 THEN
    IF my > 22 AND my < 45 THEN
      IF mx > 26 AND mx < 94 THEN keypressed 0, 63      'grn f5
      IF mx > 115 AND mx < 184 THEN keypressed 0, 64    'red f6
    END IF

    IF my > 65 THEN
      IF mx > 26 AND mx < 50 THEN keypressed 0, 59      'mon1 f1
      IF mx > 66 AND mx < 90 THEN keypressed 0, 60      'mon2 f2
      IF mx > 106 AND mx < 130 THEN keypressed 0, 61    'mon3 f3
      IF mx > 146 AND mx < 170 THEN keypressed 0, 62    'mon4 f4
    END IF
  ELSE
    IF mx > 224 THEN
      IF my > 157 AND my < 171 THEN
        IF mx > 431 AND mx < 519 THEN itrktog = 1       'man/auto trk
      END IF
      IF my > 255 AND my < 274 THEN
        IF mx > 224 AND mx < 256 THEN resource 49, 0    'pump 1
        IF mx > 296 AND mx < 328 THEN resource 50, 0    'pump 2
        IF mx > 376 AND mx < 409 THEN resource 51, 0    'pump 3
        IF mx > 448 AND mx < 482 THEN resource 52, 0    'pump 4
      END IF
      IF mx > 339 AND mx < 359 THEN
        IF my > 190 AND my < 217 THEN resource 55, 0    'pump 7
        IF my > 218 AND my < 242 THEN resource 56, 0    'pump 8
      END IF
      IF my > 294 AND my < 319 THEN
        IF mx > 264 AND mx < 283 THEN resource 53, 0    'pump 5
        IF mx > 415 AND mx < 435 THEN resource 54, 0    'pump 6
      END IF
    ELSE
      IF mx > 23 AND mx < 138 THEN
        IF my > 223 AND my < 233 THEN communicate 0, 11   'NAV 1
        IF my > 239 AND my < 249 THEN communicate 0, 12   'NAV 2
        IF my > 255 AND my < 265 THEN communicate 0, 13   'COM 1
        IF my > 271 AND my < 281 THEN communicate 0, 14   'COM 2
        IF my > 286 AND my < 310 THEN
          IF mx > 60 AND mx < 124 THEN keypressed 13, 0   'enter
        END IF
      END IF
      IF my > 223 AND my < 281 THEN
        IF mx > 145 AND mx < 165 THEN communicate -.2, 0  'freq dwn
        IF mx > 169 AND mx < 189 THEN communicate .2, 0   'freq up
      END IF
    END IF
```

```
  END IF
END IF
END SUB
```

```
'
' P R O G E X I T  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB progexit (lerror)

CLS : SCREEN 0: WIDTH 80, 25
PRINT "End of Run": PRINT

' write monitoring data file

IF indrt = 0 THEN GOTO secondfile

  OPEN fildat1$ FOR APPEND AS #1
  FOR i = 1 TO indrt
    PRINT #1, USING "#####.##"; tmlog(i);
    PRINT #1, USING " ##"; ltyp(i);
    PRINT #1, USING " ####.##"; rt(i)
  NEXT i
  CLOSE #1

' write tracking data file

secondfile:
tracking 4
IF ixtr = 0 THEN GOTO thirdfile

  OPEN fildat2$ FOR APPEND AS #2
  FOR i = 1 TO ixtr
    IF xntrv(i) > 0! THEN
      rms = SQR(ssqv(i) / xntrv(i))
    ELSE
      rms = 0!
    END IF
    PRINT #2, USING "#####.##"; tmtrv(i);
    PRINT #2, USING " ########.##"; ssqv(i);
    PRINT #2, USING " ####"; xntrv(i);
    PRINT #2, USING " ####.##"; rms
  NEXT i
  CLOSE #2

' write resource mgmt data file

thirdfile:
IF indfuel = 0 THEN GOTO fourthfile

  OPEN fildat3$ FOR APPEND AS #3
  FOR i = 1 TO indfuel
    PRINT #3, USING "#####.##"; tmfuel(i);
    PRINT #3, USING " ##"; ndpump(i);
    PRINT #3, USING " ####"; lfa(i);
    PRINT #3, USING " ####"; lfb(i);
    PRINT #3, USING " ####"; lfc(i);
    PRINT #3, USING " ####"; lfd(i)
  NEXT i
  CLOSE #3

' write communications data file

fourthfile:
IF indcom = 0 THEN GOTO prtmsg

  OPEN fildat4$ FOR APPEND AS #4
  FOR i = 1 TO indcom
    PRINT #4, USING "#####.##"; tmcom(i);
    PRINT #4, USING " ##"; icomact(i);
    PRINT #4, USING " ####.##"; freqv(i)
  NEXT i
  CLOSE #4

prtmsg:
  SELECT CASE lerror
    CASE 1
```

```
    PRINT "Exceeded maximum number of events in script"
    CASE 2
    PRINT "Exceeded script array boundary"
    CASE 3
    PRINT "Normal exit at end of script"
    CASE 4
    PRINT "User terminated"
    CASE ELSE
  END SELECT

'fix clock

PRINT : PRINT "Task run time: "; TIME$
clktics = clktics + TIMER + totlpause
zhrs = INT(clktics / 3600!)
zmin = INT((clktics - zhrs * 3600!) / 60!)
zsec = INT(clktics - (zhrs * 3600! + zmin * 60!))
IF zhrs > 23 THEN zhrs = zhrs - 24
z1$ = RIGHT$(STR$(zhrs), 2): IF LEFT$(z1$, 1) = " " THEN z1$ = "0" + RIGHT$(z1$, 1)
z2$ = RIGHT$(STR$(zmin), 2): IF LEFT$(z2$, 1) = " " THEN z2$ = "0" + RIGHT$(z2$, 1)
z3$ = RIGHT$(STR$(zsec), 2): IF LEFT$(z3$, 1) = " " THEN z3$ = "0" + RIGHT$(z3$, 1)
clkset$ = z1$ + ":" + z2$ + ":" + z3$
TIME$ = clkset$
PRINT : PRINT "Time set to: "; TIME$
END
END SUB
```

```
'
' R A T E S E T U P  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB ratesetup STATIC

SCREEN 9, , 1, 1        'switch screen pages
COLOR 15, 1

LINE (15, 10)-(624, 339), 15, B     'screen border
LINE (16, 309)-(623, 309), 15
LOCATE 4, 35: PRINT "RATING SCALES";

FOR iypos = 70 TO 275 STEP 41       'draw scales
  LINE (170, iypos)-(470, iypos), 15
  FOR ixpos = 170 TO 470 STEP 30
    LINE (ixpos, iypos)-(ixpos, iypos - 14), 15
  NEXT ixpos
  FOR ixpos = 185 TO 455 STEP 30
    LINE (ixpos, iypos)-(ixpos, iypos - 7), 15
  NEXT ixpos
  LINE (320, iypos - 15)-(320, iypos - 18), 15
NEXT iypos

FOR irow = 10 TO 35 STEP 5
  LOCATE irow, 18: PRINT "Low";
  LOCATE irow, 61: PRINT "High";
NEXT irow

LOCATE 25, 17: PRINT "Good";
LOCATE 25, 61: PRINT "Poor";
LOCATE 8, 7: PRINT "MENTAL DEMAND";
LOCATE 13, 6: PRINT "PHYSICAL DEMAND";
LOCATE 18, 6: PRINT "TEMPORAL DEMAND";
LOCATE 23, 9: PRINT "PERFORMANCE";
LOCATE 28, 13: PRINT "EFFORT";
LOCATE 33, 9: PRINT "FRUSTRATION";

COLOR 7, ibckgnd
SCREEN 9, , 0, 0        'switch back
END SUB
```

```
'
' R A T I N G S  sub ---------------------------------------------------
'   J. R. Comstock, Jr.
SUB ratings STATIC

ticpause = TIMER: tpause$ = TIME$   'hold time
'PUT (mx, my), mcursor
mxhold = mx: myhold = my

SCREEN 9, , 1, 1        'switch to second screen page
COLOR 15, 1

'LOCATE 41, 6: PRINT " Mouse or "; CHR$(27); " "; CHR$(26); " to Move"; SPC(12);
'PRINT "LEFT Mouse Button or "; CHR$(25); " for Next Scale ";

' when using CALL mouse then use above two lines & comment out two below

LOCATE 41, 17: PRINT CHR$(27); "  "; CHR$(26); "  to Move"; SPC(21);
PRINT CHR$(25); "  for Next Scale"; SPC(2);

FOR i = 1 TO 6          'initialize
  lxold(i) = 320
  lxpos(i) = -9
  lypos(i) = 31 + i * 41
NEXT i

nscale = 0: lendflg = 0: mb = 0: tslimit = 60!

scaleloop:
  nscale = nscale + 1
  IF nscale > 6 THEN
'    LOCATE 41, 6: COLOR 14, 1
'    PRINT "LEFT Mouse Button or "; CHR$(25); " for Changes";
'    PRINT SPC(6); "RIGHT Button or RETURN to Exit";

' when using CALL mouse then use above three lines & comment out two below

    LOCATE 41, 17: COLOR 14, 1
    PRINT CHR$(25); "  for Changes"; SPC(18); "RETURN or ESC to Exit";

    DO              'wait for input following all 6 scales
      mbx = mb
      IF TIMER - ticpause >= tslimit THEN GOTO exitscale
'      CALL mouse(mctl, mb, mx, my)
      IF mb - mbx = 2 THEN GOTO exitscale
      IF mb - mbx = 1 THEN EXIT DO
      ry$ = INKEY$
      IF ry$ <> "" THEN
        IF ASC(ry$) = 0 AND ASC(RIGHT$(ry$, 1)) = 80 THEN EXIT DO
      END IF
      IF ry$ = CHR$(27) OR ry$ = CHR$(13) THEN GOTO exitscale
      IF ry$ = CHR$(32) THEN EXIT DO
    LOOP

    lendflg = 1
    nscale = 1
  END IF

  PUT (lxold(nscale) - 12, lypos(nscale)), lrate1
'  mctl = 4: CALL mouse(mctl, mb, lxold(nscale), my): mctl = 3
  IF lxpos(nscale) > 0 THEN
    PUT (lxold(nscale) - 9, lypos(nscale)), lrate2
  ELSE
    lxpos(nscale) = 320
  END IF

  DO
    ry$ = INKEY$
    lchr = 0: keyfct = 0
    IF ry$ = "" THEN GOTO chkmouse
    lchr = ASC(ry$): IF lchr = 0 THEN keyfct = ASC(RIGHT$(ry$, 1)) ELSE keyfct = 0
    IF keyfct = 75 THEN
```

```
      lxpos(nscale) = lxpos(nscale) - 10
'      mctl = 4: CALL mouse(mctl, mb, lxold(nscale) - 10, my): mctl = 3
    ELSEIF keyfct = 77 THEN
      lxpos(nscale) = lxpos(nscale) + 10
'      mctl = 4: CALL mouse(mctl, mb, lxold(nscale) + 10, my): mctl = 3
    END IF

chkmouse:
    mbx = mb
'    CALL mouse(mctl, mb, mx, my)
    IF mb - mbx = 1 THEN EXIT DO
    IF mwrking = 1 THEN lxpos(nscale) = mx
    IF lxpos(nscale) < 170 THEN lxpos(nscale) = 170
    IF lxpos(nscale) > 470 THEN lxpos(nscale) = 470
    IF lxold(nscale) <> lxpos(nscale) THEN
      PUT (lxold(nscale) - 12, lypos(nscale)), lrate1
      PUT (lxpos(nscale) - 12, lypos(nscale)), lrate1
      lxold(nscale) = lxpos(nscale)
    END IF
    IF lchr = 32 OR keyfct = 80 THEN EXIT DO
    IF lendflg > 0 THEN
      IF lchr = 27 OR lchr = 13 OR mb - mbx = 2 THEN
        PUT (lxold(nscale) - 12, lypos(nscale)), lrate1
        PUT (lxold(nscale) - 9, lypos(nscale)), lrate2
        GOTO exitscale
      END IF
    END IF
    IF TIMER - ticpause >= tslimit THEN EXIT DO
  LOOP
  PUT (lxold(nscale) - 12, lypos(nscale)), lrate1
  PUT (lxold(nscale) - 9, lypos(nscale)), lrate2
GOTO scaleloop

exitscale:

OPEN fildat5$ FOR APPEND AS #5
  PRINT #5, USING "#####.##"; ticpause; TIMER - ticpause;
  FOR nscale = 1 TO 6
    PUT (lxold(nscale) - 9, lypos(nscale)), lrate2
    PRINT #5, USING " ###"; INT((lxold(nscale) - 170) / 3);
  NEXT nscale
  PRINT #5,
CLOSE #5

  COLOR 7, ibckgnd
  SCREEN 9, , 0, 0
  mx = mxhold: my = myhold
'  mctl = 4: CALL mouse(mctl, mb, mx, my): mctl = 3
  'PUT (mx, my), mcursor

  ticpause = TIMER - ticpause
  TIME$ = tpause$
  totlpause = totlpause + ticpause

END SUB
```

```
'
' R E S O U R C E  sub ---------------------------------------------
'   J. R. Comstock, Jr.
SUB resource (keyhit, ifault) STATIC

COLOR 14
'PUT (mx, my), mcursor
IF keyhit = 0 THEN GOTO tanks
IF ifault > 80 THEN GOTO pumpfail

  SELECT CASE keyhit

    CASE 99 'setup & initialize
    k1 = 0: k2 = 0: k3 = 0: k4 = 0  'pump off = 0, pump on = 2
    k5 = 0: k6 = 0: k7 = 0: k8 = 0

    lpf1 = 0: lpf2 = 0: lpf3 = 0: lpf4 = 0  'normal = 0, pump failure = 12
    lpf5 = 0: lpf6 = 0: lpf7 = 0: lpf8 = 0

    ipu1 = 0: ipu2 = 0: ipu3 = 0: ipu4 = 0  'pump rates per minute
    ipu5 = 0: ipu6 = 0: ipu7 = 0: ipu8 = 0

    iqa = 2109: iqb = 2121      'tank quantities for A & B
    iqaff = 1005: iqbff = 1010  'tank quantity supply tanks
    i30sec = -1: i10sec = -2    'initialize timer counters

    CASE 49 'pump 1
      IF lpf1 = 0 THEN
        IF k1 = 0 THEN
          k1 = 2: ipu1 = ipflow1: savefuel 1
        ELSE
          k1 = 0: ipu1 = 0: savefuel -1
        END IF
        LOCATE 27, 72: PRINT USING "####"; ipu1;
        LINE (233, 261)-(243, 268), k1, BF
      END IF

    CASE 50 'pump 2
      IF lpf2 = 0 THEN
        IF k2 = 0 THEN
          k2 = 2: ipu2 = ipflow2: savefuel 2
        ELSE
          k2 = 0: ipu2 = 0: savefuel -2
        END IF
        LOCATE 29, 72: PRINT USING "####"; ipu2;
        LINE (306, 261)-(316, 268), k2, BF
      END IF

    CASE 51 'pump 3
      IF lpf3 = 0 THEN
        IF k3 = 0 THEN
          k3 = 2: ipu3 = ipflow3: savefuel 3
        ELSE
          k3 = 0: ipu3 = 0: savefuel -3
        END IF
        LOCATE 31, 72: PRINT USING "####"; ipu3;
        LINE (384, 261)-(394, 268), k3, BF
      END IF

    CASE 52 'pump 4
      IF lpf4 = 0 THEN
        IF k4 = 0 THEN
          k4 = 2: ipu4 = ipflow4: savefuel 4
        ELSE
          k4 = 0: ipu4 = 0: savefuel -4
        END IF
        LOCATE 33, 72: PRINT USING "####"; ipu4;
        LINE (457, 261)-(467, 268), k4, BF
      END IF

    CASE 53 'pump 5
      IF lpf5 = 0 THEN
```

```
        IF k5 = 0 THEN
          k5 = 2: ipu5 = ipflow5: savefuel 5
        ELSE
          k5 = 0: ipu5 = 0: savefuel -5
        END IF
        LOCATE 35, 72: PRINT USING "####"; ipu5;
        LINE (269, 306)-(279, 313), k5, BF
      END IF

    CASE 54 'pump 6
      IF lpf6 = 0 THEN
        IF k6 = 0 THEN
          k6 = 2: ipu6 = ipflow6: savefuel 6
        ELSE
          k6 = 0: ipu6 = 0: savefuel -6
        END IF
        LOCATE 37, 72: PRINT USING "####"; ipu6;
        LINE (420, 306)-(430, 313), k6, BF
      END IF

    CASE 55 'pump 7
      IF lpf7 = 0 THEN
        IF k7 = 0 THEN
          k7 = 2: ipu7 = ipflow7: savefuel 7
        ELSE
          k7 = 0: ipu7 = 0: savefuel -7
        END IF
        LOCATE 39, 72: PRINT USING "####"; ipu7;
        LINE (342, 206)-(352, 213), k7, BF
      END IF

    CASE 56 'pump 8
      IF lpf8 = 0 THEN
        IF k8 = 0 THEN
          k8 = 2: ipu8 = ipflow8: savefuel 8
        ELSE
          k8 = 0: ipu8 = 0: savefuel -8
        END IF
        LOCATE 41, 72: PRINT USING "####"; ipu8;
        LINE (342, 221)-(352, 228), k8, BF
      END IF

    CASE ELSE
  END SELECT
'PUT (mx, my), mcursor
EXIT SUB

pumpfail:

  SELECT CASE ifault

    CASE 81 'fail p1
      lpf1 = 12: k1 = 0: ipu1 = 0: savefuel 81
      LOCATE 27, 72: PRINT USING "####"; ipu1;
      LINE (233, 261)-(243, 268), lpf1, BF

    CASE 82 'fail p2
      lpf2 = 12: k2 = 0: ipu2 = 0: savefuel 82
      LOCATE 29, 72: PRINT USING "####"; ipu2;
      LINE (306, 261)-(316, 268), lpf2, BF

    CASE 83 'fail p3
      lpf3 = 12: k3 = 0: ipu3 = 0: savefuel 83
      LOCATE 31, 72: PRINT USING "####"; ipu3;
      LINE (384, 261)-(394, 268), lpf3, BF

    CASE 84 'fail p4
      lpf4 = 12: k4 = 0: ipu4 = 0: savefuel 84
      LOCATE 33, 72: PRINT USING "####"; ipu4;
      LINE (457, 261)-(467, 268), lpf4, BF

    CASE 85 'fail p5
```

```
        lpf5 = 12: k5 = 0: ipu5 = 0: savefuel 85
        LOCATE 35, 72: PRINT USING "####"; ipu5;
        LINE (269, 306)-(279, 313), lpf5, BF

      CASE 86 'fail p6
        lpf6 = 12: k6 = 0: ipu6 = 0: savefuel 86
        LOCATE 37, 72: PRINT USING "####"; ipu6;
        LINE (420, 306)-(430, 313), lpf6, BF

      CASE 87 'fail p7
        lpf7 = 12: k7 = 0: ipu7 = 0: savefuel 87
        LOCATE 39, 72: PRINT USING "####"; ipu7;
        LINE (342, 206)-(352, 213), lpf7, BF

      CASE 88 'fail p8
        lpf8 = 12: k8 = 0: ipu8 = 0: savefuel 88
        LOCATE 41, 72: PRINT USING "####"; ipu8;
        LINE (342, 221)-(352, 228), lpf8, BF

  ' fix failed pumps

      CASE 91 'fix p1
        lpf1 = 0: k1 = 0: ipu1 = 0: savefuel 91
        LOCATE 27, 72: PRINT USING "####"; ipu1;
        LINE (233, 261)-(243, 268), lpf1, BF

      CASE 92 'fix p2
        lpf2 = 0: k2 = 0: ipu2 = 0: savefuel 92
        LOCATE 29, 72: PRINT USING "####"; ipu2;
        LINE (306, 261)-(316, 268), lpf2, BF

      CASE 93 'fix p3
        lpf3 = 0: k3 = 0: ipu3 = 0: savefuel 93
        LOCATE 31, 72: PRINT USING "####"; ipu3;
        LINE (384, 261)-(394, 268), lpf3, BF

      CASE 94 'fix p4
        lpf4 = 0: k4 = 0: ipu4 = 0: savefuel 94
        LOCATE 33, 72: PRINT USING "####"; ipu4;
        LINE (457, 261)-(467, 268), lpf4, BF

      CASE 95 'fix p5
        lpf5 = 0: k5 = 0: ipu5 = 0: savefuel 95
        LOCATE 35, 72: PRINT USING "####"; ipu5;
        LINE (269, 306)-(279, 313), lpf5, BF

      CASE 96 'fix p6
        lpf6 = 0: k6 = 0: ipu6 = 0: savefuel 96
        LOCATE 37, 72: PRINT USING "####"; ipu6;
        LINE (420, 306)-(430, 313), lpf6, BF

      CASE 97 'fix p7
        lpf7 = 0: k7 = 0: ipu7 = 0: savefuel 97
        LOCATE 39, 72: PRINT USING "####"; ipu7;
        LINE (342, 206)-(352, 213), lpf7, BF

      CASE 98 'fix p8
        lpf8 = 0: k8 = 0: ipu8 = 0: savefuel 98
        LOCATE 41, 72: PRINT USING "####"; ipu8;
        LINE (342, 221)-(352, 228), lpf8, BF

      CASE ELSE
    END SELECT
  'PUT (mx, my), mcursor
  EXIT SUB

  tanks:

  ' calculate tank quantities

  iqadev = ipu1 + ipu2 + ipu8 - (idropa + ipu7)
  iqbdev = ipu3 + ipu4 + ipu7 - (idropb + ipu8)
```

```
iqa = iqa + iqadev / 30
iqb = iqb + iqbdev / 30
iqaff = iqaff + (ipu5 - ipu1) / 30
iqbff = iqbff + (ipu6 - ipu3) / 30

' tank A empty

IF iqa < 0 AND k7 = 2 THEN
  iqa = 0: ipu7 = 0: k7 = 0
  LOCATE 39, 72: PRINT USING "####"; ipu7;
  LINE (342, 206)-(352, 213), lpf7, BF
  savefuel 70
ELSEIF iqa < 0 THEN
  iqa = 0
END IF

' tank B empty

IF iqb < 0 AND k8 = 2 THEN
  iqb = 0: ipu8 = 0: k8 = 0
  LOCATE 41, 72: PRINT USING "####"; ipu8;
  LINE (342, 221)-(352, 228), lpf8, BF
  savefuel 80
ELSEIF iqb < 0 THEN
  iqb = 0
END IF

' supply tank A side empty

IF iqaff < 0 THEN
  iqaff = 0: ipu1 = 0: k1 = 0
  LOCATE 27, 72: PRINT USING "####"; ipu1;
  LINE (233, 261)-(243, 268), lpf1, BF
  savefuel 10
END IF

' supply tank B side empty

IF iqbff < 0 THEN
  iqbff = 0: ipu3 = 0: k3 = 0
  LOCATE 31, 72: PRINT USING "####"; ipu3;
  LINE (384, 261)-(394, 268), lpf3, BF
  savefuel 30
END IF

' supply tank A side full

IF iqaff > 2000 THEN
  iqaff = 2000: ipu5 = 0: k5 = 0
  LOCATE 35, 72: PRINT USING "####"; ipu5;
  LINE (269, 306)-(279, 313), lpf5, BF
  savefuel 50
END IF

' supply tank B side full

IF iqbff > 2000 THEN
  iqbff = 2000: ipu6 = 0: k6 = 0
  LOCATE 37, 72: PRINT USING "####"; ipu6;
  LINE (420, 306)-(430, 313), lpf6, BF
  savefuel 60
END IF

' tank A full

IF iqa > 4000 THEN
  iqa = 4000: k8 = 0: k1 = 0: k2 = 0
  ipu8 = 0: LOCATE 41, 72: PRINT USING "####"; ipu8; : LINE (342, 221)-(352, 228), lpf8, BF
  ipu1 = 0: LOCATE 27, 72: PRINT USING "####"; ipu1; : LINE (233, 261)-(243, 268), lpf1, BF
  ipu2 = 0: LOCATE 29, 72: PRINT USING "####"; ipu2; : LINE (306, 261)-(316, 268), lpf2, BF
  savefuel 80: savefuel 10: savefuel 20
END IF
```

```
' tank B full

IF iqb > 4000 THEN
  iqb = 4000: k7 = 0: k3 = 0: k4 = 0
  ipu7 = 0: LOCATE 39, 72: PRINT USING "####"; ipu7; : LINE (342, 206)-(352, 213), lpf7, BF
  ipu3 = 0: LOCATE 31, 72: PRINT USING "####"; ipu3; : LINE (384, 261)-(394, 268), lpf3, BF
  ipu4 = 0: LOCATE 33, 72: PRINT USING "####"; ipu4; : LINE (457, 261)-(467, 268), lpf4, BF
  savefuel 70: savefuel 30: savefuel 40
END IF

' display tank quantities

IF lresdisp = 1 THEN
  LOCATE 32, 33: PRINT USING "####"; iqa;
  LOCATE 32, 52: PRINT USING "####"; iqb;
  LOCATE 43, 29: PRINT USING "####"; iqaff;
  LOCATE 43, 48: PRINT USING "####"; iqbff;
END IF

' save resource data automatically at 30 sec increments

i30sec = i30sec + 1
IF i30sec >= 15 THEN i30sec = 0: savefuel 0

' update scheduling window every 10 sec

i10sec = i10sec + 1
IF i10sec >= 5 THEN i10sec = 0: schedultrk 0: schedulcom 0

' calculate graphic coordinates

mga = 198 + (42 - iqa / 95)
mgb = 198 + (42 - iqb / 95)
mgaff = 283 + (45 - iqaff / 44)
mgbff = 283 + (45 - iqbff / 44)

LINE (249, mga)-(296, 240), 2, BF
LINE (249, 198)-(296, mga), 0, BF

LINE (398, mgb)-(445, 240), 2, BF
LINE (398, 198)-(445, mgb), 0, BF

LINE (225, mgaff)-(252, 328), 2, BF
LINE (225, 283)-(252, mgaff), 0, BF

LINE (376, mgbff)-(403, 328), 2, BF
LINE (376, 283)-(403, mgbff), 0, BF

'PUT (mx, my), mcursor
END SUB
```

```
'
' R E S O U R C E A U T O  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB resourceauto (lrmode) STATIC

SELECT CASE lrmode
  CASE 1
      IF iqa > 2550 AND k2 > 0 THEN resource 50, 0
      IF iqa < 2480 AND k2 = 0 THEN resource 50, 0
      IF iqb > 2550 AND k4 > 0 THEN resource 52, 0
      IF iqb < 2480 AND k4 = 0 THEN resource 52, 0

      IF iqaff > 1900 AND k5 > 0 THEN resource 53, 0
      IF iqaff < 1700 AND k5 = 0 THEN resource 53, 0
      IF iqbff > 1900 AND k6 > 0 THEN resource 54, 0
      IF iqbff < 1700 AND k6 = 0 THEN resource 54, 0

      IF iqa > 2520 AND k1 > 0 THEN resource 49, 0
      IF iqa < 2450 AND k1 = 0 THEN resource 49, 0
      IF iqb > 2520 AND k3 > 0 THEN resource 51, 0
      IF iqb < 2450 AND k3 = 0 THEN resource 51, 0

      IF iqa - iqb > 25 AND k7 = 0 THEN resource 55, 0
      IF iqa - iqb <= 25 AND k7 > 0 THEN resource 55, 0
      IF iqb - iqa > 25 AND k8 = 0 THEN resource 56, 0
      IF iqb - iqa <= 25 AND k8 > 0 THEN resource 56, 0

  CASE 2
      'PUT (mx, my), mcursor
      COLOR 2
      LOCATE 32, 33: PRINT "AUTO";
      LOCATE 32, 52: PRINT "AUTO";
      LOCATE 43, 29: PRINT SPC(4);
      LOCATE 43, 48: PRINT SPC(4);
      'PUT (mx, my), mcursor
      savefuel 78

  CASE ELSE
END SELECT

END SUB
```

```
'
' R E S O U R C E G R I D  sub ---------------------------------------
'   J. R. Comstock, Jr.
SUB resourcegrid (irescolor) STATIC

' fill supply tanks
LINE (297, 300)-(334, 328), 2, BF
LINE (448, 300)-(485, 328), 2, BF

COLOR 11: LOCATE 25, 69: PRINT "Flow Rates";
FOR i = 2 TO 16 STEP 2
  LOCATE 25 + i, 70: PRINT USING "#"; i / 2;
NEXT i
COLOR 14
FOR i = 2 TO 16 STEP 2
  LOCATE 25 + i, 75: PRINT "0";
NEXT i

COLOR irescolor
LOCATE 26, 30: PRINT "A"; : LOCATE 26, 58: PRINT "B";
LOCATE 37, 27: PRINT "C"; : LOCATE 37, 46: PRINT "D";
LOCATE 25, 44: PRINT "7"; CHR$(26);
LOCATE 30, 44: PRINT CHR$(27); "8";
LOCATE 34, 32: PRINT "1"; SPC(8); "2"; SPC(9); "3"; SPC(8); "4";
LOCATE 33, 29: PRINT CHR$(24); SPC(8); CHR$(24); SPC(9); CHR$(24); SPC(8); CHR$(24)
LOCATE 38, 34: PRINT CHR$(27); "5"; SPC(17); CHR$(27); "6";

' tank A side
LINE (247, 197)-(298, 242), , B     'tank A
LINE (223, 282)-(254, 330), , B     'supply left
LINE (295, 282)-(336, 330), , B     'supply right
LINE (245, 209)-(246, 219), , B   'marker for 2000-3000
LINE (245, 214)-(246, 214), 0
' tank B side
LINE (396, 197)-(447, 242), , B     'tank B
LINE (374, 282)-(405, 330), , B     'supply left
LINE (446, 282)-(487, 330), , B     'supply right
LINE (448, 209)-(449, 219), , B   'marker for 2000-3000
LINE (448, 214)-(449, 214), 0
' pumps 7 8
IF irescolor = 11 THEN COLOR 3
LINE (341, 205)-(353, 214), irescolor, B   'pump 7
LINE (341, 220)-(353, 229), irescolor, B   'pump 8
LINE (299, 210)-(340, 210)          'left of 7
LINE (354, 210)-(395, 210)          'right of 7
LINE (299, 225)-(340, 225)          'left of 8
LINE (354, 225)-(395, 225)          'right of 8
' pumps 1-6
LINE (232, 260)-(244, 269), irescolor, B 'p1
LINE (305, 260)-(317, 269), irescolor, B 'p2
LINE (383, 260)-(395, 269), irescolor, B 'p3
LINE (456, 260)-(468, 269), irescolor, B 'p4
LINE (268, 305)-(280, 314), irescolor, B 'p5
LINE (419, 305)-(431, 314), irescolor, B 'p6
LINE (238, 270)-(238, 281)
LINE (238, 259)-(238, 238)
LINE -(246, 238)
LINE (311, 270)-(311, 281)
LINE (311, 259)-(311, 238)
LINE -(299, 238)
LINE (389, 270)-(389, 281)
LINE (389, 259)-(389, 238)
LINE -(395, 238)
LINE (462, 270)-(462, 281)
LINE (462, 259)-(462, 238)
LINE -(448, 238)
LINE (255, 310)-(267, 310)
LINE (281, 310)-(294, 310)
LINE (406, 310)-(418, 310)
LINE (432, 310)-(445, 310)

END SUB
```

```
'
' S A V E C O M M  sub ---------------------------------------------
'   J. R. Comstock, Jr.
SUB savecomm (nactiv, freq)

indcom = indcom + 1   'data array index
IF indcom <= indcomax THEN GOTO storit3

' if data array is full then write it and continue

  OPEN fildat4$ FOR APPEND AS #4
  FOR i = 1 TO indcomax
    PRINT #4, USING "#####.##"; tmcom(i);
    PRINT #4, USING " ##"; icomact(i);
    PRINT #4, USING " ####.##"; freqv(i)
  NEXT i
  CLOSE #4
indcom = 1

storit3:
  tmcom(indcom) = tnow
  icomact(indcom) = nactiv
  freqv(indcom) = freq

END SUB
```

```
'
' S A V E D A T A  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB savedata (istim, rtbegin, rtend) STATIC

indrt = indrt + 1   'data array index
IF indrt <= indrtmax THEN GOTO storit

' if array is full then write it and continue

  OPEN fildat1$ FOR APPEND AS #1
  FOR i = 1 TO indrtmax
    PRINT #1, USING "#####.##"; tmlog(i);
    PRINT #1, USING " ##"; ltyp(i);
    PRINT #1, USING " ####.##"; rt(i)
  NEXT i
  CLOSE #1
indrt = 1

storit:
  tmlog(indrt) = rtbegin
  ltyp(indrt) = istim
  rt(indrt) = rtend - rtbegin

END SUB
```

```
'
' S A V E F U E L  sub ---------------------------------------------
'   J. R. Comstock, Jr.
SUB savefuel (npump)

indfuel = indfuel + 1   'data array index
IF indfuel <= indfuelmax THEN GOTO storit4

' if data array is full then write it and continue

  OPEN fildat3$ FOR APPEND AS #3
  FOR i = 1 TO indfuelmax
    PRINT #3, USING "#####.##"; tmfuel(i);
    PRINT #3, USING " ##"; ndpump(i);
    PRINT #3, USING " ####"; lfa(i);
    PRINT #3, USING " ####"; lfb(i);
    PRINT #3, USING " ####"; lfc(i);
    PRINT #3, USING " ####"; lfd(i)

  NEXT i
  CLOSE #3
indfuel = 1

storit4:
  tmfuel(indfuel) = tnow
  ndpump(indfuel) = npump
  lfa(indfuel) = iqa
  lfb(indfuel) = iqb
  lfc(indfuel) = iqaff
  lfd(indfuel) = iqbff

END SUB
```

```
'
' S A V E T R A C K  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB savetrack (trstart, ssqtr, xntr) STATIC

ixtr = ixtr + 1      'data array index
IF ixtr <= ixtrmax THEN GOTO storinfo

' if data array is full then write it and continue

  OPEN fildat2$ FOR APPEND AS #2
  FOR i = 1 TO ixtrmax
    IF xntrv(i) > 0! THEN
      rms = SQR(ssqv(i) / xntrv(i))
    ELSE
      rms = 0!
    END IF
    PRINT #2, USING "#####.##"; tmtrv(i);
    PRINT #2, USING " ########.##"; ssqv(i);
    PRINT #2, USING " ####"; xntrv(i);
    PRINT #2, USING " ####.##"; rms
  NEXT i
  CLOSE #2
ixtr = 1

storinfo:
  tmtrv(ixtr) = trstart
  ssqv(ixtr) = ssqtr
  xntrv(ixtr) = xntr

END SUB
```

```
'
' S C H E D U L C O M  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB schedulcom (i) STATIC

IF i = 99 THEN istart2 = 1   'setup
LINE (608, 27)-(610, 155), 0, BF 'erase wide line
LINE (613, 27)-(615, 155), 0, BF

comloop1:
IF istart2 + 1 > maxmcom THEN EXIT SUB
indsched = istart2

comloop2:
i = mcomtsk(indsched): t1 = etime(i) - tnow
i = mcomtsk(indsched + 1): t2 = etime(i) - tnow

IF t2 <= 0! THEN istart2 = istart2 + 2: GOTO comloop1
IF t1 >= 480! THEN EXIT SUB
lbeg = 27 + INT(t1 / 3.75)
IF lbeg < 27 THEN lbeg = 27
lend = 27 + INT(t2 / 3.75)
IF lend > 155 THEN lend = 155

LINE (608, lbeg)-(615, lend), 14, BF 'fill man
indsched = indsched + 2
IF indsched + 1 > maxmcom THEN EXIT SUB
GOTO comloop2

END SUB
```

```
'
' S C H E D U L T R K  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB schedultrk (i) STATIC

IF i = 99 THEN istart1 = 1  'setup
LINE (583, 27)-(585, 155), 0, BF 'erase wide line
LINE (588, 27)-(590, 155), 0, BF

manloop1:
IF istart1 + 1 > maxmauto THEN EXIT SUB
indsched = istart1

manloop2:
i = mauto(indsched): t1 = etime(i) - tnow
i = mauto(indsched + 1): t2 = etime(i) - tnow

IF t2 <= 0! THEN istart1 = istart1 + 2: GOTO manloop1
IF t1 >= 480! THEN EXIT SUB
lbeg = 27 + INT(t1 / 3.75)
IF lbeg < 27 THEN lbeg = 27
lend = 27 + INT(t2 / 3.75)
IF lend > 155 THEN lend = 155

LINE (583, lbeg)-(590, lend), 14, BF 'fill man
indsched = indsched + 2
IF indsched + 1 > maxmauto THEN EXIT SUB
GOTO manloop2

END SUB
```

```
'
' S C R I P T C O D E  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB scriptcode (levent) STATIC

IF lcomtask = 0 THEN    'skip com events (recode) if task OFF
  IF levent = 57 THEN levent = 59: savecomm levent, 0!
  IF levent = 58 THEN levent = 60: savecomm levent, 0!
END IF

CALL rsout(levent)  'send event code to RS-232 port

  SELECT CASE levent 'script code actions

    CASE 10 'ratings
      ratings
      CALL rsout(250)

    CASE 11, 12 'scale 1
      IF lmontask = 1 THEN
        IF ltogm1 = 0 THEN
          IF levent = 11 THEN moncode = 5 ELSE moncode = 9
          rtmon1 = tnow
          ltogm1 = 1
        END IF
      END IF

    CASE 21, 22 'scale 2
      IF lmontask = 1 THEN
        IF ltogm2 = 0 THEN
          IF levent = 21 THEN moncode = 6 ELSE moncode = 10
          rtmon2 = tnow
          ltogm2 = 1
        END IF
      END IF

    CASE 31, 32 'scale 3
      IF lmontask = 1 THEN
        IF ltogm3 = 0 THEN
          IF levent = 31 THEN moncode = 7 ELSE moncode = 11
          rtmon3 = tnow
          ltogm3 = 1
        END IF
      END IF

    CASE 41, 42 'scale 4
      IF lmontask = 1 THEN
        IF ltogm4 = 0 THEN
          IF levent = 41 THEN moncode = 8 ELSE moncode = 12
          rtmon4 = tnow
          ltogm4 = 1
        END IF
      END IF

    CASE 51 'green off
      IF lighttask = 1 THEN
        IF ltoggrn = 0 THEN
          warnlights 1
          rtgrn = tnow
          ltoggrn = 1
        END IF
      END IF

    CASE 57 'NGT 504 voice msg
      IF lcomtask = 1 THEN savecomm 57, 0!

    CASE 58 'other callsign msg
      IF lcomtask = 1 THEN savecomm 58, 0!

    CASE 61 'red on
      IF lighttask = 1 THEN
        IF ltogred = 0 THEN
```

```
            warnlights 4
            rtred = tnow
            ltogred = 1
          END IF
        END IF

      CASE 71 'manual
        itrcode = 1
        mantrksel = 0
        msgleft = 1: msgrt = 1
        automessage msgleft, msgrt

      CASE 72 'auto
        itrcode = 2
        mantrksel = 1
        msgleft = 1: msgrt = 2
        automessage msgleft, msgrt

      CASE 73 'auto going off message
        mantrksel = 1
        msgleft = 3
        automessage msgleft, msgrt

      CASE 74 'low trk difficulty
'       trkxinc = .02: trkyinc = .015
        trkxinc = .01: trkyinc = .0075
        savetrack tnow, trkxinc, 0!

      CASE 75 'med trk difficulty
        trkxinc = .03: trkyinc = .0225
        savetrack tnow, trkxinc, 0!

      CASE 76 'hi trk difficulty
        trkxinc = .04: trkyinc = .03
        savetrack tnow, trkxinc, 0!

      CASE 78 'AUTO Res Mgmt
        lresdisp = 0
        resourceauto 2

      CASE 79 'MANUAL Res Mgmt
        lresdisp = 1
        resource 0, 1
        savefuel 79

      CASE 81 TO 98
        IF lpumpfail = 1 THEN
          resource 1, levent
        ELSEIF levent > 90 THEN
          resource 1, levent
        END IF

      CASE 99 'end of script, terminate run
        progexit 3

      CASE 101 TO 106
        levtask = levent - 100

      CASE ELSE
    END SELECT
END SUB
```

```
'
' S C R N G E T S  sub ---------------------------------------------
'   J. R. Comstock, Jr.
SUB scrngets

COLOR 3
PRINT "SYSTEM MONITORING"
PRINT "TRACKING"
PRINT "SCHEDULING"
PRINT "COMMUNICATIONS"
PRINT "RESOURCE MANAGEMENT"
COLOR 10: PRINT "AUTO END"
COLOR 3: PRINT "PUMP STATUS"
GET (0, 0)-(136, 7), lab1
GET (0, 8)-(64, 15), lab2
GET (0, 16)-(80, 23), lab3
GET (0, 24)-(112, 31), lab4
GET (0, 32)-(152, 39), lab5
GET (0, 40)-(64, 47), lab6
GET (0, 48)-(88, 55), lab7
CLS

'monitoring wedge

FOR ix = 1 TO 6
  LINE (ix, ix)-(ix, 13 - ix), 14
NEXT ix
GET (1, 1)-(7, 15), lwedg
CLS

'mouse cursor

  LINE (1, 1)-(1, 10), 15
  LINE (2, 2)-(2, 9), 15
  LINE (3, 3)-(3, 8), 15
  LINE (4, 4)-(4, 7), 15
  LINE (5, 5)-(5, 7), 15
  LINE (6, 6)-(6, 7), 15
  LINE (7, 7)-(7, 7), 15
GET (1, 1)-(7, 11), mcursor
CLS

'tracking target

CIRCLE (20, 20), 12, 10
LINE (20, 20)-(20, 20), 10
LINE (20, 10)-(20, 15), 10
LINE (0, 20)-(8, 20), 10
LINE (20, 25)-(20, 30), 10
LINE (32, 20)-(40, 20), 10
GET (0, 10)-(40, 30), jet1
CLS

' ratings arrows

FOR iy = 0 TO 11
  LINE (14 - iy, iy)-(14 + iy, iy), 14
NEXT iy
GET (2, 0)-(26, 11), lrate1
CLS
FOR iy = 0 TO 8
  LINE (12 - iy, iy)-(12 + iy, iy), 7
NEXT iy
GET (3, 0)-(21, 8), lrate2
CLS

END SUB
```

```
'
' S C R N S T U F F  sub ------------------------------------------
'   J. R. Comstock, Jr.
SUB scrnstuff

' Static Screen Info

LINE (1, 0)-(638, 10), 3, BF
LINE (1, 170)-(638, 180), 3, BF
LINE (0, 0)-(639, maxyscr), 15, B
LINE (520, 11)-(520, 169), 15
LINE (520, 181)-(520, maxyscr), 15
LINE (520, 1)-(520, 10), 0
LINE (520, 170)-(520, 180), 0
LINE (200, 11)-(200, maxyscr), 15
LINE (200, 1)-(200, 10), 0
LINE (200, 170)-(200, 180), 0
LINE (1, 157)-(199, 157), 3
PUT (40, 2), lab1
PUT (320, 2), lab2
PUT (540, 2), lab3
PUT (40, 172), lab4
PUT (270, 172), lab5
PUT (535, 172), lab7

'scheduling window grid

COLOR 11
FOR i = 0 TO 8
  LOCATE (i * 2) + 4, 69: PRINT USING "##"; i;
NEXT i
FOR i = 1 TO 17
  LOCATE i + 3, 72: PRINT "-";
NEXT i
LOCATE 21, 74: PRINT "T  C";
LINE (574, 27)-(574, 155)
LINE (586, 27)-(587, 155), 14, B
LINE (611, 27)-(612, 155), 14, B

END SUB
```

```
'
' T R A C K G R I D  sub ---------------------------------
'   J. R. Comstock, Jr.
SUB trackgrid (igridcol) STATIC

      LINE (353, 90)-(367, 90), igridcol
      LINE (360, 85)-(360, 95), igridcol
      FOR ix = 240 TO 480 STEP 40
        LINE (ix - 3, 90)-(ix + 3, 90), igridcol
        LINE (ix, 88)-(ix, 92), igridcol
      NEXT ix
      FOR iy = 36 TO 144 STEP 18
        LINE (360, iy - 2)-(360, iy + 2), igridcol
        LINE (357, iy)-(363, iy), igridcol
      NEXT iy
      LINE (320, 72)-(323, 72), igridcol
      LINE (320, 72)-(320, 74), igridcol
      LINE (320, 108)-(323, 108), igridcol
      LINE (320, 108)-(320, 106), igridcol
      LINE (400, 72)-(397, 72), igridcol
      LINE (400, 72)-(400, 74), igridcol
      LINE (400, 108)-(397, 108), igridcol
      LINE (400, 108)-(400, 106), igridcol

END SUB
```

```
'
' T R A C K I N G  sub ---------------------------------
'   J. R. Comstock, Jr.
SUB tracking (jetcode) STATIC

  SELECT CASE jetcode

    CASE 1 ' M A N U A L   T R A C K I N G ---------------------

       xsin = xsin + trkxinc
       IF xsin > pi2 THEN xsin = 0
       jetxoff = SIN(xsin) * xgain

       ysin = ysin + trkyinc
       IF ysin > pi2 THEN ysin = 0
       jetyoff = SIN(ysin) * ygain

      jetx = initx - (jetxoff)
      jety = inity - (jetyoff)

      CALL qjoy(joyx, joyy)
'     LOCATE 39, 4: PRINT joyx; joyy; itrksens;

      moffx = moffx + (joyxinit - joyx) \ itrksens
      moffy = moffy + (joyyinit - joyy) \ itrksens

      jetx = jetx - moffx
      jety = jety - moffy

      IF jetx > 479 THEN jetx = 479: moffx = -139 - jetxoff
      IF jetx < 201 THEN jetx = 201: moffx = 139 - jetxoff
      IF jety > 138 THEN jety = 138: moffy = -58 - jetyoff
      IF jety < 11 THEN jety = 11: moffy = 69 - jetyoff

      IF jetx <> jetxb OR jety <> jetyb THEN
        PUT (jetxb, jetyb), jet1: PUT (jetx, jety), jet1
        jetxb = jetx: jetyb = jety
      END IF

'    LINE (jetx + 20, jety + 10)-(jetx + 21, jety + 10), 14 'leaves trail

' save data at selected interval

      IF tnow - trstart >= trkincsav THEN
        savetrack trstart, ssqtr, xntr
        trstart = tnow: ssqtr = 0!: xntr = 0!
      END IF

      xdev = jetx - initx
      ydev = (inity - jety) * 1.3714
      xntr = xntr + 1!
      ssqtr = ssqtr + (xdev * xdev + ydev * ydev)

'LOCATE 3, 48: PRINT USING "####.##"; xdev; ydev; 'show x & y

    CASE 2 ' A U T O   T R A C K I N G ------------------------

      loffset = 0

' return track target to center

      IF jetx > 343 THEN jetx = jetx - 3: loffset = 1
      IF jetx < 337 THEN jetx = jetx + 3: loffset = 1
      IF jety > 83 THEN jety = jety - 2: loffset = 1
      IF jety < 77 THEN jety = jety + 2: loffset = 1

' provide small random offsets when on auto

      IF loffset = 0 THEN
        IF iswitch = 1 THEN
          iswitch = 0
          PUT (jetx, jety), jet1
```

```
          trackgrid 8
          PUT (jetx, jety), jet1
        END IF
        mantrksel = 1
        IF RND < .3 THEN
          jetx = 339 + INT(RND * 3!)
          jety = 79 + INT(RND * 3!)
        END IF
      ELSE
        mantrksel = 0
      END IF

' if new position, then put up target

      IF jetx <> jetxb OR jety <> jetyb THEN
        PUT (jetxb, jetyb), jet1: PUT (jetx, jety), jet1
        jetxb = jetx: jetyb = jety
      END IF

    CASE 3 'switch to manual  ----------------------------------

      moffx = 0: moffy = 0: xsin = 0!: ysin = 0!
      ssqtr = 0!: xntr = 0!
      trstart = tnow
      PUT (jetx, jety), jet1
      trackgrid 14
      PUT (jetx, jety), jet1

    CASE 4 'switch to auto  ------------------------------------

      IF xntr > 0! THEN savetrack trstart, ssqtr, xntr: xntr = 0!
      savetrack tnow, 0!, 0!
      iswitch = 1

    CASE 99 'setup ------------------------
      jetxb = initx: jetyb = inity
      jetx = initx: jety = inity
      PUT (jetxb, jetyb), jet1
      xsin = 0!: ysin = 0!
      moffx = 0: moffy = 0
      iswitch = 0: xgain = 110!: ygain = 45!

    CASE ELSE
  END SELECT
END SUB
```

```
'
' W A R N L I G H T S  sub ----------------------------------------
'   J. R. Comstock, Jr.
SUB warnlights (lcode)

  'PUT (mx, my), mcursor
  SELECT CASE lcode

    CASE 1 'green off
    LINE (29, 26)-(71, 40), 0, BF

    CASE 2 'green on
    LINE (29, 26)-(71, 40), 2, BF

    CASE 3 'red off
    LINE (119, 26)-(161, 40), 0, BF

    CASE 4 'red on
    LINE (119, 26)-(161, 40), 12, BF

    CASE ELSE
  END SELECT
  'PUT (mx, my), mcursor
END SUB
```

MATREMX Program Listing

```
'Comstock  MATREMX (Ver. 1.2) remote display, task control & speech generation
'  03-25-91  J. R. Comstock, Jr. NASA LaRC: (804) 864-6643, FTS 928-6643
'
'  Connects with task computer via COM1 serial port link.
'  Hardware requirements:  Any PC or Compatible that will work
'  with the Heath Voice Card (Model HV-2000).  It has been tested
'  on a Heathkit 158 and a Leading Edge Model D.
'  Even the slowest of PCs should work fine with this program.
'  The Voice Card we used is available through the Heath Company,
'  Benton Harbor, Michigan 49022.  May be modified for use with
'  other synthesizers or digitizers.
'
'  Developed using Microsoft QuickBasic 4.5. Requires invoking
'  QuickBasic with command line QB /L SJ, in order to use the SJ.QLB
'  and compile time SJ.LIB libraries.  SJ.QLB and SJ.LIB are created
'  from SERIAL.OBJ and QJOY.OBJ.
'
DEFINT I-N

DECLARE SUB talk (icond)
DIM SHARED m1a$(100), m1b$(100), m1c$(100)
DIM SHARED M2a$(100), m2b$(100), m2c$(100)
COMMON SHARED lerrsect
ON ERROR GOTO errorhandler

start:
iscrrow = 3: ikeyrow = 3: lrestart = 0: lnewstart = 0: clktics = 0!
totlpause = 0!
COLOR 15, 1: CLS

' setup port and close it

OPEN "COM1:19200,N,8,1" FOR RANDOM AS #5 LEN = 256
CLOSE #5

PRINT : PRINT SPC(15); "MAT remote display and speech generation   (Ver 1.2)"
PRINT STRING$(80, 196);
COLOR 7, 1

talk 99  'set up voice synthesizer messages and open device

' wait til 255 to start time

VIEW PRINT 4 TO 25  'set up lines 4-24 to scroll
PRINT "  Waiting for start signal from task computer... (ESC to Quit)": PRINT

wait255:
  a$ = INKEY$
  IF a$ = CHR$(27) THEN talk 100: VIEW PRINT: COLOR 7, 0: CLS : END
  IF a$ = CHR$(98) OR a$ = CHR$(66) THEN PRINT "  User started": GOTO begin
  CALL rsin(ircv)
  IF ircv = 255 THEN PRINT "  Start signal rcvd": GOTO begin
GOTO wait255

begin:
PRINT
SOUND 1000, 1
clktics = TIMER
TIME$ = "00:00:00"

' R E A L T I M E   R C V L O O P ------------------------------------

rcvloop:
CALL rsin(ircv)     'get char from RS-232 port

a$ = INKEY$
IF a$ = "" THEN GOTO update
keyasc = ASC(a$)

IF keyasc = 0 THEN
  keyfunct = ASC(RIGHT$(a$, 1))
  CALL rsout(keyfunct + 100)
```

```
ELSE
  keyfunct = 0
  CALL rsout(keyasc)
END IF

IF a$ = CHR$(27) THEN GOTO progexit
IF a$ = "a" OR a$ = "A" THEN ircv = 57 'sim com event by 1 key
IF a$ = "b" OR a$ = "B" THEN ircv = 58 'sim com event by 2 key

update:
IF ircv < 0 THEN GOTO rcvloop     'if no char loop back

' display script item

PRINT SPC(2); TIME$; : PRINT USING " Scr####"; ircv;

  SELECT CASE ircv
    CASE 10
      ticpause = TIMER
      tpause$ = TIME$
      PRINT " Begin Ratings";

    CASE 250
      ticpause = TIMER - ticpause
      TIME$ = tpause$
      totlpause = totlpause + ticpause
      PRINT " Ratings "; ticpause; " sec";

    CASE 57
      talk 57

    CASE 58
      talk 58

    CASE 99, 254
      lnewstart = 1: GOTO progexit

    CASE ELSE
  END SELECT
PRINT
GOTO rcvloop

' P R O G R A M    E X I T  ----------------------------------------------

progexit:

IF lnewstart = 1 THEN GOTO fixclock

PRINT
PRINT "  SELECT:  RETURN - New Start,  C - Continue,  ESC - Exit"

DO
  r$ = INKEY$
  IF r$ = "C" OR r$ = "c" THEN PRINT "  Continuing": GOTO rcvloop
  IF r$ = CHR$(13) THEN lrestart = 1: EXIT DO
  IF r$ = CHR$(27) THEN lrestart = 0: EXIT DO
LOOP

fixclock:
VIEW PRINT
talk 100  'close voice device
COLOR 7, 0: CLS
PRINT : PRINT "Task run time: "; TIME$
clktics = clktics + TIMER + totlpause
zhrs = INT(clktics / 36001)
zmin = INT((clktics - zhrs * 36001) / 601)
zsec = INT(clktics - (zhrs * 36001 + zmin * 601))
IF zhrs > 23 THEN zhrs = zhrs - 24
z1$ = RIGHT$(STR$(zhrs), 2): IF LEFT$(z1$, 1) = " " THEN z1$ = "0" + RIGHT$(z1$, 1)
z2$ = RIGHT$(STR$(zmin), 2): IF LEFT$(z2$, 1) = " " THEN z2$ = "0" + RIGHT$(z2$, 1)
z3$ = RIGHT$(STR$(zsec), 2): IF LEFT$(z3$, 1) = " " THEN z3$ = "0" + RIGHT$(z3$, 1)
clkset$ = z1$ + ":" + z2$ + ":" + z3$
```

```
TIME$ = clkset$
PRINT : PRINT "Time set to: "; TIME$

IF lnewstart = 1 OR lrestart = 1 THEN GOTO start
END

' E R R O R H A N D L E R ----------------------------------------

errorhandler:
lerrnum = ERR
COLOR 7, 0: CLS
PRINT "Multi-Attribute Task Remote Terminal Program": PRINT
PRINT "ERROR NUMBER: "; lerrnum: PRINT

IF lerrnum = 24 THEN
  PRINT "Problem with RS-232 line"
END IF

IF lerrnum = 53 AND lerrsect = 1 THEN
  CLOSE #1
  PRINT "*** ERROR IN READING REQUIRED FILE:  MSG57.VOI"
END IF

IF lerrnum = 53 AND lerrsect = 2 THEN
  CLOSE #2
  PRINT "*** ERROR IN READING REQUIRED FILE:  MSG58.VOI"
END IF

PRINT : PRINT "EXITING Program": PRINT
END
```

```
'
'  T A L K   S U B  ------------------------------------------------
'   J. R. Comstock, Jr.
SUB talk (icond) STATIC

  SELECT CASE icond

    CASE 99  'setup voice messages
      msg1 = 0: msg2 = 0
      lerrsect = 1
      OPEN "I", #1, "MSG57.VOI"
      i = 0
      DO UNTIL EOF(1)
        i = i + 1
        LINE INPUT #1, m1a$(i)
        LINE INPUT #1, m1b$(i)
        LINE INPUT #1, m1c$(i)

      LOOP
        msg1max = i

      CLOSE #1
      lerrsect = 2
      OPEN "I", #2, "MSG58.VOI"
      i = 0
      DO UNTIL EOF(2)
        i = i + 1
        LINE INPUT #2, M2a$(i)
        LINE INPUT #2, m2b$(i)
        LINE INPUT #2, m2c$(i)
      LOOP
        msg2max = i
      CLOSE #2

      OPEN "O", 3, "HV"

    CASE 100  ' close device
      CLOSE #3

    CASE 57
      msg1 = msg1 + 1: IF msg1 > msg1max THEN msg1 = 1
      PRINT USING " Own  ###"; msg1;
      PRINT #3, m1a$(msg1)
      PRINT #3, m1b$(msg1)
      PRINT #3, m1c$(msg1)

    CASE 58
      msg2 = msg2 + 1: IF msg2 > msg2max THEN msg2 = 1
      PRINT USING " Other###"; msg2;
      PRINT #3, M2a$(msg2)
      PRINT #3, m2b$(msg2)
      PRINT #3, m2c$(msg2)

    CASE ELSE
  END SELECT
END SUB
```

Voice Message Listing

Voice Messages for MAT Battery Communications Task
Own Callsign Messages (MSG57.VOI) In presentation order:

NGT504, NGT504 Set First Navigation 109.7
NGT504, NGT504 Set First Radio 121.3
NGT504, NGT504 Set Second Navigation 111.5
NGT504, NGT504 Set Second Radio 119.9
NGT504, NGT504 Set Second Navigation 112.3
NGT504, NGT504 Set First Navigation 110.5
NGT504, NGT504 Set First Radio 121.9
NGT504, NGT504 Set Second Radio 121.3
NGT504, NGT504 Set First Navigation 109.7
NGT504, NGT504 Set Second Radio 121.9
NGT504, NGT504 Set Second Navigation 113.1
NGT504, NGT504 Set First Radio 121.1
NGT504, NGT504 Set First Navigation 110.7
NGT504, NGT504 Set Second Navigation 112.3
NGT504, NGT504 Set First Navigation 110.1
NGT504, NGT504 Set First Radio 121.7
NGT504, NGT504 Set Second Radio 120.7
NGT504, NGT504 Set Second Radio 120.1
NGT504, NGT504 Set Second Navigation 111.9
NGT504, NGT504 Set First Radio 120.9

Other Callsign Messages (MSG58.VOI) In presentation order:

NLS217, NLS217 Set First Radio 121.3
NRK362, NRK362 Set Second Radio 123.3
NAL478, NAL478 Set Second Navigation 117.7
NAL478, NAL478 Set Second Radio 118.3
NDL183, NDL183 Set Second Navigation 117.3
NLS217, NLS217 Set First Radio 126.1
NRK362, NRK362 Set First Navigation 113.9
NLS217, NLS217 Set First Radio 125.5
NRK362, NRK362 Set Second Navigation 113.5
NLS217, NLS217 Set Second Radio 119.1
NAL478, NAL478 Set Second Navigation 115.3
NAL478, NAL478 Set First Navigation 109.7
NDL183, NDL183 Set First Navigation 113.1
NLS217, NLS217 Set First Radio 124.9
NRK362, NRK362 Set Second Radio 121.3
NDL183, NDL183 Set First Navigation 115.7
NRK362, NRK362 Set Second Navigation 115.5
NAL478, NAL478 Set First Navigation 112.1
NDL183, NDL183 Set First Radio 127.7
NDL183, NDL183 Set Second Radio 120.1

Voice Message Listing

MSG57.VOI voice script file

```
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) navigation (661 12pa) 1 0 9 (581 2pa) point (701 3pa) 7
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) radio (661 12pa) 1 2 1 (581 2pa) point (701 3pa) 3
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) navigation (661 12pa) 1 1 1 (581 2pa) point (701 3pa) 5
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) radio (661 12pa) 1 1 9 (581 2pa) point (701 3pa) 9
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) navigation (661 12pa) 1 1 2 (581 2pa) point (701 3pa) 3
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) navigation (661 12pa) 1 1 0 (581 2pa) point (701 3pa) 5
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) radio (661 12pa) 1 2 1 (581 2pa) point (701 3pa) 9
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) radio (661 12pa) 1 2 1 (581 2pa) point (701 3pa) 3
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) navigation (661 12pa) 1 0 9 (581 2pa) point (701 3pa) 7
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) radio (661 12pa) 1 2 1 (581 2pa) point (701 3pa) 9
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) navigation (661 12pa) 1 1 3 (581 2pa) point (701 3pa) 1
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) radio (661 12pa) 1 2 1 (581 2pa) point (701 3pa) 1
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) navigation (661 12pa) 1 1 0 (581 2pa) point (701 3pa) 7
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) navigation (661 12pa) 1 1 2 (581 2pa) point (701 3pa) 3
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) navigation (661 12pa) 1 1 0 (581 2pa) point (701 3pa) 1
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) radio (661 12pa) 1 2 1 (581 2pa) point (701 3pa) 7
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) radio (661 12pa) 1 2 0 (581 2pa) point (701 3pa) 7
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) radio (661 12pa) 1 2 0 (581 2pa) point (701 3pa) 1
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) second (621) navigation (661 12pa) 1 1 1 (581 2pa) point (701 3pa) 9
=(M ! ; 8A 2pa 2D AR 881) n (751) gee (2pa 661) t (2pa 731) 5 (671) 0 (651) 4
(5pa 701) n (641) gee (2pa 831) t (2pa 501) 5 (581) 0 (621) 4
(! 721 3pa) set (801 3pa) first (621) radio (661 12pa) 1 2 0 (581 2pa) point (701 3pa) 9
```

Voice Message Listing

MSG58.VOI voice script file

```
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) l (2pa 66I) s (2pa 73I) 2 (67I) 1 (65I) 7
(5pa 70I) n (64I 2pa) l (2pa 83I) s (2pa 50I) 2 (58I) 1 (62I) 7
(! 72I 3pa) set (80I 3pa) first (62I) radio (66I 12pa) 1 2 1 (58I 2pa) point (70I 3pa) 3
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) r (2pa 66I) k (2pa 73I) 3 (67I) 6 (65I) 2
(5pa 70I) n (64I 2pa) r (2pa 83I) k (2pa 50I) 3 (58I) 6 (62I) 2
(! 72I 3pa) set (80I 3pa) second (62I) radio (66I 12pa) 1 2 3 (58I 2pa) point (70I 3pa) 3
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) a (2pa 66I) l (2pa 73I) 4 (67I) 7 (65I) 8
(5pa 70I) n (64I 2pa) a (2pa 83I) l (2pa 50I) 4 (58I) 7 (62I) 8
(! 72I 3pa) set (80I 3pa) second (62I) navigation (66I 12pa) 1 1 7 (58I 2pa) point (70I 3pa) 7
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) a (2pa 66I) l (2pa 73I) 4 (67I) 7 (65I) 8
(5pa 70I) n (64I 2pa) a (2pa 83I) l (2pa 50I) 4 (58I) 7 (62I) 8
(! 72I 3pa) set (80I 3pa) second (62I) radio (66I 12pa) 1 1 8 (58I 2pa) point (70I 3pa) 3
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) d (2pa 66I) l (2pa 73I) 1 (67I) 8 (65I) 3
(5pa 70I) n (64I 2pa) d (2pa 83I) l (2pa 50I) 1 (58I) 8 (62I) 3
(! 72I 3pa) set (80I 3pa) second (62I) navigation (66I 12pa) 1 1 7 (58I 2pa) point (70I 3pa) 3
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) l (2pa 66I) s (2pa 73I) 2 (67I) 1 (65I) 7
(5pa 70I) n (64I 2pa) l (2pa 83I) s (2pa 50I) 2 (58I) 1 (62I) 7
(! 72I 3pa) set (80I 3pa) first (62I) radio (66I 12pa) 1 2 6 (58I 2pa) point (70I 3pa) 1
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) r (2pa 66I) k (2pa 73I) 3 (67I) 6 (65I) 2
(5pa 70I) n (64I 2pa) r (2pa 83I) k (2pa 50I) 3 (58I) 6 (62I) 2
(! 72I 3pa) set (80I 3pa) first (62I) navigation (66I 12pa) 1 1 3 (58I 2pa) point (70I 3pa) 9
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) l (2pa 66I) s (2pa 73I) 2 (67I) 1 (65I) 7
(5pa 70I) n (64I 2pa) l (2pa 83I) s (2pa 50I) 2 (58I) 1 (62I) 7
(! 72I 3pa) set (80I 3pa) first (62I) radio (66I 12pa) 1 2 5 (58I 2pa) point (70I 3pa) 5
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) r (2pa 66I) k (2pa 73I) 3 (67I) 6 (65I) 2
(5pa 70I) n (64I 2pa) r (2pa 83I) k (2pa 50I) 3 (58I) 6 (62I) 2
(! 72I 3pa) set (80I 3pa) second (62I) navigation (66I 12pa) 1 1 3 (58I 2pa) point (70I 3pa) 5
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) l (2pa 66I) s (2pa 73I) 2 (67I) 1 (65I) 7
(5pa 70I) n (64I 2pa) l (2pa 83I) s (2pa 50I) 2 (58I) 1 (62I) 7
(! 72I 3pa) set (80I 3pa) second (62I) radio (66I 12pa) 1 1 9 (58I 2pa) point (70I 3pa) 1
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) a (2pa 66I) l (2pa 73I) 4 (67I) 7 (65I) 8
(5pa 70I) n (64I 2pa) a (2pa 83I) l (2pa 50I) 4 (58I) 7 (62I) 8
(! 72I 3pa) set (80I 3pa) second (62I) navigation (66I 12pa) 1 1 5 (58I 2pa) point (70I 3pa) 3
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) a (2pa 66I) l (2pa 73I) 4 (67I) 7 (65I) 8
(5pa 70I) n (64I 2pa) a (2pa 83I) l (2pa 50I) 4 (58I) 7 (62I) 8
(! 72I 3pa) set (80I 3pa) first (62I) navigation (66I 12pa) 1 0 9 (58I 2pa) point (70I 3pa) 7
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) d (2pa 66I) l (2pa 73I) 1 (67I) 8 (65I) 3
(5pa 70I) n (64I 2pa) d (2pa 83I) l (2pa 50I) 1 (58I) 8 (62I) 3
(! 72I 3pa) set (80I 3pa) first (62I) navigation (66I 12pa) 1 1 3 (58I 2pa) point (70I 3pa) 1
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) l (2pa 66I) s (2pa 73I) 2 (67I) 1 (65I) 7
(5pa 70I) n (64I 2pa) l (2pa 83I) s (2pa 50I) 2 (58I) 1 (62I) 7
(! 72I 3pa) set (80I 3pa) first (62I) radio (66I 12pa) 1 2 4 (58I 2pa) point (70I 3pa) 9
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) r (2pa 66I) k (2pa 73I) 3 (67I) 6 (65I) 2
(5pa 70I) n (64I 2pa) r (2pa 83I) k (2pa 50I) 3 (58I) 6 (62I) 2
(! 72I 3pa) set (80I 3pa) second (62I) radio (66I 12pa) 1 2 1 (58I 2pa) point (70I 3pa) 3
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) d (2pa 66I) l (2pa 73I) 1 (67I) 8 (65I) 3
(5pa 70I) n (64I 2pa) d (2pa 83I) l (2pa 50I) 1 (58I) 8 (62I) 3
(! 72I 3pa) set (80I 3pa) first (62I) navigation (66I 12pa) 1 1 5 (58I 2pa) point (70I 3pa) 7
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) r (2pa 66I) k (2pa 73I) 3 (67I) 6 (65I) 2
(5pa 70I) n (64I 2pa) r (2pa 83I) k (2pa 50I) 3 (58I) 6 (62I) 2
(! 72I 3pa) set (80I 3pa) second (62I) navigation (66I 12pa) 1 1 5 (58I 2pa) point (70I 3pa) 5
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) a (2pa 66I) l (2pa 73I) 4 (67I) 7 (65I) 8
(5pa 70I) n (64I 2pa) a (2pa 83I) l (2pa 50I) 4 (58I) 7 (62I) 8
(! 72I 3pa) set (80I 3pa) first (62I) navigation (66I 12pa) 1 1 2 (58I 2pa) point (70I 3pa) 1
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) d (2pa 66I) l (2pa 73I) 1 (67I) 8 (65I) 3
(5pa 70I) n (64I 2pa) d (2pa 83I) l (2pa 50I) 1 (58I) 8 (62I) 3
(! 72I 3pa) set (80I 3pa) first (62I) radio (66I 12pa) 1 2 7 (58I 2pa) point (70I 3pa) 7
=(M ! ; 8A 2pa 2D AR 88I) n (75I 2pa) d (2pa 66I) l (2pa 73I) 1 (67I) 8 (65I) 3
(5pa 70I) n (64I 2pa) d (2pa 83I) l (2pa 50I) 1 (58I) 8 (62I) 3
(! 72I 3pa) set (80I 3pa) second (62I) radio (66I 12pa) 1 2 0 (58I 2pa) point (70I 3pa) 1
```

SERIAL.ASM with RSOUT and RSIN through COM1 address

```
PAGE        ,132
NAME  RS232
Title RS232 Input-Output Functions
 .model medium,basic
DATA        SEGMENT WORD PUBLIC 'DATA'

DATA        ENDS
DGROUP      GROUP       DATA
CODE        SEGMENT PARA PUBLIC 'CODE'
ASSUME CS:CODE
PUBLIC RSOUT,RSIN                       ;Quick Basic function names

args                    EQU 6
Device_Status EQU 1021  ;Status port
Device_Input  EQU 1016  ;Receive port
Device_Output EQU 1016  ;Transmit port
DR                      EQU 1                   ;Data Ready
THBE                    EQU 32                  ;Transmit Hold Buffer Empty

;*****************************************************************************
; Output byte on COM1: assume it is port 1016 decimal
;
; usage:  CALL RSOUT(I%)
;
; return: nothing
;
;*****************************************************************************

RSOUT       PROC                                            ;this function ignores handshaking
            push        bp                                  ;Save BP
            mov         bp,sp
            mov         dx,Device_Status        ;Check the status
stat:       in          al,dx                               ;Get status
            and         al,THBE                             ;Mask Transmit Hold Buffer Empty byte
            cmp         al,THBE
            jne         stat                                ;Wait until Hold Buffer is empty
            mov         dx,Device_Output        ;Get device port
            mov         bx,word ptr [bp+args]   ;get output word address in program
            mov         ax,[bx]                             ;get output word from program
            out         dx,al                               ;output data
            pop         bp                                  ;Restore BP
            ret         2                                   ;Clear Stack
RSOUT       ENDP

;*****************************************************************************
; Input byte from COM1: assume it is port 1016 decimal
;
; usage:  CALL RSIN(I%)
;
; return: I% = OffffH is nothing was received
;         i% = byte value of received data
;
;*****************************************************************************

RSIN        PROC
            push        bp                                  ;Save BP
            mov         bp,sp
            mov         dx,Device_Status        ;Check for Data Ready Flag
            in          al,dx                               ;Get status
            and         al,DR                               ;Mask out Data Ready bit
            cmp         al,DR
            je          statok
            mov         ax,0ffffh               ;No data received yet
            jmp         skip
statok:     mov         dx,Device_Input                     ;Data Received
            in          al,dx                               ;Get data
            mov         ah,0                                ;Zero out high order byte
skip:       mov         bx,word ptr [bp+args]   ;Get address of data in program
            mov         [bx],ax                             ;Save data to program
            pop         bp                                  ;Restore BP
```

```
            ret         2                               ;Clear stack
RSIN        ENDP

CODE ENDS

END
```

```
    QJOY.ASM  yields full two-byte value from joystick port

PAGE    ,132
NAME    QJOY
TITLE - QJOY - Joystick SUBROUTINES to give full range readout on joystick
; position
;
;
DATA    SEGMENT PARA PUBLIC 'DATA'
DATA    ENDS
DGROUP  GROUP   DATA
;

 .model medium,basic
QJOY_CODE    SEGMENT PARA PUBLIC 'CODE'
ASSUME  CS:QJOY_CODE

PUBLIC  QJOY

;
;*****************************************************************************
;
;DESCRIPTION:  This routine
;
;  Qbasic Call Procedure
;           CALL QJOY(X%,Y%) ' Read joystick positions into variables
;                                 ' X% and Y%
;          .
;          .
;
;           X%          X joystick position, location (integer) BP+08
;           Y%          Y joystick position, location (integer) BP+06
;
;           DOS call: 35H functions
;
;   DESTROYS: No registers
;
;*****************************************************************************
;
QJOY        PROC    far
            push        bp                      ;Save register used in routine
            mov         bp,sp                   ;Save current stack pointer for arguments
            push        ax
            push        bx
            push        cx
            push        dx
            mov         ah,84h
            mov         dx,1h
            int         15h                     ; Get the joystick values
            push        bx                      ; Save bx for later use
            mov         bx,[bp+8] ;cs:xoff
            mov         word ptr [bx],ax
            pop         ax                      ; Recover bx
            mov         bx,[bp+6] ;cs:yoff
            mov         word ptr [bx],ax
            pop         dx
            pop         cx
            pop         bx
            pop         ax
            pop         bp
            ret         4
;
;           Local data
;           NONE
;
QJOY        endp
;
QJOY_CODE    ENDS
       END
```

AFTERMAT Program Listing

```
' AFTERMAT.BAS:  RATING SCALE FACTOR COMPARISONS
'  Version 1.0  (06/15/90) Chapman
'  Version 2.0  (09/30/91) Comstock
' J. R. Comstock, Jr.,  NASA Langley Research Center,  804-864-6643
'
' This program is designed to permit the Subject to make paired
' comparisons of each pair of rating scale factors.  Output of the
' program may be used as input to APPLYWT program for applying these
' weights to Rating Scale data.
'
' Written using Microsoft QuickBasic 4.5, no special hardware requirements.
'
DEFINT I-N
DECLARE SUB compare (asp1$, val1!, asp2$, val2!)
DIM vs(6), label$(6), rv(6), rv5(6), lcomp1(15), lcomp2(15), lchosen(15)
COMMON SHARED lexit, lch, j

DATA 5,3,4,2,3,2,2,3,6,4,1,6,4,1,5
FOR i = 1 TO 15
  READ lcomp1(i)
NEXT i

DATA 4,5,6,4,6,6,3,1,5,3,2,1,1,5,2
FOR i = 1 TO 15
  READ lcomp2(i)
NEXT i

label$(1) = " Mental Demand "
label$(2) = "Physical Demand"
label$(3) = "Temporal Demand"
label$(4) = "  Performance  "
label$(5) = "    Effort     "
label$(6) = "  Frustration  "

begin:
  COLOR 7, 1: CLS
  lexit = 0
  FOR i = 1 TO 6
    vs(i) = 0
  NEXT i

' Display directions

  LOCATE 1, 72: PRINT "Ver: 2.0"
  COLOR 15, 1: PRINT SPC(22); "RATING SCALE FACTOR COMPARISONS"
  COLOR 7, 1: PRINT : PRINT
  PRINT "      The evaluation you are about to complete is a technique that has been"
  PRINT " developed by NASA to assess the relative importance of six factors in"
  PRINT " determining how much workload you experienced.  The procedure is simple:"
  PRINT " You will be presented with a series of pairs of rating scale titles (for"
  PRINT " example, Effort vs. Mental Demands) and asked to choose which of the items"
  PRINT " was more important to YOUR experience of workload in the task(s) that you"
  PRINT " just performed.  Each pair of scale titles will appear on a separate screen."
  PRINT
  PRINT "      Just select the item that you thought was more important for the"
  PRINT " task you just completed."
  PRINT : PRINT : PRINT
  PRINT SPC(20); "<Press RETURN to Begin, ESC to Quit>";

' Wait for CR or ESC

  DO
    a$ = INKEY$
    IF a$ = CHR$(27) THEN COLOR 7, 0: CLS : END
    IF a$ = CHR$(13) THEN EXIT DO
  LOOP

' Display static screen stuff

   CLS
   LOCATE 2, 4: PRINT "Rating Scale Factor Comparisons"
   LOCATE 2, 60: PRINT "Comparison    of 15";
```

```
   LOCATE 23, 1: PRINT STRING$(80, 196);
   LOCATE 24, 3
   PRINT "Select the More Important Factor by Number or "; CHR$(25); CHR$(24);
   PRINT " keys"; SPC(8); "ESC to Restart";

' Comparisons

FOR j = 1 TO 15
  LOCATE 2, 71: PRINT USING "##"; j;
  i1 = lcomp1(j)
  i2 = lcomp2(j)
  compare label$(i1), vs(i1), label$(i2), vs(i2)
  IF lexit > 0 THEN GOTO begin
  lchosen(j) = lch
  LOCATE 11, 30: PRINT SPC(30);
  LOCATE 13, 39: PRINT SPC(2);
  LOCATE 15, 30: PRINT SPC(30);

  told = TIMER
  DO        'delay loop in case user holds key down too long
    tnow = TIMER
    eltime = tnow - told
    IF eltime < 0! THEN EXIT DO
    a$ = INKEY$
  LOOP UNTIL eltime > .4

NEXT j

' Display Summary Info

  CLS
  LOCATE 2, 2
  COLOR 15, 1: PRINT "  SUMMARY of Comparisons Chosen:": COLOR 7, 1
  LOCATE 4, 14
  PRINT "Top Item"; SPC(11); "Bottom Item"; SPC(9); "Item chosen"
  PRINT

FOR j = 1 TO 15
  i1 = lcomp1(j)
  i2 = lcomp2(j)
  IF lchosen(j) = 1 THEN i3 = i1 ELSE i3 = i2
  PRINT USING "  ##   "; j;
  PRINT CHR$(179); "  "; label$(i1); "  "; CHR$(179); "  "; label$(i2);
  COLOR 15, 1: PRINT "  "; CHR$(179); "  "; label$(i3): COLOR 7, 1
NEXT j

 PRINT : PRINT
 PRINT "  Enter RETURN to Continue,  (ESC to Start Comparisons Over)";

 DO
   a$ = INKEY$
   IF a$ = CHR$(27) THEN GOTO begin
 LOOP UNTIL a$ = CHR$(13)

CLS
LOCATE 2, 12: PRINT "Rating Weights:": PRINT
FOR i = 1 TO 6
  PRINT SPC(5); i; "  "; label$(i); SPC(4); vs(i)
NEXT i

COLOR 15, 1
LOCATE 13, 4
LINE INPUT "Enter Subject Initials (or ID Number): "; Init$
Init$ = UCASE$(Init$)

d$ = DATE$: t$ = TIME$
fildat$ = "WT" + LEFT$(d$, 2) + MID$(d$, 4, 2) + LEFT$(t$, 2) + "." + MID$(t$, 4, 2)
fildat$ = fildat$ + "X"

PRINT
PRINT SPC(3); "Saving Comparisons for Subject  "; Init$; "  in file  "; fildat$
PRINT : PRINT SPC(3); "(Filename based on present Date and Time)"
```

```
ict = 0
filout$ = fildat$

' Save data to output file

saveit:
OPEN filout$ FOR OUTPUT AS #1
PRINT #1, d$; " "; t$; " "; fildat$; " "; Init$
FOR i = 1 TO 15
  PRINT #1, USING "##"; lchosen(i);
NEXT i
PRINT #1,

FOR i = 1 TO 6
  PRINT #1, USING "##"; vs(i);
NEXT i
PRINT #1,
CLOSE #1

ict = ict + 1
IF ict = 1 THEN filout$ = "WTLAST.WGT": GOTO saveit

LOCATE 22, 4
PRINT "This Concludes the Rating Comparisons,  Press RETURN to EXIT!"

DO
  a$ = INKEY$
LOOP UNTIL a$ = CHR$(13)

COLOR 7, 0: CLS
END
```

```
'
'  C O M P A R E  Sub  ----------------------------------------
'
SUB compare (asp1$, val1!, asp2$, val2!)

   COLOR 15, 1
   LOCATE 11, 30: PRINT "1  "; asp1$;
   LOCATE 15, 30: PRINT "2  "; asp2$;
   COLOR 7, 1
   LOCATE 13, 39: PRINT "or"
   lexit = 0

  DO
    a$ = INKEY$
    IF a$ = CHR$(27) THEN lexit = 1: EXIT SUB
    IF a$ = CHR$(49) THEN val1 = val1 + 1: lch = 1: EXIT SUB
    IF a$ = CHR$(50) THEN val2 = val2 + 1: lch = 2: EXIT SUB

    IF LEN(a$) = 2 THEN
      IF ASC(a$) = 0 AND ASC(RIGHT$(a$, 1)) = 72 THEN
        val1 = val1 + 1: lch = 1: EXIT SUB
      END IF
      IF ASC(a$) = 0 AND ASC(RIGHT$(a$, 1)) = 80 THEN
        val2 = val2 + 1: lch = 2: EXIT SUB
      END IF
    END IF
  LOOP

END SUB
```

APPLYWT Program Listing

```
' APPLYWT.BAS  Applies Weightings to Rating data files  (Version 2.0)
'  Version 1.0  (Summer 90) Chapman
'  Version 2.0  (09/30/91) Comstock
'  J. R. Comstock, Jr.  NASA Langley Research Center  804-864-6643
'
' Rating data files are created by MAT Battery (e.g. MD092712.45X), and
' Weighting files (e.g. WT092714.45X) are created by program RATEWGT.
' This program reads both Rating data files and Weighting files and creates
' a new Rating file with a "W" suffix having original Rating info plus
' Mean Ratings and Weighted Ratings.
' Program created using Microsoft QuickBasic 4.5.  No special hardware or
' libraries needed.
'
DEFINT I-N
DECLARE SUB header ()
DIM vs(6)
ON ERROR GOTO errorhandler

begin:

' read WTLAST.WGT file

lerrsect = 1
OPEN "WTLAST.WGT" FOR INPUT AS #1
LINE INPUT #1, wtid$
LINE INPUT #1, wt15$
FOR i = 1 TO 6
 INPUT #1, vs(i)
NEXT i
CLOSE #1

wtfil$ = MID$(wtid$, 21, 12)
dt$ = LEFT$(wtid$, 20)
subid$ = MID$(wtid$, 33, 10)

menu1:

header
LOCATE 5, 12: PRINT "Weighting File Menu:"
LOCATE 8, 8: PRINT "1 - Use:  "; wtfil$; "  "; dt$; subid$
LOCATE 10, 8: PRINT "2 - Select a New Weighting File"
LOCATE 12, 8: PRINT "3 - EXIT  (Esc)"
DO
  a$ = INKEY$
  IF a$ = "1" THEN GOTO menu2
  IF a$ = "2" THEN EXIT DO
  IF a$ = "3" OR a$ = CHR$(27) THEN COLOR 7, 0: CLS : END
LOOP

wgtdirectory:
header
PRINT : PRINT "  Directory of Weighting Files:"
lerrsect = 5: PRINT : FILES "WT*.??X"

selwgt:
PRINT : LINE INPUT "  Enter Name of Weighting file: "; wtfil$
wtfil$ = UCASE$(wtfil$)

' read selected weighting file

lerrsect = 2
OPEN wtfil$ FOR INPUT AS #1
LINE INPUT #1, wtid$
LINE INPUT #1, wt15$
FOR i = 1 TO 6
 INPUT #1, vs(i)
NEXT i
CLOSE #1

wtfil$ = MID$(wtid$, 21, 12)
dt$ = LEFT$(wtid$, 20)
subid$ = MID$(wtid$, 33, 10)
```

```
menu2:
header
COLOR 15, 1
PRINT "  Using Weighting File:  "; wtfil$; "  "; dt$; subid$
COLOR 7, 1

PRINT : PRINT "  Directory of Task Rating Files:"
lerrsect = 6: PRINT : FILES "MD*.??X"

selratfil:
lerrsect = 3
PRINT : LINE INPUT "  Enter Name of Task Rating File: "; ratfil$
ratfil$ = UCASE$(ratfil$)

OPEN ratfil$ FOR INPUT AS #1

   filenew$ = LEFT$(ratfil$, 11) + "W"
   OPEN filenew$ FOR OUTPUT AS #2
   LINE INPUT #1, ln1$
   PRINT #2, ln1$
   header
   PRINT : PRINT ln1$

   iline = 0
  DO UNTIL EOF(1)
    iline = iline + 1
    INPUT #1, etime, rtime, rv(1), rv(2), rv(3), rv(4), rv(5), rv(6)
    sum = 0!: sumw = 0!
    FOR i = 1 TO 6
      sum = sum + rv(i)
      rv5(i) = (INT((rv(i) + 4) / 5)) * 5
      sumw = sumw + rv5(i) * vs(i)
    NEXT i
    xmean = sum / 6!
    wscore = sumw / 15!

    PRINT USING "#####.##"; etime; rtime;
    FOR i = 1 TO 6
      PRINT USING " ###"; rv(i);
    NEXT i
    PRINT USING " ###.###"; xmean; wscore

    PRINT #2, USING "#####.##"; etime; rtime;
    FOR i = 1 TO 6
      PRINT #2, USING " ###"; rv(i);
    NEXT i
    PRINT #2, USING " ###.###"; xmean; wscore
  LOOP
  CLOSE #1
  CLOSE #2

 PRINT
 COLOR 15, 1
 PRINT "  File "; ratfil$; "  Weighted by "; wtfil$; ",  Created "; filenew$
 PRINT "  Press Any Key When Ready..."
 COLOR 7, 1
 DO
   a$ = INKEY$
 LOOP UNTIL a$ <> ""
 GOTO menu1
END

' E R R O R H A N D L E R    ----------------------------------------

errorhandler:
lerrnum = ERR

IF lerrnum = 53 AND lerrsect = 1 THEN
  CLOSE
  wtfil$ = ""
  RESUME wgtdirectory
```

```
END IF

IF lerrnum = 53 AND lerrsect = 2 THEN
  CLOSE
  PRINT "  Weighting File  "; wtfil$; "  NOT FOUND!"
  RESUME selwgt
END IF

IF lerrnum = 53 AND lerrsect = 3 THEN
  CLOSE
  PRINT "  Task Rating File  "; wtfil$; "  NOT FOUND!"
  RESUME selratfil
END IF

COLOR 7, 0: CLS
PRINT "ERROR NUMBER: "; lerrnum: PRINT

IF lerrnum = 53 AND lerrsect = 5 THEN
  PRINT "  No Weighting Files Found!  -  EXITING PROGRAM!"
  END
END IF

IF lerrnum = 53 AND lerrsect = 6 THEN
  PRINT "  No Task Rating Files Found! - EXITING PROGRAM!"
  END
END IF

PRINT : PRINT "  Press RETURN to RESTART Program from the beginning..."
PRINT "  ESC to Quit"
DO
  ry$ = UCASE$(INKEY$)
  IF ry$ = CHR$(27) THEN END
  IF ry$ = CHR$(13) THEN EXIT DO
LOOP

CLOSE
RESUME begin
```

```
'
' H E A D E R  Sub ---------------------------------------------
'
SUB header

COLOR 7, 1: CLS
LOCATE 1, 72: PRINT "Ver: 2.0";
COLOR 15, 1
LOCATE 2, 15: PRINT "THIS PROGRAM APPLIES WEIGHTINGS TO RATING DATA FILES"
PRINT STRING$(80, 205);
COLOR 7, 1
END SUB
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1992	Technical Memorandum

4. TITLE AND SUBTITLE
The Multi-Attribute Task Battery for Human Operator Workload and Strategic Behavior Research

5. FUNDING NUMBERS
WU 505-64-13

6. AUTHOR(S)
J. Raymond Comstock, Jr. and Ruth J. Arnegard

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23665-5225

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-104174

11. SUPPLEMENTARY NOTES
Comstock: Langley Research Center, Hampton, VA; Arnegard: Old Dominion University, Norfolk, VA

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited

Subject Category 61

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Multi-Attribute Task (MAT) Battery provides a benchmark set of tasks for use in a wide range of laboratory studies of operator performance and workload. The battery incorporates tasks analogous to activities that aircraft crewmembers perform in flight, while providing a high degree of experimenter control, performance data on each subtask, and freedom to use non-pilot test subjects. Features not found in existing computer-based tasks include an auditory communications task (to simulate Air Traffic Control communication), a resource management task permitting many avenues or strategies of maintaining target performance, a scheduling window which gives the operator information about future task demands, and the option of manual or automated control of tasks. Performance data are generated for each subtask. In addition, the task battery may be paused and onscreen workload rating scales presented to the subject. The MAT Battery requires a desktop computer (80286/386/486 processor) with color graphics (at least 640 x 350 pixel). The communications task requires a serial link to a second desktop computer with a voice synthesizer or digitizer card.

14. SUBJECT TERMS
Human performance evaluation; Multiple task environments; Workload evaluation; Synthetic work battery

15. NUMBER OF PAGES
114

16. PRICE CODE
A06

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		