VEHICLE DATA ACQUISITION SYSTEM

OCTOBER 1979

AUTOMATED GUIDEWAY TRANSIT TECHNOLOGY PROGRAM

U.S. DEPARTMENT OF TRANSPORTATION
Urban Mass Transportation Administration
Office of Technology Development and Deployment
Washington, D.C. 20590

This document is available to the U.S. public through
National Technical Information Service,
Springfield, Virginia 22161
NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

The United States Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the object of this report.
A Vehicle Data Acquisition System (VDAS) has been developed by the Port of Seattle which collects and continuously records 20 minutes of data from 34 test points in a specially instrumented Sea-Tac Satellite Transit System vehicle. The recorded data is useful to maintenance personnel to rapidly diagnose intermittent and total vehicle failures, thus reducing vehicle downtime and maintenance costs. This system uses an onboard microprocessor system based on the Intel 8080 and a semiconductor memory to record data which can then be dumped to a microprocessor terminal in the maintenance area for analysis. A six-month in-service test on one vehicle demonstrated that the VDAS can speed up corrective maintenance as well as identify marginal conditions and incipient failures.
TA
1207
H38
c. 2
I. BACKGROUND

A. General

In October, 1976, the Port of Seattle requested Department of Transportation (DOT) funding for a Vehicle Data Acquisition System (VDAS) which was to be used on the Satellite Transit System (STS) at Sea-Tac International Airport. That request was granted in July, 1977 and this document constitutes the final report for the project.

The motivating factor behind the VDAS concerned the amount of time and energy involved in maintaining the operational readiness of the STS vehicles (12 cars of the SLT class). At the time the grant request was made, 75% of maintenance labor was being expended on troubleshooting and only 25% on actual repair. The maintenance group realized that as the passenger volume increased at Sea-Tac, the need for more vehicles would make the system more complex and the system integrity would begin to disintegrate. Normally, during the off peak periods of the year, half the twelve car fleet is required to provide service. Thus, if a car had a known intermittent problem, the luxury exists of keeping that vehicle out of service until the problem is finally isolated. Due to the aforementioned anticipated increase in passenger volume, the Port has already purchased twelve new vehicles for future delivery and is presently running at full capacity. Obviously, the proposed data acquisition system has come at a most convenient time.

B. History

Since 1973, the Sea-Tac International Airport has been operating a subway system providing transportation between the main terminal and the two satellite terminals through two independent underground loops (see Figure 1). A separate shuttle section interconnects the two loops in the main airport terminal.

During the six years of service the maintenance on the transit vehicles has been continually updated and improved. Most mechanical equipment failures could be found and corrected through periodic maintenance procedures. Electronic control malfunctions on the vehicles have always been hard to find due to their dynamic nature. The vehicles are unmanned in normal service and it would be a tedious task to manually monitor a vehicle while it is in that mode. The idea was born to develop a system that monitors the vehicle control logic in the same manner in which a flight recorder assembles data on commercial airliners. The ability to look at the sequence of events occurring prior to the perceived fault would allow technicians to spend more time improving the transit system rather than tracking down intermittent repetitious fault conditions.
C. Project Theory

We began the project with two distinct goals in mind:

1. To significantly reduce the maintenance costs of the STS through decreasing the number of failures and maintaining a high level of availability.

2. To determine if sufficient long-term maintenance cost improvement could be realized by increasing the number of VDAS units in the vehicle fleet.

The VDAS would primarily be used in identifying elusive, intermittent problems by "capturing" critical performance parameters during the most recent twenty minutes of operation. This data would be analyzed, and a picture of the operating system before, during and after failure would be developed.

II. DESIGN CONSIDERATIONS

The STS vehicles are automatically controlled by hardwired wayside controls. A computer system supervises the train operation in the loop, but it does not have to be running for loop control.

The vehicles receive speed and stop commands from the wayside through antennas underneath the vehicle, stopping in alignment with the station door and all interlocking switching for multiple car trains. All of the ATO equipment is located in the vehicles' control consoles at the front and rear (see Figures 2 and 3).

The VDAS ties into these different control circuits at strategic monitor points. These points were carefully selected such that examination of their status at any given time should yield accurate system information directly related to sub-system function, thereby isolating problems to specific sub-systems. The maintenance crew keeps two spare electronic sub-system units on hand at all times. Thus, when the VDAS indicates a faulty unit, it can quickly be replaced resulting in significant reduction in downtime.

Based upon past failure history the Maintenance personnel selected 34 test points for the VDAS (See Appendix A). They consist of four major types of signals:

1. Static relay voltages. All relays feed off the +36 VDC vehicle battery.

2. Static signals between control units. These are digital in nature, alternating between 0VDC and either +6VDC or +15 VDC.
3. Analog signals within a feedback loop. Threshold levels are indicative of different modes of operation.

4. Pulse integrations. Peak detectors and control circuitry for interrupt enabling.

A breakdown of the number of test points located within specific subsystems may be found in Table 1.

Early in the system development, we decided to use as many off-the-shelf products as possible. The Port is not set up for developmental work and, in addition, Port labor costs are too high for that type of activity.

We chose a microprocessor for the VDAS controller and initially wanted to use a cassette tape for transferring the accumulated data. The cassette idea was later scrapped in favor of a microdiskette system for its better reliability and ease of usage. The diskette is not a dedicated unit. When a failure occurs, the maintenance personnel connect the diskette drive to the vehicle VDAS and dump the contents of memory onto the diskette for data reduction and analysis.

A Prolog microprocessor system was chosen primarily for its convenient 19 in. rack mounting which is the vehicle format. (See Figure 2.) Height limitations within the vehicle eliminate most of the existing single board computers. It might be emphasized that any of the commercially available microprocessing systems are capable of performing the tasks necessary were it not for the aforementioned size limitation. If this had been known in advance, the vehicles could probably have been designed to accommodate other systems.

For program development and data formatting, a SOL computer was purchased along with a North Star Disk and accompanying Disk Operating System (DOS) (see Figure 4). 8080 assembly coding and Basic program generation capabilities are part of the SOL software configuration. The 8080 assembly code is used for the actual data acquisition on the vehicle and a Basic language program is resident in the SOL computer for data reduction and formatting.

We chose to consider the acquired data as a single contiguous file which would simplify our data handling procedures. The system was designed such that acquired data would be stored in consecutive memory locations until the top of RAM was reached (see Figure 5 for memory mapping). At that point, the address counter would "roll-over," allowing the newest data to be stored at location 2000, writing over previously stored information. To keep track of the proper sequencing in the data stored, the relative time of occurrence accompanies the data bytes (see Figure 6 for data storage formatting).

We did consider using an Intel development system which would have reduced the software development time. The advantage with development systems is their ability to function in-circuit as the software is being written. For our purposes, this advantage did not outweigh the considerably higher cost. 

18/03
To acquire the data from the vehicle itself, we designed signal-conditioning circuit cards which were compatible with the Prolog mainframe. This resulted in a reliable customized interface suitable for accurate reading of the parameters we were after. For systems requiring a greater number of monitoring points, this may become an unwieldy solution, but for the number of points we were monitoring, it was not an insurmountable task.

One of the problems we ran into interfacing with the STS vehicle concerned ensuring that the VDAS would not interfere with the normal operation of the ATO controls. This was solved through the vehicle contractor, Westinghouse Corp., and resulted in a termination board (see Figure 7) installed between the test points and the interface electronics. With the termination board installed, any type of VDAS circuit violation such as shorted output, or voltage feedback, would not affect the vehicle circuitry.

Another design problem occurred when we decided to use the diskette drive. Having already purchased the Prolog computer, we were faced with the task of interfacing the North Star Disk drive to it. The Prolog System had an inverted data bus and used differing timing signals for its strobing, for example. These problems were solved through the addition of another interface card (Figures 8) which conditioned these signals for acceptance by the disk controller.

Because the test vehicle operates off a non-standard voltage level (36 VDC), we had some difficulty locating a suitable power supply for the VDAS. We needed +5, +15, and -15 VDC. We eventually found a supply with sufficient reliability to satisfy our requirements. Previous supplies had a tendency to burn out during revenue operation, causing considerable delay in the VDAS evaluation.

We wanted to be able to interrupt the VDAS software at any time (i.e., in the event of a system failure) to allow us to dump the memory contents on disk. Also, we knew that a timing relationship was required for the data to have any meaning. To achieve these goals, we developed a hardware priority interrupt system using a Prolog priority interrupt card.

In this way, a real-time clock would generate an interrupt, the interrupt card would determine its priority and branch to the appropriate service routine while holding the other interrupts in a high impedance state until the task was finished. The dump procedure is initiated by depressing a button on the VDAS front panel (see Figure 2).

One consideration that has a direct bearing on the efficacy of the system is the amount of memory required. We needed to store about 20 minutes' worth of vehicle data to allow the Maintenance personnel sufficient time to get to the test vehicle following a failure and initiate the dump procedure. We started with the assumption of 6 data words, which yields one bit for each of 34 monitored conditions plus eight bits for resolving the vehicle speed to 0.1 mph, leaving six expansion bits which are unused at this time. In addition, two words relating to the running real time are required. Referring to Figure 9, this would require 96,000 words of core memory.
This is an obvious problem as most microprocessors have the capability of addressing only 65K of memory, the 8080 included. So we decided to employ the maximum 65K less whatever the controlling software required, but only store the data whenever a change occurred in any given bit of the 6 data words. This has been a workable solution yielding 56K of usable core memory for data storage (main program and related utilities constitute 9K of total memory). This allows about 12 minutes of data storage which has proven sufficient for our needs.

III. EQUIPMENT

A. Hardware

1. SOL Terminal Computer—See Figure 10.
   This equipment was purchased for software development and data analysis. It operates as a stand alone computer under control of a program contained in a personality module which allows the assembled output of developed programs to be compatible with the VDAS CPU, in this case the Intel 8080. The mainframe is designed around the S-100 bus, an industry standard geared for the hobby market. In essence, SOL combines a Central Processing Unit (CPU) with several S-100 peripheral modules-memory, keyboard interface, video interface, audio cassette interface, plus parallel/serial I/O interfaces. The system also includes a five slot backplane board for adding other I/O or memory modules that are S-100 compatible.

   The SOL uses an 8080 as its CPU for control of system functions. The internal structure of the CPU employs 3 separate buses:

   i. 16 bit address bus.
   ii. 8 bit bi-directional data bus.
   iii. 28 bit Control bus.

   The S-100 bus is a composite of these buses and other control signals arranged in a standardized format such that I/O modules can be developed independent of the host system yet with the certainty that they will be operational when installed in the S-100 backplane.

   In addition to the S-100 bus structure, the SOL uses an eight bit keyboard input port, an eight bit parallel input port, an eight bit parallel output port and an eight bit sense switch logic input port.

   A Universal Asynchronous Receiver/Transmitter (UART) is installed for processing serial data from the audio cassette I/O circuits and the serial input line. This is advantageous since the UART would otherwise have to be incorporated as a software module.
All CPU data and address lines are tri-stated to allow multiple device connection. The SOL also has video generation capability in ROM which allows a dot matrix format for video output.

As previously mentioned, we decided to use a diskette for mass storage rather than the audio cassette. We chose a North Star Micro-Disk System for its availability and S-100 bus compatibility. To allow the disk to "talk" to the SOL, we purchased a software package which contained all the I/O routines required to interface the North Star software with the SOL hardware.

This package fit entirely within the 256 byte block North Star has allocated for its I/O area thus taking nothing away from the user's workspace, and without requiring any hardware interfacing.

Aside from the disk unit, one other peripheral was installed in the SOL system, a Practical Automation Printer.

This unit gives us hardcopy of our system software and the data from the VDAS. It has selectable baud rates (up to 1200 bps in serial mode) and uses a 20mA current loop or 8 bit parallel bus for data input.

2. Prolog (See Figure 1)

The basic Prolog System consists of eight printed circuit boards:

i. Priority Interrupt (#8118-1)
ii. I/O (#8113-1)
iii. 32 bit input (#8114)
iv. CPU (#8821)
v. RAM (#8119 - 4 each)

These boards are manufactured by Prolog to fit into a 19 inch wide card rack with the S-100 bus backplane wiring. Their usage is fairly straightforward.

To provide the necessary signal conditioning to the test points prior to inputting them to the Prolog System, the Port built four prototype custom PCB's:

i. 9001 - 36 vdc input card inputs relay logic to system via opto-isolators.
ii. 9002 - CMOS input card. Provides high input impedance and also contains hardware tachometer scaling for determining vehicle speed.
iii. 9003 - OP AMP Analog input card.
iv. 9004 - Control Card.

These cards were wire-wrapped on a standard Prolog utility dip card (#P561) having a 56 pin card edge connector consistent with their off-the-shelf system boards. Details of these custom PCB's are shown in Appendix B.
To allow transferral of acquired data, we purchased a controller from North Star and adapted it for use with the Prolog System (see Figure 8). The controller was configured such that when the vehicle experienced a failure, a technician would connect the disk unit to the Prolog System and press the "DUMP" button (see Figure 2) generating a hardware interrupt, halting the CPU and branching to the memory dump routine. We ran into some initial problems in locating the software properly because the I/O routine required certain locations for its use and we had previously used those same locations. This is definitely something which should be researched before purchasing a similar system.

B. Software

There are two major programs developed for the system:

1. VDAS Data Handler
2. Data Analysis Routine

The SOL computing system, in conjunction with the North Star DOS, was used for program development.

This section will address just the routines developed for the VDAS itself rather than delve into all the necessary bookkeeping procedures peculiar to our particular system. The rationale behind this is that the important information is the manner in which the data was gathered, and how that data was assembled into a form usable by our maintenance group. This information would be valuable to future users, whereas the manner of implementation would not necessarily be of value to a user with a different system, which would probably be the case.

1. VDAS Data Handler

This is an assembly language program written for the Intel 8080 microprocessor. Data from the Westinghouse vehicle is input to the processor every 100 msec. If any data has changed from the last sample, all data is stored in random access memory along with a time reference. If no data has altered, no storage will take place.

A memory wrap-around is provided when the top of memory (FFFF hex - see Figure 5) is reached. The program revitalizes the memory pointer to the beginning of storage (2000 hex) such that "new" data will write over "old" data. In this manner it is possible to get at least twelve (12) minutes worth of data at any given time.

The program is interrupt driven such that when not executing one of the interrupt routines, the CPU is in a halt state with interrupts enabled (the processor automatically disables interrupts upon initiation of an interrupt subroutine). When
an interrupt is received

a "RST" (restart)

instruction vectors the software to the start address of the routine which handles the particular interrupt. The only exception is the "Power-On" Routine which takes advantage of the fact that the processor automatically vectors to address zero when power comes on. Thus, the hardware interrupt procedure provided by the Prolog System is not needed and address zero contains logic appropriate for handling the power-on situation.

The remaining logic is arranged as a continuous assembly of the following interrupt modules:

i. Monitor
ii. Disable
iii. Memory Dump
iv. Reinitialize

The monitor routine (see Appendix C for flow diagrams) stores data from the vehicle into memory. It contains logic to determine whether data has been altered since the last time data was stored. If that is true, the data is stored. A memory limit check is also provided incorporating the memory wraparound feature when memory overflow occurs. It is driven with a clock which provides an interrupt every 100 msec.

The Disable routine captures one last frame of data including the time reference—and then disables the real time clock interrupt to "freeze the memory." It also readies the system for dumping the contents of memory by initializing the appropriate handshaking flags. The time reference is not correlated to Greenwich time. It has meaning only within the VDAS frame of reference.

The memory dump routine outputs the contents of memory from 2000-FFFF hex onto a microdiskette with the oldest data placed at the beginning of the diskette. If the overflow flag has been set, then the current memory pointer points to the oldest data. If overflow has not occurred, then the dump takes place from 2000 hex to the current value of the memory pointer.

The reinitialize routine clears flags, turns off indicators and clears memory of old test data. The real-time clock is re-enabled and the system is once again ready for a new set of data.

There are separate subroutines called by the interrupt routines which accomplish straightforward tasks such as memory limit checking, writing to disk and inputting data from the test vehicle.
All of the interrupt routines are designed to be completed within the interrupt disable timeout which is automatically generated as part of the hardware interrupting procedure. Following completion of their tasks, the routines return to the idle mode with all interrupts fully enabled.

2. Data Analysis

The Data Analysis program is a Basic language program which is used for gathering maintenance information. It is run only on the SOL computer and uses the data dumped out from the VDAS as input. The principle features of this program are:

i. Test point data from the Westinghouse vehicle is displayed in binary format on a CRT or a line printer depending on user selection. The data is identified with appropriate heading and a legend is included at the end of each display for further explanation (see Figure 12 for a representative example).

ii. Three levels of resolution are provided to enable focusing on an area of interest within a considerable amount of data. The lowest level of resolution encompasses the entire data with a scan of 48 frames, which is approximately 1 out of every 150 data points or frames. The medium level of resolution involves a scan between any two points within the low resolution scan. There are 10 frames which is about 1 out of 15 data points. The high resolution scan encompasses every frame within a specified range.

The program is structured such that the operator may go from one level of resolution to another with relative ease. The scan at each resolution may be truncated to add even further flexibility in usage. Also, it is possible to effect a pause in the program by keyboard entry. This enables the user to examine data on the CRT more easily.

iii. The user may choose between a total scan or time-specified scan at the lowest resolution upon entry to the program. The time-specified scan can be used effectively if the user knows approximately the time between when a fault occurred and the "stop monitor" button is depressed. Entering this time will allow the user to truncate the display to the area of interest more quickly.
IV. TEST PERIOD RESULTS

The VDAS data test points were chosen by the maintenance group based upon their previous experience in diagnosing vehicle failures. For instance, there is a point which determines direction (forward or reverse), this point, in conjunction with another point which determines speed (brake, acceleration, maintain speed) can indicate the dynamic performance of the vehicle at any time. If the data indicated that the control logic was in an accelerating reverse mode when it should have been in forward braking, you can immediately narrow your choices to the appropriate areas.

Figure 12 shows a sample Data Analysis Output. The time is referenced to the vehicle data itself, having no relationship to the actual time of occurrence. The time reference does, however, indicate a timing relationship between the actual test points themselves. It is very important to plot the dynamic performance of each test point versus the time of occurrence. It must be remembered that data is stored only when one of the data points alters state. With a sampling rate of 100 ms there often are periods of time upwards of one to three seconds where data is not stored. By plotting events according to their true time of occurrence, one may develop an accurate assessment of how the control logic was behaving during the time a failure occurred.

Appendix A contains the vehicle control schematics showing where each of the test points are located. As can be seen, they tend to follow a certain line of logic from inception to completion. In this manner, the technicians can isolate failures to a given sub-system. If the data shows that a start command was received (test point 3) but that the motors weren't energized (test point 4), then a fault is indicated in the motor control logic.

Appendix D contains data analyses for three separate failures recorded by the VDAS.

The first failure occurred during an "add train" operation where the test vehicle was being placed in revenue operation. The data shows a loss of "code load" 1.8 seconds after the wayside transmitters came on, and only .3 second after the wayside transmitters started sending the D.O.C. word.

The second failure occurred at the main terminal in the South loop. The test vehicle was operating in a two-car configuration and the vehicles were given a command to reverse their direction following a scheduled stop at the main terminal.

In both cases, it appears that the interference from the wayside Info Transmission prevented speed commands from being decoded. This is evidenced by the intermittent nature of code loaded after the A relay was reset. This is due to jamming of the speed code signal by the wayside information signal.
The third failure occurred during normal operation as the vehicle was attempting to accelerate from 15 to 20 mph. The lack of an overspeed failure (test points 31 and 32) and presence of a balance failure (test point 33) and an A-relay dropout 1 second later suggests a ramp failure or tach failure. Specifically, "+ ramp" is suspect because of a speed decrease (test point 20) and is one of the two servo inputs, the other being "Tach 1." The 20 mph crystal oscillator which generates the "+ ramp" signal probably didn't start oscillating. The stability of code loaded (test point 34) from the speed decoder does not suggest a failure in the decode circuitry.

In addition to these examples, we found many other faults through a combination of VDAS and troubleshooting techniques. For example, the vehicle stopped on a reset from the wayside controller. Upon examination of the VDAS data we discovered that Tach 2 (test point 21) had dropped out for no apparent reason. (Tach 1 & 2 are identical circuits backing each other up. We determine speed from Tach 1 and simply detect the absence or presence of Tach 2.) The technicians immediately examined the Tach 2 signal and found that the waveform was rounded showing 5MHZ noise on the edges. The Tach 2 circuitry was replaced and the car re-entered revenue operation after being out of service a mere three minutes.

Another advantage we have found in using the VDAS concerns the isolation of recurring trouble spots. Quite often, a failure is caused by noise on the vehicle send and receive circuitry. VDAS will pinpoint this problem and document it in such a fashion as to convince management that it is worthwhile to re-design the antennas or associated circuitry causing the problem. In this manner, the VDAS is used as a tool to reduce the amount of time spent on troubleshooting and increase the man-hours expended on improving system efficiency.

V. CONCLUSION AND RECOMMENDATIONS

The maintenance group is thoroughly convinced of the efficacy of the VDAS. They only wish they had more test units available. This would be the prime recommendation arising from the project. We configured one test vehicle to determine if the system had value and wanted to ascertain whether installation of the VDAS on all vehicles would be cost effective. Both goals have been met. The VDAS pinpoints vehicle problems, and is the only way we know of which will provide maintenance with dynamic system information. We have seen problems on the test vehicle reflected in other vehicles, but have no way of knowing if the other vehicle is exhibiting identical symptoms. Yet, even from information gleaned from the single test vehicle, we can narrow the guesses to a manageable few in the instances where similar problems appear in other vehicles.

The VDAS should be viewed as a tool, not a panacea to all system problems. While it is possible to monitor a great many testpoints through expansion, it is felt that this approach would only serve to confuse the issue which is to significantly reduce vehicle downtime. This end result is better served by carefully selecting testpoints according to their own inherent value within the system. This, in conjunction with highly trained personnel, can reduce that downtime through failure analyses conducted with a systematic approach provided by using the VDAS as a diagnostic tool. The results from the data analysis must be viewed from the systems viewpoint.
It is important that personnel performing the analysis know exactly what should be happening at any given time. The VDAS can indicate something is awry, but it takes the skill of a trained technician to interpret that clue properly.

We are presently forging ahead with the task of incorporating VDAS into all of our vehicles. It is not necessary to have a computer in each vehicle. We feel that by configuring each vehicle such that a computing unit can easily be installed and removed as an individual vehicle exhibits an intermittent problem we can reduce the total system cost while maintaining a high level of system integrity. Our plans call for three or four computing units for a total fleet of 24 vehicles.

It is also important to keep accurate records of system problems such that an empirical database may be constructed. This is especially true in the case of those problems which are intermittent. They may occur only once every week or less, but with proper record keeping, enough information will be gathered such that the solution will eventually come to light. These records can also be very helpful in training new personnel and keeping present personnel up-to-date on the vehicle electronics.
FIGURE 1

SEA-TAC INTERNATIONAL AIRPORT
SATELLITE TRANSIT SYSTEM
FIGURE 2
Picture of VDAS Control Circuitry
FIGURE 3
Picture of VDAS Termination Board
FIGURE 4
Picture of SOL Computer with North Star Disk
<table>
<thead>
<tr>
<th>PROLOG</th>
<th>SOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>Interrupt service</td>
<td>AL-8 portion of AL9-6</td>
</tr>
<tr>
<td>Initialize routine</td>
<td>DOS</td>
</tr>
<tr>
<td>Power on Initialize</td>
<td>28FF</td>
</tr>
<tr>
<td>Power on routine</td>
<td>2A00</td>
</tr>
<tr>
<td>00FF</td>
<td>0100</td>
</tr>
<tr>
<td>0100</td>
<td>017F</td>
</tr>
<tr>
<td>Power on routine</td>
<td>Re-initialize routine</td>
</tr>
<tr>
<td>017F</td>
<td>0277</td>
</tr>
<tr>
<td>Input routine</td>
<td>Dump routine</td>
</tr>
<tr>
<td>0277</td>
<td>037F</td>
</tr>
<tr>
<td>0280</td>
<td>03FF</td>
</tr>
<tr>
<td>Re-initialize routine</td>
<td>Disable routine</td>
</tr>
<tr>
<td>02FF</td>
<td>03FF</td>
</tr>
<tr>
<td>0300</td>
<td>0400</td>
</tr>
<tr>
<td>Dump routine</td>
<td>Unused</td>
</tr>
<tr>
<td>037F</td>
<td>0400</td>
</tr>
<tr>
<td>0380</td>
<td>08FF</td>
</tr>
<tr>
<td>Disable routine</td>
<td>Subroutines: STWD1</td>
</tr>
<tr>
<td>03FF</td>
<td>08FF</td>
</tr>
<tr>
<td>0400</td>
<td>0C00</td>
</tr>
<tr>
<td>Unused</td>
<td>Subroutines: STWD1</td>
</tr>
<tr>
<td>08FF</td>
<td>0C00</td>
</tr>
<tr>
<td>0C00</td>
<td>Subroutines: STWD1</td>
</tr>
<tr>
<td>0DFF</td>
<td>Disk write routine</td>
</tr>
<tr>
<td>0E00</td>
<td>5FF</td>
</tr>
<tr>
<td>0OFF</td>
<td>6000</td>
</tr>
<tr>
<td>1000</td>
<td>Disk write routine</td>
</tr>
<tr>
<td>RAM</td>
<td>User memory</td>
</tr>
<tr>
<td>17FF</td>
<td>User memory</td>
</tr>
<tr>
<td>1800</td>
<td>(RAM)</td>
</tr>
<tr>
<td>Disk boot</td>
<td>(partitioning not to scale)</td>
</tr>
<tr>
<td>1FFF</td>
<td>BFFF</td>
</tr>
<tr>
<td>2000</td>
<td>SOLOS ROM</td>
</tr>
<tr>
<td>User memory</td>
<td>SOLOS ROM</td>
</tr>
<tr>
<td>(RAM)</td>
<td>SOLOS RAM</td>
</tr>
<tr>
<td></td>
<td>CBFF</td>
</tr>
<tr>
<td></td>
<td>CC00</td>
</tr>
<tr>
<td></td>
<td>CBFF</td>
</tr>
<tr>
<td></td>
<td>CC00</td>
</tr>
<tr>
<td></td>
<td>CFFF</td>
</tr>
<tr>
<td></td>
<td>D000</td>
</tr>
<tr>
<td></td>
<td>E7FF</td>
</tr>
<tr>
<td></td>
<td>E800</td>
</tr>
<tr>
<td></td>
<td>EBFF</td>
</tr>
<tr>
<td></td>
<td>EC00</td>
</tr>
<tr>
<td></td>
<td>FFFF</td>
</tr>
</tbody>
</table>

**PORT OF SEATTLE COMMISSION**

**PROJECT** STS-VDAS

**TITLE** MEMORY MAP: PROLOG & SOL SYSTEMS

**DRAWN BY** A. HAUG **DATE** APPROVED

**FIGURE 5**
### Bit Assignment Table

<table>
<thead>
<tr>
<th>IN '00</th>
<th>TP-7H</th>
<th>TP-7L</th>
<th>TP-6</th>
<th>TP-5</th>
<th>TP-4</th>
<th>TP-3</th>
<th>TP-2</th>
<th>TP-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR</td>
<td>DOC</td>
<td>DTX</td>
<td>MDL</td>
<td>NC2</td>
<td>FP4</td>
<td>PR</td>
<td>LS</td>
<td></td>
</tr>
<tr>
<td>TP-17H</td>
<td>TP-17L</td>
<td>TP-16H</td>
<td>TP-16L</td>
<td>TP-11H</td>
<td>TP-14L</td>
<td>TP-9</td>
<td>TP-8</td>
<td></td>
</tr>
<tr>
<td>TP-18H</td>
<td>TP-18L</td>
<td>A REL</td>
<td>TCH 2</td>
<td>TP-19H</td>
<td>TP-19L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP-24</td>
<td>TP-23</td>
<td>TP-22</td>
<td>TP-21</td>
<td>TP-21</td>
<td>TP-18H</td>
<td>TP-18L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B/P</td>
<td>UPD</td>
<td>A REL</td>
<td>TCH 2</td>
<td>A REL</td>
<td>TCH 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP-30</td>
<td>TP-31</td>
<td>TP-29</td>
<td>TP-28</td>
<td>TP-27</td>
<td>TP-26</td>
<td>TP-25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P SIG</td>
<td>OSII</td>
<td>DCMD</td>
<td>DSEN</td>
<td>CB</td>
<td>CA</td>
<td>PC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP-34</td>
<td>TP-33</td>
<td></td>
<td></td>
<td>BAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP-20</td>
<td>BIT6</td>
<td>TP-20</td>
<td>BIP7</td>
<td>BIT6</td>
<td>BIT5</td>
<td>BIT4</td>
<td>BIT3</td>
<td></td>
</tr>
<tr>
<td>BUSY</td>
<td>(128)</td>
<td>BIT7</td>
<td>(64)</td>
<td>BIT6</td>
<td>(256)</td>
<td>BIT4</td>
<td>(2)</td>
<td></td>
</tr>
<tr>
<td>TACH #1</td>
<td></td>
<td>(32)</td>
<td></td>
<td>(16)</td>
<td>(8)</td>
<td>(4)</td>
<td>(2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(17)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### OUT '06

<table>
<thead>
<tr>
<th>Device</th>
<th>Bit Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT USED</td>
<td>47</td>
</tr>
<tr>
<td>DISABLE TIME INTERRUPT</td>
<td>57</td>
</tr>
<tr>
<td>COMPLETE DUMP INDICATOR</td>
<td>35</td>
</tr>
<tr>
<td>ERROR INDICATOR</td>
<td>27</td>
</tr>
<tr>
<td>NOT USED</td>
<td>17</td>
</tr>
<tr>
<td>DUMPING INDICATOR</td>
<td>13</td>
</tr>
<tr>
<td>READY TO DUMP</td>
<td>27</td>
</tr>
<tr>
<td>MONITOR INDICATOR</td>
<td>11</td>
</tr>
</tbody>
</table>

---

**Port of Seattle Commission**

**Project:** STS - VDAS

**Title:** BIT ASSIGNMENT IN PROLOG SYSTEM

**Drawn by:** P. Stutz

**Date:** 2/24/79

**Approved:** 3/14/78

**Figure 6**
74367 74368

DATA LINES

STALL
BOARD READ

STANDARD INTERFACE
ADAPTED FOR PROLOG

PORT OF SEATTLE COMMISSION

PROJECT STS-VDAS
TITLE ADAPT NORTHSTAR-DISK-I/O TO PROLOG
DRAWN BY PETE STUTZ
DATE 3/16/78
APPROVED

FIGURE 8b
IV: Memory Amount (# of 8 bit words)
S: Sample rate (sec)
T: Length of time which may be stored using M & S above (min)
B: Word length (bits)

The following equation may be used to determine core memory size:

\[
M = \frac{BT \times 60 \text{ sec}}{S} = \frac{60BT}{S}
\]

In our system, the following are given:

\[
M = 56K = 57,344 \quad (1K = 1024)
\]
\[
S = .1 \text{ sec}
\]
\[
B = 8 \text{ bits}
\]

\[
T = \frac{MS}{60B} = \frac{(57344)(.1)}{60(8)} = 11.95 \text{ min}
\]

FIGURE 9
Core Memory Size Calculation
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LINE SW CLOSED</td>
</tr>
<tr>
<td>2</td>
<td>PR RELAY CLOSED</td>
</tr>
<tr>
<td>3</td>
<td>FP RELAY CLOSED</td>
</tr>
<tr>
<td>4</td>
<td>MAIN CONTACTOR CLOSED</td>
</tr>
<tr>
<td>5</td>
<td>MOTOR OVERLOAD</td>
</tr>
<tr>
<td>6</td>
<td>DOC TX TO WAYSIDE</td>
</tr>
<tr>
<td>7</td>
<td>ANY DOC-VEH.OR INFO.RX</td>
</tr>
<tr>
<td>8</td>
<td>BIT RATE + DOC</td>
</tr>
<tr>
<td>9</td>
<td>FLIN + REL</td>
</tr>
<tr>
<td>10</td>
<td>LOW BRAKE CLAMP</td>
</tr>
<tr>
<td>11</td>
<td>ZERO SPEED</td>
</tr>
<tr>
<td>12</td>
<td>REL FROM INFO RX</td>
</tr>
<tr>
<td>13</td>
<td>PROP</td>
</tr>
<tr>
<td>14</td>
<td>HARD PROP</td>
</tr>
<tr>
<td>15</td>
<td>PROGRAM STOP BRAKING</td>
</tr>
<tr>
<td>16</td>
<td>MAX. CONTROL BRAKING</td>
</tr>
<tr>
<td>17</td>
<td>PROP (XP SIG.)</td>
</tr>
<tr>
<td>18</td>
<td>BRAKING (XP SIG.)</td>
</tr>
<tr>
<td>19</td>
<td>PROP (AP SIG.)</td>
</tr>
<tr>
<td>20</td>
<td>&quot;</td>
</tr>
<tr>
<td>21</td>
<td>TACH 1 SPEED</td>
</tr>
<tr>
<td>22</td>
<td>TACH 2 &gt; 2.5 VOLTS</td>
</tr>
<tr>
<td>23</td>
<td>A RELAY PULLED IN</td>
</tr>
<tr>
<td>24</td>
<td>UPDATE RECEIVED</td>
</tr>
<tr>
<td>25</td>
<td>PROP</td>
</tr>
<tr>
<td>26</td>
<td>PCS TX ON</td>
</tr>
<tr>
<td>27</td>
<td>COUP. #1 END</td>
</tr>
<tr>
<td>28</td>
<td>COUP. #2 END</td>
</tr>
<tr>
<td>29</td>
<td>FWD SENSE FROM REVERSER</td>
</tr>
<tr>
<td>30</td>
<td>FWD + CAR AHEAD</td>
</tr>
<tr>
<td>31</td>
<td>P SIG. INTERLOCK</td>
</tr>
<tr>
<td>32</td>
<td>TACH 1 FAULT</td>
</tr>
<tr>
<td>33</td>
<td>TACH 2 FAULT</td>
</tr>
<tr>
<td>34</td>
<td>BALANCE FAULT</td>
</tr>
<tr>
<td>35</td>
<td>CODE LOAD</td>
</tr>
</tbody>
</table>

**FIGURE 12a**

Data Analysis Overlay Showing Bit Placement
Sample Test Data Analysis Output
<table>
<thead>
<tr>
<th>Sub-system</th>
<th># TP's</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car Logic</td>
<td>1</td>
</tr>
<tr>
<td>Traction Controller</td>
<td>3</td>
</tr>
<tr>
<td>Brake/Propulsion</td>
<td>4</td>
</tr>
<tr>
<td>Speed Maintaining</td>
<td>7</td>
</tr>
<tr>
<td>Speed Decoder</td>
<td>1</td>
</tr>
<tr>
<td>Information Receiver</td>
<td>3</td>
</tr>
<tr>
<td>Vehicle Transmitter</td>
<td>1</td>
</tr>
<tr>
<td>Train Line Wire</td>
<td>5</td>
</tr>
<tr>
<td>AC Power Controls</td>
<td>5</td>
</tr>
</tbody>
</table>

**TABLE 1**

Four test points not shown deal with the vehicle doors. These points were later determined to be so obvious as to not require system monitoring (ie. doors are either open or closed).
APPENDIX A

Test Point Drawings
VDAS

BASIC GUIDELINES FOR INTERCONNECTION

- ALL 36VDC SIGNAL THRU OPTICAL ISOLATORS

- NO PHYSICAL CONNECTION BETWEEN BATTERY
  GROUND AND SYSTEM GROUND

- ALL INTERCONNECTIONS ON TERMINAL BLOCKS,
  EITHER ALREADY EXISTING, SPARE TERMINALS
  OR TB-3 (TOTALLY UNUSED SO FAR);
  TB3-55/56/57/58 USED FOR RECENT MODIFICATION

- USE SPARE CIRCUITRY ON BOARDS WHERE
  AVAILABLE (UNUSED BUFFERS ETC.)

- NO CHANGE TO EXISTING CIRCUIT BOARDS UNLESS
  ABSOLUTELY NO OTHER CHOICE

- VDAS CRADLE, 104-PIN CONNECTOR NUMBERS ARE DESIGNATED
  WITH "#" AND PIN NUMBER.

REV C 7/20/78
REV B 4/3/76
REV A 12/13/77

PORT OF SEATTLE COMMISSION
PROJECT STS - VDAS TESTPOINT DEFINITION
TITLE GUIDELINES FOR INTERCONNECT
DRAWN BY P. STUTZ DATE 11-9-77 APPROVED

DESIGN NO.
DRAWING NO.
PORT 120-9
SHT 1/327
CONTACT IS BYPASSED
BY JUMPER FROM
TB 11-2 TO TB 11-4
MAKE AVAILABLE BY
REMOVING JUMPER

REF: VEHICLE SCHEMATIC
W 3907C II - SHT 11

PORT OF SEATTLE COMMISSION

PROJECT: STS - VDAS TESTPOINT DEFINITION
TITLE: TESTPOINT # 4 - MAIN CONTACTOR
DRAWN BY: P. STUTZ

DESIGN NO.:

DRAWING NO.: LCD P-330

REV C 7/20/78
REV B 4/3/78

SHT 5 (32)
USE TB3-3 DOOR LEAF #1
TB3-4 DOOR LEAF #2
TB3-5 DOOR LEAF #3
TB3-6 DOOR LEAF #4

REV C 7/20/76
REV B 4/3/76
WILL NOT BE USED IN VDAS PER MIKE KISMAN
REV A 12/9/77

PORT OF SEATTLE COMMISSION
PROJECT STS - VDAS TESTPOINT DEFINITION
TITLE TESTPOINTS 12/13/14/15 - DOOR LEAF OPEN LIMIT SWITCH
DRAWN BY P. STUTZ DATE 11/8/77
APPROVED

FOR 120-9

SHT 13(32)
TO TACH 1

S/M CRADLE 1117F10

TERMINATION BOARD

VDAS CUSTOM INTERFACE

102 Hz / MPH

THRESHOLD SET AT 2.5V PEAK
AT TBI-53

GATE COUNTER AND ENTER COUNT AS 8-BIT NUMBER INTO COMPUTER
ONE-SHOT CIRCUIT FOR SIGNAL CONDITIONING

INFO ONLY

AT TACH DETECTOR
THRESHOLD 2.1V

REV C 7/10/73
REV B 4/3/73
REV A 12/9/77

PORT OF SEATTLE COMMISSION

PROJECT STS - VDAS TESTPOINT DEFINITION

TITLE TESTPOINT #20 - TACH #1

DRAWN BY P. STUTZ DATE 11-2-73 APPROVED

SHT 18 (327)
PORT OF SEATTLE COMMISSION

PROJECT: STS - VDAS TESTPOINT DEFINITION

TITLE: TESTPOINT #21 - TACH #2

DRAWN BY: P. STUTZ  DATE: 11-2-77  APPROVED

REV C  7/20/79
REV B  4/3/78
REV A  12/19/77

DESIGN NO.

DRAWING NO.

LCO P-330

SHT 19 (32)
TLW OVERALL SCHEMATIC
3964.C50 - SHT 2

TERMINATION BOARD
VDAS CUSTOM INTERFACE
OPTION:
1. CURRENT TRANSFORMER
2. TUNING CAPACITOR  - IN USE

PORT OF SEATTLE COMMISSION
PROJECT
STS - VDAS TESTPOINT DEFINITION

TITLE
TESTPOINT #25 - VEHICLE TX, TX CURRENT

DRAWN BY
P. STUTZ
DATE
11-7-77
APPROVED

REV C 7/20/78
REV B 4/3/78

DESIGN NO.

DRAWING NO.
LCO P - 330
SHT 23 (32)
PORT OF SEATTLE COMMISSION

PROJECT STS - VDAS TESTPOINT DEFINITION

TITLE TESTPOINT #29 - DIRECTION COMMAND (FWD)

DRAWN BY P. STULTZ DATE 11-7-77 APPROVED

DESIGN NO. LCO P-230

DRAWING NO. SHT 23 (327)
CUT EXISTING CIRCUIT BOARD LAND, ADD PIGGYBACK BOARD FOR OPTO-ISOLATOR

PORT OF SEATTLE COMMISSION

<table>
<thead>
<tr>
<th>PROJECT</th>
<th>DESIGN NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>STS - VDAS TESTPOINT DEFINITION</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TITLE</th>
<th>DRAWING NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>TESTPOINT #32 OS I OVERSPEED TACH 2</td>
<td>LCD P-330</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DRAWN BY</th>
<th>APPROVED</th>
</tr>
</thead>
<tbody>
<tr>
<td>P. STUTZ</td>
<td>11-8 77</td>
</tr>
</tbody>
</table>

SHT 3D (327)
CUT EXISTING CIRCUIT BOARD LAND, ADD PIGGY-BACK BOARD FOR OPTO ISOLATOR.

PORT OF SEATTLE COMMISSION
PROJECT STS - VDAS TESTPOINT DEFINITION
TITLE TESTPOINT #33 - BAL / BALANCE TACH1 VS. TACH 2
DRAWN BY P. STUTZ
DATE 11-8-77
APPROVED

REV C 7/20/76
REV B 4/3/78

DESIGN NO. LCO P-330
DRAWING NO. SHT 31(32)
APPENDIX B

Prolog System Information
**NOTE**

* R12, R13 = 3.3k/1W
  All others = 7.5k/1W

D1 to D16 are the input portions of 16 optical couplers.

PORT OF SEATTLE COMMISSION

PROJECT: STS- VDAC

TITLE: ISOLATED INPUT CARD (36VDC - TTL LEVEL)

DRAWN BY: P. STUTZ  DATE: 12/14/77  APPROVED

**DRAWING NO.**

**DESIGN NO.**
PORT OF SEATTLE COMMISSION

PROJECT STS - VDAC

TITLE ISOLATED INPUT CARD (3.0VDC - TTL LEVEL)

DRAWN BY P. STUTZ

DATE 10/19/77

APPROVED
PORT OF SEATTLE COMMISSION

PROJECT PROLOG UTILITY DIP CARD

TITLE HI-Z INPUT, SPEED INDICATOR

DRAWN BY P. STUZ
DATE APPROVED

PROLOG PART NUMBER # P561

GRID SPACING .5 IN.
GRID SPACING .3 IN.

COMPONENT SIDE

1M 3.3M 10K 18K
1 2 3 4
MC14050 MC14050 MC14050 10K
MC14025 MC14025 MC14025 18K
MC14013 18K
MC140510 MC140510 MC140510
MC14024 MC14024 MC14024
MC14092 JUMPER JUMPER JUMPER
7446 7446 7446 7446
18K
TP420
+3VPEAK
102Hz/MPH

TP421
+3VPEAK
102Hz/MPH

TP420
Fast Time

TP471
Time constant long enough to maintain Q
TP#420
Short Time Contact

TP#421
Time Constant long enough to maintain A

100K

MC14523B

100K

45V

MC14523B

100K

45V

MC14523B

100K

45V
TP420
Short Time
Gate

TP421
Time constant long enough to maintain Q
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>RES</td>
<td>7447</td>
</tr>
<tr>
<td>4</td>
<td>NA47</td>
<td></td>
<td>7450</td>
</tr>
<tr>
<td>3</td>
<td>R/C/D</td>
<td></td>
<td>R/C/D</td>
</tr>
<tr>
<td>2</td>
<td>NA747</td>
<td>RES</td>
<td>NA747</td>
</tr>
<tr>
<td>1</td>
<td>7407</td>
<td>7432</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Control Card

Misc:
see slide 3(3) for changes
TP of t=30

MC 14050A

+5

+15

-15

COM

MC 14050B

ground pin 11,14
NOTE:
ALL RESISTORS - 1/4W
OP. AMPS - 747
BUFFERS - MC 1405B

CURRENT TRANSFORMER.
PORT OF SEATTLE COMMISSION

PROJECT STS - VDAS

TITLE LOCATION OF VDAS IN VEHICLE (#2 END)

DRAWN BY P. STUTZ DATE 11-29-77 APPROVED

DESK 111-1 (6) DRAWING NO.
<table>
<thead>
<tr>
<th>TESTPOINT #</th>
<th>TERMINAL #</th>
<th>CRADLE CONN.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TB5 - 46</td>
<td>J1 - 1</td>
</tr>
<tr>
<td>2</td>
<td>TB4 - 54</td>
<td>J1 - 2</td>
</tr>
<tr>
<td>3</td>
<td>TB4 - 55</td>
<td>J1 - 3</td>
</tr>
<tr>
<td>4</td>
<td>TB3 - 1</td>
<td>J1 - 4</td>
</tr>
<tr>
<td>5</td>
<td>TB5 - 21</td>
<td>J1 - 5</td>
</tr>
<tr>
<td>6</td>
<td>TB6 - 18</td>
<td>J1 - 34</td>
</tr>
<tr>
<td>7</td>
<td>TB7 - 3</td>
<td>J1 - 6</td>
</tr>
<tr>
<td>8</td>
<td>TB1 - 34</td>
<td>J1 - 30</td>
</tr>
<tr>
<td>9</td>
<td>TB2 - 12</td>
<td>J1 - 31</td>
</tr>
<tr>
<td>10</td>
<td>TB2 - 14</td>
<td>J1 - 35</td>
</tr>
<tr>
<td>11</td>
<td>TB5 - 29</td>
<td>J1 - 36</td>
</tr>
<tr>
<td>12</td>
<td>TB3 - 3</td>
<td>J1 - 7</td>
</tr>
<tr>
<td>13</td>
<td>TB3 - 4</td>
<td>J1 - 8</td>
</tr>
<tr>
<td>14</td>
<td>TB3 - 5</td>
<td>J1 - 9</td>
</tr>
<tr>
<td>15</td>
<td>TB3 - 6</td>
<td>J1 - 10</td>
</tr>
<tr>
<td>16</td>
<td>TB5 - 37</td>
<td>J1 - 68</td>
</tr>
<tr>
<td>17</td>
<td>TB5 - 38</td>
<td>J1 - 69</td>
</tr>
<tr>
<td>18</td>
<td>TB3 - 15</td>
<td>J1 - 70</td>
</tr>
<tr>
<td>19</td>
<td>TB4 - 11</td>
<td>J1 - 71</td>
</tr>
<tr>
<td>20</td>
<td>TB1 - 11</td>
<td>J1 - 72</td>
</tr>
<tr>
<td>21</td>
<td>TB5 - 38</td>
<td>J1 - 74</td>
</tr>
<tr>
<td>22</td>
<td>TB1 - 53</td>
<td>J1 - 76</td>
</tr>
<tr>
<td>23</td>
<td>TB1 - 51</td>
<td>J1 - 77</td>
</tr>
<tr>
<td>24</td>
<td>TB1 - 55</td>
<td>J1 - 82</td>
</tr>
<tr>
<td>25</td>
<td>TB1 - 57</td>
<td>J1 - 83</td>
</tr>
<tr>
<td>26</td>
<td>TB2 - 8</td>
<td>J1 - 84</td>
</tr>
<tr>
<td>27</td>
<td>TB3 - 16</td>
<td>J1 - 85</td>
</tr>
<tr>
<td>28</td>
<td>TB4 - 7</td>
<td>J1 - 86</td>
</tr>
<tr>
<td>29</td>
<td>TB3 - 18</td>
<td>J1 - 87</td>
</tr>
<tr>
<td>30</td>
<td>TB3 - 19</td>
<td>J1 - 88</td>
</tr>
<tr>
<td>31</td>
<td>TB5 - 39</td>
<td>J1 - 12</td>
</tr>
<tr>
<td>32</td>
<td>TB6 - 44</td>
<td>J1 - 13</td>
</tr>
<tr>
<td>33</td>
<td>TB5 - 45</td>
<td>J1 - 14</td>
</tr>
<tr>
<td>34</td>
<td>TB3 - 24</td>
<td>J1 - 15</td>
</tr>
<tr>
<td>35</td>
<td>TB3 - 25</td>
<td>J1 - 62</td>
</tr>
<tr>
<td>36</td>
<td>TB3 - 26</td>
<td>J1 - 63</td>
</tr>
<tr>
<td>37</td>
<td>TB3 - 29</td>
<td>J1 - 64</td>
</tr>
</tbody>
</table>

**PORT OF SEATTLE COMMISSION**

**PROJECT** STS - YDAS - CAR WIRING

**TITLE** TERMINAL TO YDAS WIRING - TESTPOINTS

**DRAWN BY** P. STUTZ

**DATE** 1-29-77

**RESPONSE** 120-9

**APPROVED**

**DESIGN NO.**

**DRAWING NO.** SK 7711 - 2/8

*SHIELDED PAIR SHIELD ONLY TERMINATED AT TB3*
<table>
<thead>
<tr>
<th>DESIGNATION</th>
<th>TERMINAL</th>
<th>CRADLE CONN.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GND</td>
<td>TB3-54</td>
<td>J1-39</td>
</tr>
<tr>
<td></td>
<td>-55</td>
<td>J1-40</td>
</tr>
<tr>
<td></td>
<td>-56</td>
<td>J1-52</td>
</tr>
<tr>
<td></td>
<td>-57</td>
<td>J1-54</td>
</tr>
<tr>
<td></td>
<td>-58</td>
<td>J1-55</td>
</tr>
<tr>
<td>+5VDC</td>
<td>-46</td>
<td>J1-50</td>
</tr>
<tr>
<td></td>
<td>-47</td>
<td>J1-51</td>
</tr>
<tr>
<td></td>
<td>-48</td>
<td>J1-53</td>
</tr>
<tr>
<td></td>
<td>-49</td>
<td>J1-56</td>
</tr>
<tr>
<td></td>
<td>-50</td>
<td>J1-67</td>
</tr>
<tr>
<td>-15VDC</td>
<td>-45</td>
<td>J1-95</td>
</tr>
<tr>
<td></td>
<td>-44</td>
<td>J1-96</td>
</tr>
<tr>
<td></td>
<td>-43</td>
<td>J1-97</td>
</tr>
<tr>
<td>COM</td>
<td>-42</td>
<td>J1-98</td>
</tr>
<tr>
<td></td>
<td>-41</td>
<td>J1-99</td>
</tr>
<tr>
<td>+15VDC</td>
<td>-40</td>
<td>J1-100</td>
</tr>
<tr>
<td></td>
<td>-39</td>
<td>J1-101</td>
</tr>
<tr>
<td></td>
<td>TB3-32</td>
<td>J1-102</td>
</tr>
<tr>
<td>TERMINAL TB3-</td>
<td>CRADLE CONNECTOR J1-</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>81</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>83</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>91</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>49</td>
<td></td>
</tr>
</tbody>
</table>

GND

+5VDC

PORT OF SEATTLE COMMISSION

PROJECT STS - VDAS - CAR WIRING
TITLE TERMINAL TO VDAS WIRING - SPARES, ETC.
DRAWN BY P. STUTZ DATE 11-24-77 APPROVED

DESIGN NO.

DRAWING NO. SK 7711-4 (C)
<table>
<thead>
<tr>
<th>TB 3</th>
<th>DESIGNATION</th>
<th>NOTE: &quot;S&quot; - SPARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1-4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>J1-7</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>J1-9</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>J1-20</td>
<td>4</td>
<td>&quot;S&quot; - 15</td>
</tr>
<tr>
<td>J1-21</td>
<td>5</td>
<td>&quot;S&quot; - 22</td>
</tr>
<tr>
<td>J1-23</td>
<td>6</td>
<td>&quot;S&quot; - 29</td>
</tr>
<tr>
<td>J1-25</td>
<td>7</td>
<td>&quot;S&quot; - 10</td>
</tr>
<tr>
<td>J1-26</td>
<td>8</td>
<td>&quot;S&quot; - 23</td>
</tr>
<tr>
<td>J1-27</td>
<td>9</td>
<td>&quot;S&quot; - 25</td>
</tr>
<tr>
<td>J1-48</td>
<td>10</td>
<td>25 &quot;S&quot;</td>
</tr>
<tr>
<td>J1-58</td>
<td>11</td>
<td>30 &quot;S&quot;</td>
</tr>
<tr>
<td>J1-64</td>
<td>12</td>
<td>(COM) SPARE</td>
</tr>
<tr>
<td>J1-66</td>
<td>13</td>
<td>(COM) SPARE</td>
</tr>
<tr>
<td>J1-67</td>
<td>14</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-69</td>
<td>15</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-83</td>
<td>16</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-85</td>
<td>17</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-87</td>
<td>18</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-89</td>
<td>19</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-91</td>
<td>20</td>
<td>SPARE</td>
</tr>
<tr>
<td>J1-101</td>
<td>21</td>
<td>+15VDC</td>
</tr>
<tr>
<td>J1-99</td>
<td>22</td>
<td>COM</td>
</tr>
<tr>
<td>J1-97</td>
<td>23</td>
<td>-15VDC</td>
</tr>
<tr>
<td>J1-51</td>
<td>24</td>
<td>+5VDC</td>
</tr>
<tr>
<td>J1-66</td>
<td>25</td>
<td>GND</td>
</tr>
</tbody>
</table>

PORT OF SEATTLE COMMISSION

PROJECT STS - VDAS -
TITLE VEHICLE TERMINAL BLOCK ASSIGNMENT
DRAWN BY P. STUTZ DATE 11-29-77
APPROVED

DESIGN NO. SK 7711 - 5 (6)
PORT OF SEATTLE COMMISSION

PROJECT STS - VDA6 - CRADLE CHASSIS CONNECTOR

TITLE PIN NUMBERS

DRAWN BY P. STUTZ  DATE 11-22-77  APPROVED

VIEW TOWARDS CRADLE CONNECTOR PINS.
CRADLE WIRING TO BE AT REAR, WIRE WRAP
PORT OF SEATTLE COMMISSION

PROJECT  STS - VDAS - VEHICLE CARD CAGE

TITLE  ASSEMBLY DETAILS  FRONT VIEW

DRAWN BY  P. STUTZ  DATE  11-16-77  APPROVED

AMP CONNECTOR
CTD WESTINGHOUSE TERMINAL BLOCKS
CONNECTOR FACES TO REAR!
MCA255 (3 PLACES)

P

J2-54

0V

+15V

-15V

LED3-ANODE

LED1-ANODE

LED2-ANODE

J2-40

J2-103

J2-39

+15

-15

1K

1K

2N4856

PIGGY BACK BOARD

MOUNTED ON PC BOARD 209P432G01

SPEED MAINTAINING CRADLE #1117F10

NOTE:
* - ADD TO CRADLE WIRING

REF: TESTPOINT DEFINITIONS-REVA -12/9/77

PORT OF SEATTLE COMMISSION

PROJECT  STS-VDAS

TITLE  PIGGY BACK BOARD TO OVERSPD DET #209P432G01

DRAWN BY  P. STUTZ  DATE  2/7/74

FOR 120-9
APPENDIX C

VDAS Data Handler & Data Analysis Software Listings
3. NO
   \[ R_2 = \Phi \]  
   \[ \text{PRINT: "DO YOU WISH TO GO BACK TO A LOWER RESOLUTION OR SO FRAMES?"} \]  
   \[ \text{YES \& NO} \]  
   \[ \text{INPUT THE ANSWER (C3)} \]  
   \[ C_3 = \Phi \]  
   \[ \text{RETURN} \]  

4. YES
   \[ C_3 = \Phi \]  
   \[ \text{RETURN} \]  

6

7

\[ \text{GOSUB 8400} \]  
\[ \text{PRINT: DO YOU WISH TO SCAN ALL THE DATA?} \]  
\[ \text{INPUT THE ANSWER (W1)} \]  
\[ W_1 = \Phi \]  
\[ \text{RETURN} \]  
\[ \text{GOSUB 8400} \]  
\[ W_1 \text{ VALID ?} \]  
\[ \text{YES} \]  
\[ \text{RETURN} \]  
\[ \text{GOSUB 8400} \]  
\[ W_1 \text{ VALID ?} \]  
\[ \text{NO} \]  
\[ \text{GOSUB 8400} \]  
\[ \text{INPUT THE ANSWER (W2)} \]  
\[ \text{GOSUB 8400} \]  
\[ W_2 = \Phi \]  
\[ \text{RETURN} \]  
\[ \text{W2 \text{ VALID ?}} \]  
\[ \text{NO} \]  
\[ \text{GOSUB 4400} \]  
\[ \text{MEMORY FILLED WITH DATA?} \]  
\[ \text{NO} \]  
\[ \text{GOSUB 4400} \]  
\[ \text{FIND LAST FRAME OF DATA} \]  
\[ \text{RETURN} \]  
\[ \text{COMPUTE STARTING POINT FOR PRINTOUT} \]  
\[ \text{W2 \text{ VALID ?}} \]  
\[ \text{NO} \]  
\[ \text{MEMORY FILLED WITH DATA?} \]  
\[ \text{NO} \]  
\[ \text{GOSUB 4400} \]  
\[ \text{FIND LAST FRAME OF DATA} \]  
\[ \text{RETURN} \]  
\[ \text{COMPUTE STARTING POINT FOR PRINTOUT} \]
REM... THIS PROGRAM PUTS TRAIN DATA INTO
2 REM... A MEANINGFUL FORMAT FOR ANALYSIS
3 REM... 1 JUNE 78
4 INPUT "TYPE 1 FOR HARD COPY 0 FOR CRT", D1.
5 REM... IF ANSWER ILLEGAL, SAY SO, GO BACK & GET VALID ANSWER
6 IF D1=0 OR D1=1 THEN DOSUB 8000:IF D1=0 OR D1=1 THEN 6
7 LINE #0,0:LINE #1,80:
8 REM... INITIALIZE ARRAY STARTING POINTER
9 K=0
10 DIM D(50,8):P$(20),P(50),N(50),T(50)
11 REM... D2 IS THE PROGRAMMING DECISION DISPLAY VARIABLE
12 D2=0
13 REM... ASSIGN DISPLAY VARIABLE VALUES AS FOLLOWS,
14 REM... 0-CRT, 1-SERIAL OUTPUT, 2-PARALLEL OUTPUT,
15 REM... 3-CUSTOM OUTPUT
16 REM*** INPUT EVERY 150TH FRAME OF DATA FROM DISK*****
17 OPEN #6,"TDATA"
18 X=100
19 FOR I=1 TO 56400 STEP X
20 N=I/8
21 FOR J=I/10 TO 56400 STEP X
22 N(J)=I/8
23 GOSUB #10C1
24 REM... CALCULATE TIMES
25 GOSLIB 830
26 NEXT J
27 I=56400
28 NEXT I
29 I=156
30 GOSLIB 830
31 JOB REM... CHOOSE SCAN MODE
32 GOSUB 840
33 L1=48
34 L2=0
35 REM... IF FULL SCAN IS DESIRED, SET ARRAY ACCORDINGLY
36 IF K=0 THEN L1=0
37 REM*** PRINT OUT TEST POINT DATA AT EVERY 150TH FRAME***
38 DOSUB 9000
39 REM... NOW TAKE A CLOSER LOOK AT THE DATA
40 REM... FIRST, PICK A STARTING POINT
41 U=5
42 IF U=0 THEN 160
43 REM... IF TOO LARGE, SAY SO, GO BACK & GET A NEW NUMBER
44 IF U>7050 THEN DOSUB 8000
45 IF U>7050 THEN 160
46 U=U-150
47 IF U=0 THEN 160
48 REM... IF NO MULTIPLE OF 150, SAY SO, GO BACK & GET ONE THAT IS
49 IF U=0 THEN DOSUB 8000
50 U=U-150
51 IF U=0 THEN 160
52 IF U=0 THEN 100
53 INPUT S1
54 U=51
55 REM... CHECK FRAME NUMBER FOR VALIDITY
56 IF U=0 THEN 160
57 REM... IF TOO LARGE, SAY SO, GO BACK & GET A NEW NUMBER
58 IF U>7050 THEN DOSUB 8000
59 IF U>7050 THEN 160
60 U=U-150
61 IF U=0 THEN 160
62 IF U=0 THEN 100
63 IF U=0 THEN
149 GOTO 142
150 DOSUB 8000
155 GOTO 100
160 REM**** NOW, INPUT EVERY 15TH FRAMES FROM DISK****
170 OPEN #0, "TDATA"
175 SO=51*8
177 REM...SET ABBREVIATED PRINT VARIABLE FOR MEDIUM RESOLUTION
178 S7=2
180 IF SO=56400 THEN X4=935
185 X4=120
200 FOR I1=50 TO 50+X4 STEP X2
205 J=(I1-SO)/X2
210 N(J)=I1/8
220 GOSUB 1.00(1
221 REM...CALCULATE TIMES
222 DOSUB 8000
223 NEXT I1
226 CLOSE #0
227 IF X4>1100 THEN X5=10.
230 IF X4<1100 THEN X5=7
235 L1=5
238 L2=1
240 B1=0
245 REM***PRINT OUT TEST POINT DATA FOR EVERY 15TH DATA POINT***
246 DOSUB 9000
259 ! #02,
260 ! #02, "CLOSER LOOK REQUIRED?"
261 ! #02, "O-NO,1-YES"
262 REM...IF ANSWER ILLEGAL, SAY SO,GO BACK & GET VALID ANSWER
270 IF C1=0 OR C1=1 THEN 440
275 REM...IF ANSWER ILLEGAL, SAY SO,GO BACK & GET VALID ANSWER
280 DOSUB 8000
285 GOTO 400
290 REM...IF CLOSER LOOK NOT WANTED, CONSIDER MORE PRINTOUT...
295 REM.......AT THIS RESOLUTION.
300 IF C1=0 THEN PRINT #D2, "ANOTHER FRAME?"
305 IF C1=0 THEN PRINT #D2, "O-NO,1-YES"
310 REM...IF CLOSER LOOK WANTED, CONTINUE ON......
315 IF C1=1 THEN 500
320 INPUT R1
325 IF R1=1 OR R1=0 THEN 480
330 REM...IF MORE PRINTOUT NOT WANTED AT THIS RESOLUTION...
335 REM.......CONSIDER A LOWER RESOLUTION PRINTOUT
340 IF R1=0 THEN 730
345 REM...IF MORE PRINTOUT WANTED AT THIS RESOLUTION....
350 REM...........DO BACK AND GET IT.
355 IF R1=1 THEN 100
360 IF X4>1100 THEN X8=(X4+1)/8
365 IF X4>1100 THEN X8=X4/8
370 IF K4=1
375 REM...INITIALIZE HEADING & LEGEND PRINT VARIABLE
380 IF #02, "REFERENCE A STARTING POINT FOR A CLOSER LOOK"
385 INPUT S2
390 IF S2<7153 THEN L3=15
395 IF S2>7153 THEN L3=7167-62
400 S4=S2*8
405 IF S2<0 OR S2>7167 THEN 540
410 IF K4=1 THEN 552
415 ! #02, "HOW MANY FRAMES?"
420 ! #02, "IF STARTING FRAME)=7152, NUMBER OF FRAMES"
425 ! #02, "WILL BE PREDETERMINED AND RUN THRU LAST FRAME."
533 IF S2+F2>7167 THEN #D2,"TOO MANY FRAMES FOR SELECTED"
534 INPUT F2
535 IF S2+F2>7167 THEN #D2,"STARTING POINT."
536 IF S2+F2<=7167 THEN 552
537 REM.. IF TOO MANY FRAMES OR ILLEGAL STARTING POINT SELECTED.
538 REM..... SAY SO, GO BACK AND GET VALID INFORMATION
539 GOSUB 8000
540 GOTO 500
541 REM.. SET ARRAY LENGTH AND SAMPLE RATE
542 IF S4=57224 THEN X7=1.20
543 IF S4=57224 THEN X7=57336-S4
544 IF S4=57216 THEN F2=15
545 IF S4=57224 THEN F2=(57336-S4)/8
546 K4=K4+1
547 X3=8
548 B1=0
549 REM.. IF LESS THAN 15 FRAMES REMAIN, SET ARRAY ACCORDINGLY.
550 L1=L3+IF F2<15 THEN L1=F2
551 L2=1
552 REM******INPUT EACH FRAME OF DATA FROM DISK******
553 OPEN #0,"TDATA"
554 FOR I1=S4 TO S4+X7 STEP X3
555 J=(I1-S4)/X3
556 NCJ=I1/8
557 GOSUB 1000
558 GOSUB 8300
559 NEXT I1
560 CLOSE #0
561 REM.. SET ABBREVIATED PRINT DISPLAY FOR HIGHEST RESOLUTION
562 S7=3
563 REM******PRINT OUT TEST POINT DATA FOR EACH FRAME******
564 GOSUB 9000
565 REM.. ADJUST NUMBER OF FRAMES AND STARTING POINT......
566 REM..... TO PRINT OUT MORE DATA
567 F2=F2-16
568 S2=S2+16
569 REM.. IF FRAMES ALL PRINTED, CONTINUE ON............
570 IF F2=0 THEN 620
571 REM.. IF NOT, GO BACK AND DO MORE
572 IF F2=0 THEN 514
573 ! #D2,"MORE FRAMES?"
574 ! #D2,"0-NO,1-YES"
575 INPUT R2
576 IF R2=0 OR R2=1 THEN GOTO 680
577 REM.. IF ANSWER ILLEGAL, SAY SO, GO BACK & GET VALID ANSWER
578 GOSUB 8000
579 GOTO 620
580 REM.. IF MORE FRAMES ARE TO BE PRINTED, GO BACK AND DO IT
581 IF R2=1 THEN 500
582 REM.. IF NOT, CONSIDER LOWER RESOLUTION SCAN
583 ! #D2,"DO YOU WISH TO GO BACK TO A LOWER RESOLUTION SCAN"
584 ! #D2,"IF 15 FRAMES?"
585 ! #D2,"1-YES,0-NO"
586 INPUT C3
587 REM.. IF ANSWER ILLEGAL, SAY SO, GO BACK & GET VALID ANSWER
588 IF C3=0 OR C3=1 THEN GOSUB 8000
589 IF C3=0 OR C3=1 THEN 690
590 REM.. IF LOWER RESOLUTION SCAN WANTED, GO BACK......
591 IF C3=1 THEN 100
592 REM.. IF NOT, CONSIDER LOWEST RESOLUTION SCAN
593 IF C3=0 THEN ! #D2,"DO YOU WISH TO GO BACK TO LOWEST"
594 IF C3=0 THEN ! #D2,"RESOLUTION SCAN OF 150 FRAMES."
595 IF C3=0 THEN ! #D2,"1-YES,0-NO"
750 IF U$=0 THEN INPUT W$  
752 REM . IF ANSWER ILLEGAL,SAY SO,GO BACK & GET VALID ANSWER  
755 IF W$<0 OR W$>1 THEN GOSUB 8000  
757 IF W$<0 OR W$>1 THEN 730  
760 IF W$=1 THEN GOTO 24  
995 REM . DO LINE FEEDS BEFORE EXITING PROGRAM  
999 GOTO 360000  
1000 REM . READ & DISPLAY INCOMING DATA  
1010 READ #0 X11,86(J,1),86(J,2),86(J,3),86(J,4)  
1020 READ #0 X11+4,86(J,5),86(J,6),86(J,7),86(J,8)  
1030 IF 0 THEN 1040  
1040 D(J,1),D(J,2),D(J,3),D(J,4),D(J,5),D(J,6),D(J,7),D(J,8)  
1050 RETURN  
4400 REM .**NON-WRAPAROUND ROUTINE******  
4401 REM . SET NON-WRAPAROUND VARIABLE  
4402 K6=1  
4405 FOR J=48 TO 0 STEP -1  
4410 IF D(J,1)+D(J,2)+D(J,3)+D(J,4)>0 THEN 4416  
4415 IF D(J,5)+D(J,6)+D(J,7)+D(J,8)>0 THEN 4425  
4416 IF J=0 THEN "CONTINUE PROGRAM BY TYPING "CONT" "  
4417 IF J=0 THEN "ONLY ONE FRAME OF DATA PRESENT!"  
4418 IF J=0 THEN "REFERENCE BACK IN TIME ILLEGAL!"  
4419 IF J=0 THEN K5=1  
4420 IF J=0 THEN STOP  
4421 IF J=0 THEN RETURN  
4422 IF J=0 THEN RETURN  
4423 T(48)=T(J)  
4424 RETURN  
4425 NEXT J  
4430 IF 0 THEN "ALL ZERO DATA ON DISK"  
4435 RETURN  
8000 REM .**INVALID ENTRY ROUTINE******  
8005 IF 0 THEN 8010  
8010 IF 0 THEN "ILLEGAL CHOICE,TRY AGAIN"  
8015 IF 0 THEN 8020  
8020 RETURN  
8300 REM .**TIME CALCULATION******  
8310 T1=256*+D(J,1)  
8320 T(J)=(T1+D(J,2))/10  
8330 RETURN  
8400 REM .**SCAN MODE DECISION ROUTINE******  
8410 IF 0 THEN "DO YOU WISH TO SCAN ALL THE DATA?"  
8420 IF 0 THEN "YES,NO"  
8430 INPUT W1  
8435 REM . IF ANSWER ILLEGAL,SAY SO,GO BACK & GET VALID ANSWER  
8440 IF W1<0 OR W1>1 THEN GOSUB 8000  
8445 IF W1<0 OR W1>1 THEN 8460  
8450 IF W1=0 THEN 8465  
8455 REM . IF TOTAL SCAN REQUESTED,GO NO FURTHER HERE....  
8460 RETURN  
8462 REM . OTHERWISE,GET TIME REFERENCE.  
8465 IF 0 THEN "HOW MANY SECONDS BACK FROM THE TIME"  
8466 IF 0 THEN "THE 'STOP MONITOR' BUTTON WAS DEPRESSED"  
8467 IF 0 THEN "DO YOU WISH TO LOOK?"  
8468 IF 0 THEN 8469  
8469 K1=1 REM . SET PARTIAL SCAN VARIABLE  
8470 INPUT W2  
8471 IF W2<0 AND S2=2 THEN LET B1=15  
8472 IF W2<0 AND S7=1 THEN LET B1=48  
8473 IF W2<0 THEN 8525  
8474 REM . IF NO TIME ENTRY FOUND,ENTER NON-WRAPAROUND ROUTINE  
8475 IF T(48)+T(47)=0 THEN GOSUB 4400  
8477 REM . IF ONLY ONE FRAME OF DATA FOUND,GO NO FURTHER....  
8478 IF K5=1 THEN RETURN  
8479 REM . IF OPERATOR ATTEMPTS TO GO BACK TOO FAR(IN TIME)....
8480 REM...SAY SO. GO BACK & GET VALID TIME
8483 IF T(48)-W2(T)<0 THEN GOSUB 9000
8485 IF T(48)-W2(T)<0 THEN 8465
8487 REM...COMPUTE TIME ADJUSTMENT
8490 T3=T(48)-82
8492 REM.......TO FIND STARTING POINT FOR PRINTOUT.
8495 FOR J=0 TO 48
8500 IF T3(T)=0 THEN 8510
8505 13010 8520
8508 REM...SET STARTING POINT
8511 B1=J-1
8512 REM..RESET LAST TIME ENTRY FOR NON-WRAPAROUND PRINTOUT
8515 RETURN
8520 NEXT J
8525 RETURN
8530 REM..PRINT LEGEND
8540 ! #D1,TAB(10),"LEGEND"
8550 ! #D1,TAB(10),"------"
8560 ! #D1,TAB(10),"1-TRUE "
8570 ! #D1,TAB(10),"0-FALSE"
8580 ! #D1,TAB(10),"B-BRAKE"
8590 ! #D1,TAB(10),"P-PROPULSION"
8600 ! #D1,TAB(10),"U(OR X)-UNDEFINED"
8610 1 ! #D1,TAB(10),"TEST POINT 20 GIVEN IN MPH"
8620 RETURN
8630 REM**PRINT HEADING(HARDCOPY AVAILABLE BY HITTING 'H' KEY)**
8640 IF INP(252)=72 THEN D1=1
8650 IF S7=3 AND K4>2 THEN 9020
8660 ! #D1,TAB(25),"TEST DATA"
8670 ! #D1,TAB(33),"(TP 1 THRU 34)"
8680 ! #D1,
8690 ! #D1,"DATA TIME",TAB(31),1,1,1,TAB(38),1,TAB(41),1,TAB(44),1,
8700 ! #D1,TAB(48),2,TAB(52),2,2,2,2,2,2,2,2,2,
8710 ! #D1,TAB(70),3,3,3,3,3
8720 ! #D1,"PNT (SEC)",TAB(12),1,2,3,4,5,6,7,
8730 ! #D1,"TEST DATA"
8740 ! #D1,TAB(27),8,9,
8750 ! #D1,TAB(31),0,1,6,TAB(38),7,TAB(41),8,TAB(44),9,
8760 ! #D1,TAB(48),0,TAB(52),1,2,3,4,5,6,7,8,9,
8770 ! #D1,TAB(70),0,1,2,3,4
8780 ! #D1,"----"-----"
8790 9020 FOR J=B1 TO L1 STEP L2
8800 REM..IF ABBREVIATED PRINTOUT WANTED,HIT 'C' KEY
8810 IF INP(252)=67 THEN GOSUB 20000
8820 FOR K1 TO 8
8830 REM...FIRST,MAKE NECESSARY DECIMAL-TO-BINARY CONVERSIONS
8840 IF K2 AND K8 THEN GOSUB 9500
8850 REM...THEN CONVERT SELECTED BINARY DATA TO LETTERS
8860 IF K3 THEN GOSUB 9600
8870 IF K4 THEN GOSUB 9650
8880 IF K5 THEN GOSUB 9700
8890 IF K6 THEN GOSUB 9750
8900 IF K7 THEN GOSUB 9800
8910 IF K8 THEN GOSUB 9850
8920 REM.GO TO NEXT WORD
8930 9105 NEXT K
8940 REM*****PRINT DATA & HEADINGS******
8950 GOSUB 9900
8960 REM...IF PAUSE IS WANTED DURING PRINTOUT,HIT 'P' KEY
8970 IF INP(252)=80 THEN 9126
8980 REM...IF SWITCH FROM PRINTER TO CRT IS NEEDED,HIT 'T' KEY
8990 IF INP(252)=84 THEN D1=0
9000 REM*****THEN,GO TO NEXT FRAME ******
9010 NEXT J
9020 REM..SKIP PROGRAMMED PAUSE IF HIGHEST RESOLUTION PRINTOUT
9132 IF S7=0 THEN 9138
9133 INPUT"HIT RETURN TO CONTINUE......",O$
9134 IF S7>0 THEN 9140
9135 RESTORE
9136 REM...IF PRINTOUT NOT COMPLETE, SKIP PRINTING LEGEND
9137 IF F2>15 THEN RETURN
9140 REM***** PRINT LEGEND******
9150 GOSUB 8000
9160 RETURN
9500 REM******DECIMAL TO BINARY CONVERSION******
9503 D8=128
9508 D9=0(K,J)
9515 FOR B=1 TO 8
9516 Q=(K-3)*8+(B+2)
9524 IF D9(Q) THEN 9536
9527 Q(Q)=1
9530 D9=DS4-D8
9533 GOTO 9539
9536 Q(Q)=0
9539 D8=D8/2
9545 NEXT B
9551 RETURN
9600 REM*****CREATE PRINT ARRAY FOR DATA WORD#0******
9608 REM...CONVERT SELECTED BINARY DATA TO LETTERS
9609 IF 0(4)+0(3)=1 THEN LET P$(1,2)="11"
9611 IF 0(4)+0(3)=0 THEN LET P$(1,2)="00"
9614 IF 0(4)=0 AND 0(3)=1 THEN LET P$(1,2)="01"
9617 IF 0(4)=1 AND 0(3)=0 THEN LET P$(1,2)="10"
9620 REM...PUT REMAINING OUTPUT ARRAY VALUES IN PRINT ARRAY
9623 P(6)=0(5)
9626 P(5)=0(6)
9629 P(4)=0(7)
9632 P(3)=0(8)
9635 P(2)=0(9)
9638 P(1)=0(10)
9640 RETURN
9650 REM*****CREATE PRINT ARRAY FOR DATA WORD#01******
9653 REM...FIRST CONVERT SELECTED BINARY DATA TO LETTERS
9655 IF 0(12)+0(11)=1 THEN LET P$(3,4)="11"
9656 IF 0(12)+0(11)=0 THEN LET P$(3,4)="00"
9661 IF 0(12)>0 AND 0(11)=1 THEN LET P$(3,4)="01"
9664 IF 0(12)=1 AND 0(11)=0 THEN LET P$(3,4)="10"
9667 IF 0(14)+0(13)=1 THEN LET P$(5,6)="11"
9670 IF 0(14)+0(13)=0 THEN LET P$(5,6)="00"
9673 IF 0(14)=0 AND 0(13)=1 THEN LET P$(5,6)="01"
9676 IF 0(14)=1 AND 0(13)=0 THEN LET P$(5,6)="10"
9678 REM...THEN PUT REMAINING OUTPUT IN PRINT ARRAY
9680 P(11)=0(15)
9682 P(10)=0(16)
9684 P(9)=0(17)
9686 P(8)=0(18)
9688 RETURN
9700 REM*****CREATE PRINT ARRAY FOR DATA WORD #2******
9703 REM...FIRST CONVERT SELECTED BINARY DATA TO LETTERS
9705 IF 0(23)+0(24)=1 THEN LET P$(7,8)="11"
9706 IF 0(23)+0(24)=0 THEN LET P$(7,8)="00"
9711 IF 0(23)=1 AND 0(24)=0 THEN LET P$(7,8)="01"
9714 IF 0(25)+0(24)=1 THEN LET P$(9,10)="11"
9717 IF 0(25)+0(24)=0 THEN LET P$(9,10)="00"
9720 IF 0(25)+0(26)=0 THEN LET P$(9,10)="01"
9726 IF 0(25)=0 AND 0(26)=1 THEN LET P$(9,10)="10"
9729 REM...THEN PUT REMAINING OUTPUT IN PRINT ARRAY
9730 P(24)=0(19)
9733 P(23)=0(20)
```
P(22) = 0(21)
P(21) = 0(22)
RETURN

RE***CREATE PRINT ARRAY FOR DATA WORD#03*****
P(32) = 0(27)
P(31) = 0(26)
P(30) = 0(29)
P(29) = 0(30)
P(28) = 0(31)
P(27) = 0(32)
P(26) = 0(33)
P(25) = 0(34)
RETURN

RE***CREATE PRINT ARRAY FOR DATA WORD#04*****
P(34) = 0(41)
P(33) = 0(42)
RETURN

RE***CREATE PRINT ARRAY FOR DATA WORD#05*****
P(28) = 0(J, 8)/5
RETURN

RE***PRINT TEST POINT DATA*****
#D1, N(J), TAB(4), T(J),
#D1, TAB(12), P(1), P(2), P(3), P(4), P(5), P(6), TAB(25), P(1, 2),
#D1, TAB(27), P(8), P(9),
#D1, TAB(32), P(10), P(11), TAB(36), P(5, 6), TAB(39), P(3, 4),
#D1, TAB(42), P(9, 10), TAB(45), P(7, 8),
#D1, P(20), TAB(52), P(21), P(22), P(23), P(24),
#D1, P(25), P(26), P(27), P(28), P(29),
#D1, TAB(70), P(30), P(31), P(32), P(33), P(34)
RETURN

RE***ABBREVIATED PRINTOUT ROUTINE******

IF S7=1 THEN 20000
IF S7=2 THEN 20000
IF S7=3 THEN 20000
REM... FOR LOW RESOLUTION SCAN
20005 #D2.
20010 FOR J=0 TO 10
20015 #D2, N(J),
20020 NEXT J
20025 #D2.
20030 FOR J=11 TO 20
20035 #D2, N(J),
20040 NEXT J
20045 #D2.
20050 FOR J=21 TO 30
20055 #D2, N(J),
20060 NEXT J
20065 #D2.
20070 FOR J=31 TO 40
20075 #D2, N(J),
20080 NEXT J
20085 #D2.
20090 FOR J=41 TO 47
20095 #D2, N(J),
20100 NEXT J
20105 #D2.
20110 INPUT "NEW FRAME NUMBER? ", X9
20115 J=X9/150
20120 IF X9>7050 THEN #D2, "ILLEGAL CHOICE, TRY AGAIN!"
20125 IF X9>7050 THEN 20000
20130 GOTO 20000
```
20200 REM...FOR MEDIUM RESOLUTION SCAN
20210 FOR J=0 TO 7
20215 ! #D2,N(J),
20220 NEXT J
20225 ! #D2,
20230 IF X5=7 THEN 20280
20235 FOR J=8 TO 10
20240 ! #D2,N(J),
20245 NEXT J
20250 ! #D2,
20260 INPUT "NEW FRAME NUMBER? ",X9
20265 IF X9<(S2+X4)/8 THEN ! #D2,"ILLEGAL CHOICE,TRY AGAIN!"
20270 IF X9<(S2+X4)/8 THEN 20000
20275 J=(X9-S2)/15
20280 GOTO 20500
20285 REM...FOR HIGH RESOLUTION SCAN
20290 IF S2>=7.15 THEN 20350
20295 FOR J=0 TO 10
20300 ! #D2,N(J),
20305 NEXT J
20310 ! #D2,
20315 FOR J=11 TO X7/8
20320 ! #D2,N(J),
20325 NEXT J
20330 ! #D2,
20335 GOTO 20375
20340 FOR J=0 TO X7/8
20345 ! #D2,N(J),
20350 NEXT J
20355 ! #D2,N(J),
20360 NEXT J
20365 ! #D2,
20370 INPUT "NEW FRAME NUMBER? ",X9
20375 IF X9<(S2+X7)/8 THEN ! #D2,"ILLEGAL CHOICE,TRY AGAIN!"
20380 IF X9<(S2+X7)/8 THEN 20000
20385 J=X9-S2
20390 RETURN
30000 REM*****PAPER ADVANCE*****
30010 FOR I=1 TO 19
30020 ! #1,
30030 NEXT
30040 STOP
OUTPUT ROUTINE

START

LOAD MEMORY POINTER (LDPD)

DUMP FLAG?

YES

RDY TO DUMP?

NO

SET "ERROR" INDICATOR

IDLE

SET "DUMPING" FLAG (WITH DISABLE)

ILLUMINATE DUMPING LAMP

TEST FOR OVERFLOW (LIMIT)

MEMORY POINTER RE-INITIALIZED?

NO

OVERFLOW INDICATOR SET?

NO

RE-INITIALIZE MEMORY POINTER TO 2000 (WRITE)

NO

OUTPUT DATA FROM MEMORY TO DISK (WRITE)

YES

SET "DUMP COMPL." FLAG (WITH DISABLE)

SET "RDY TO DUMP" BIT ALLOWING RE-ENTRY

ILLUMINATE "READY TO DUMP" & "DUMP COMPL." LAMPS

IDLE
INPUT ROUTINE

START

CHECK FLAG
   a. reset
   b. monitoring

RESET?
   YES
   SET MONITORING FLAG AND TURN ON INDICATOR
   YES
   INCREMENT LSB OF INCREMENT COUNTER
   NO
   LSB FULL?
      NO
      SET ERROR INDICATOR
      YES
      INCREMENT MSB OF INTERRUPT COUNTER

MONITORING?
   NO
   IDLE

IDLE

LAST WORD?
   YES
   LOAD FINAL MEMORY POINTER
   NO
   STORE MEMORY POINTER (STWDL)

TEST FOR MEMORY OVERFLOW (LIMIT)

STORE LSB OF TIME

STORE MSB OF TIME

TEST FOR MEMORY OVERFLOW (LIMIT)

STORE LSB OF TIME

TEST FOR MEMORY OVERFLOW (LIMIT)

STORE TRAIN DATA

DATA CHANGE?
   YES
   LAST WORD?
      NO
      NO
      NO
      IDLE
      YES
      SHIFT DATA (NEW LOC. TO OLD)
   NO
   YES
   LAST WORD?
      NO
      NO
      NO
      IDLE
      YES
      SHIFT DATA (NEW LOC. TO OLD)
   NO
   YES
   LAST WORD?
      NO
      NO
      NO
      IDLE
      YES
      SHIFT DATA (NEW LOC. TO OLD)
   NO
   YES
   LAST WORD?
Power On Routine:

START

INITIALIZE STACK POINTER

INITIALIZE INTERRUPT COUNTER

INITIALIZE OVERFLOW INDICATOR

INITIALIZE FINAL MEMORY PTR. AND STORE

CLEAR DATA STORAGE MEMORY (2000-FFFF)

ZERO FLAG WORD

TURN OFF ALL INDICATORS

INPUT BASELINE DATA (TRIJIN)

IDLE
READY
ASSMX 0 8000
0000 00 0020 START NOP
0001 C3 40 00 0030 JMP FINIT
0004 DS START+18H-6 0040 DS START+18H-6
0018 C3 80 01 0050 INT4 JMP VDASI ;SERVICE INPUT ROUTINE
0020 C3 80 02 0060 DS START+20H-6 ;SERVICE REINITIALIZE ROUTINE
0023 DS START+28H-6 0070 INT3 JMP STKTO ;SERVICE OUTPUT ROUTINE
0028 C3 80 03 0080 INT2 JMP DUMP ;SERVICE INITIALIZE STACK POINTER
0033 DS START+40H-6 0090 INT1 JMP DSABL ;SERVICE DISABLE ROUTINE
0040 31 FF 0100 DB 13H
0042 13 0110 INTO JMP PWKOH ;SERVICE POWER ON ROUTINE
0043 C3 00 01 0120 DS START+100H-6
0046 FB 0130 IDLE EI ;ENABLE INTERRUPTS
0047 76 0140 HLT ;WAIT
0048 C3 46 00 0150 JMP IDLE ;SPIN HERE UNTIL ANOTHER INTERRUPT OCCURS
0100 3E 00 0190 DS START+100H-6
0102 32 00 10 0200 PWON MVI A,00H ;INITIALIZE INTERRUPT COUNTER
0105 32 01 10 0210 STA 1000H
0108 32 03 10 0220 STA 1001H
010B 32 05 10 0230 STA 1002H ;INITIALIZE OVERFLOW INDICATOR
010E 21 FF 1F 0240 MWRT LXX H,1FFH ;INITIALIZE FINAL MEMORY POINTER
0110 7C 0250 MOV A,L ;AND STORE AWAY
011F 32 03 10 0260 MOV A,L ;AND STORE AWAY
0112 7D 0270 STA 1003H
0113 32 04 10 0280 ZMEM MOV M,A ;CLR FINAL MEMORY LOCATION
0116 3E 00 0290 INX H
0118 77 0300 MOV A,H
011A 7C 0310 XRI 00H ;TEST FOR LIMIT ADDRESS
011B EE 00 0320 JZ FLGW
011C 2A 01 0330 MVI A,00H ;IF NOT LIMITING, CONTINUE CLEARING
011D 3E 00 0340 JMP ZMEM
011F 32 18 01 0350 FLGW LXX D,1002H ;SET FLAG POINTER
0125 11 02 10 0360 FLO1 MVI A,00H ;ZERO ACCUMULATOR
0128 3E 00 0370 STAX D
012A 12 0380 OUT 06H ;AND TURN OFF ALL DISPLAYS
012D 11 10 10 0390 BSLIN LXX D,1010H ;SET UP BASELINE STORAGE POINTER
0130 CD 50 01 0395 CALL TRNIN ;BRING DATA FROM TRAIN
0133 C3 46 00 0396 JMP IDLE ;RETURN TO EXEC
0136 8F 00 0398 DS START+150H-6
013D 0B 00 0400 TRNIN IN 00H ;INPUT TRAIN DATA
0140 DS 12 0410 STAX D ;AND STORE AWAY
0145 12 0420 INX D ;AND MOVE TO NEXT BASELINE DATA LOCATION
014A 13 0430 INX D
014E 13 0440 IN OIH ;ETCETERA
0152 12 0450 STA D
0155 DB 01 0460 INX D
0158 12 0470 STA D
015B 13 0480 IN O2H
015E 02 0490 STA D
0300 CD 70 02 2070 RSTR2 CALL STUDY ;STORE FINAL MEMORY POINTER
0303 C9 2080 RET ;GO BACK TO IDLE
0304 02 08 2081 DS START=3EH-$
0306 3A 02 10 2082 ERREN LDA 1002H ;LOAD FLAG WORD
0308 3A 10 05 2084 ORI 181H ;JOINING WITH "ERROR" BIT
030E D3 06 2086 OUT 06H ;TURN ON ERROR INDICATOR
0317 C9 2088 RET ;AND GO BACK TO IDLE
031E 20 2105 DS START=0EOOH-$
0320 1B 2110 BASE EQU 1800H
0323 9E 2115 *******THERE IS FOLLOWING ROUTINE PLACES 56K OF TEST DATA
0327 00 2116 **********ON DISK FOR LATTER ANALYSIS**********
032B 3A 90 1B 2120 WRITE LDA BASE+300H ;START DRIVE MOTOR
032F 1E 32 2130 MOV D,30 ;WAIT 30 SECTOR TIMES
0333 CD 00 19 2140 CALL BASE+1DH
0337 3A 01 1B 2150 LDA BASE+301H ;LOAD HEAD
033B 1B 00 2160 MOV D,13 ;WAIT 13 SECTOR TIMES
033F CD 00 19 2170 CALL BASE+1DH
0343 3A 1C 1B 2180 LDA BASE+31CH ;SET UP HEAD TO MOVE OUT
0347 13 10 1B 2190 TRACK 0 LDA BASE+310H ;GET STATUS
034B 80 01 2200 ANI 1 ;MASK FOR TRACK ZERO FLAG
034F C2 28 0E 2210 JNZ FIRST ;EXIT IF TRACK ZERO FOUND
0353 3A 09 1B 2220 LDA BASE+309H ;SET STEP FLIP FLOP
0357 E3 2230 XTHL ;PAUSE
035B E3 2240 XTHL
035F C2 08 1B 2250 LDA BASE+308H ;RESET STEP FLIP FLOP
0363 1E 02 2260 MOV D,2 ;WAIT 2 SECTOR TIMES
0367 CD 00 19 2270 CALL BASE+1DH
036B C2 13 0E 2280 JMP TRACK 0 ;LOOP UNTIL TRACK ZERO FOUND
036F 1A 1D 1B 2290 FIRST LDA BASE+310H ;SET UP HEAD TO MOVE IN
0373 E0 17 04 2300 LXI B,400H+23 ;8=SECTORS,C=23-TRACKS
0377 CD CX 19 2310 FIND CALL BASE+1CH ;WAIT FOR NEXT SECTOR
037F 3A 30 1B 2320 LDA BASE+330H ;GET "B" STATUS
0383 E6 07 2330 ANI OFH ;MASK FOR SECTOR COUNT
0387 FE 06 2340 CPI 6 ;LOOK FOR SECTOR 6
038B 3A 01 0E 2350 JNZ FIND
038F 3A 04 1B 2360 WRITE LDA BASE+304H ;WRITE COMMAND
0393 1E 02 2370 WRSTAT LDA BASE+310H ;GET STATUS
0397 E6 08 2380 ANI 8 ;MASK FOR "WRITE RDY" FLAG
039B CA 41 0E 2390 JZ WRSTAT ;WAIT FOR WRITE READY
039F E5 05 23A0 MOV D,5 ;SAVE COUNTERS
03A3 11 00 1A 23B0 LXI D,BASE+200H ;SET UP TO WRITE ZEROS
03A7 01 0F 00 23C0 LET B,15 ;ZEROES=15
03AB 50 1D 2430 ZERLP LDA D ;WRITE A ZERO
03AD 1D 0D 2440 DCR C ;COUNT
03AF 52 C2 50 0E 2450 JNZ ZERLP ;IF NOT DONE, WRITE MORE ZEROS
03B3 4E FB 2460 MOV E,OFH ;IF DONE, SET UP "SYNC"
03B7 1A 2470 . LDA D ;WRITE IT
03B9 7E 2480 WRAP MOV A,H ;GET DATA FROM VRAM MEMORY
03BD 3F 2490 MOV B,A ;SET UP DATA FOR WRITING
03C1 4B 2500 TAA B ;COUNT CRC
03C5 08 2510 MLT
03C9 47 2520 MOV B,A
03CD 1A 2530 LDA D ;WRITE DATA
03D5 23 2540 INX H ;BUFFER MEMORY POINTER
03D9 3C 2550 MOV A,H ;CHECK FOR MEMORY OVERFLOW
03DE 85 2560 ORA L
03E1 C2 66 0E 2570 JNZ CONT ;IF NOT REACHED, DON'T WRAPAROUND
03E4 26 20 2580 MOV H,20H ;IF SO, REINITIALIZE TO 2000H
03E8 0D 2590 CONT DCR C ;COUNT BYTES
03EB 22 58 0E 2600 JNZ WRAP ;WRITE 256 TIMES
03EE 5B 2610 MOV E,B ;WRITE CRC
03F0 1A 2620 LDA D
03F4 C1 2630 POP B ;GET TRACK & SECTOR COUNTERS
03F8 CD CE 19 2640 CALL BASE+1CH ;WAIT FOR NEXT SECTOR
03FE 05 2650 DCR B ;COUNT SECTORS
O7E1 C2 3E UE 2650 JNZ WRIT LOOP IU SECTORS PER TRACK
O7E4 3A 09 18 2670 LDA BASE+309H ;SET HEAD FLIP FLOP
O7E7 E3 2680 XTHL ;PAUSE
O7E8 E3 2690 XTHL
O7E9 3A 08 18 2700 LDA BASE+308H ;RESET HEAD FLIP FLOP
O7EC 16 0A 2710 HVI D,10 ;WAIT IU SECTORS
O7ED CD 00 19 2720 CALL BASE+1D0H
O7EE 06 0A 2730 HVI B,10 ;RESET SECTOR COUNT TO 10
O7EF 00 0A 2740 DCR C ;COURT TRACK
O7E4 C2 3E 0E 2750 JNZ WRIT LOOP 23 TRACKS
O7E7 C9 2760 RET ;GO BACK TO "DUMP" ROUTINE

BASE 1800 2120 2140 2170 2180 2190 2220 2250 2270 2290 2310
2320 2360 2370 2410 2640 2670 2700 2720

CLRN 0290 1476 1484
BSLN 012D
BUMP 0246 1324
CM3 0299
CMT 0666 2570
DATN1 01A6 0780
DATN2 039E 2020
DBIT 0318 1540
DLED 031D 1600
DSPA 0380 0110
DSK 0341 1675 1705
DUMP 0300 0090
ENTR 030B
ERR 03EO 0690 1430 1570 1840
EXIT 0259
EXIT2 01F5 0920
EXIT3 0289
EXIT4 0346
FIND 0E31 2350
FIRST 02E6 2210
FLAG 03A9
FLG 0128
FLG2 0190 0670
FLGWD 0125 0320
INLE 0046 0180 0396
INCFP 01AC
INCME 0197 0674
INT 004D
INT1 0030
INT2 0028
INT3 0020
INT4 0018
LDV01 0260 0950 1510 1810
LIN 01D7 1100
LIMIT 0160 0960 0990 1040 1060 1850 1880 1890 1930
LMTST 0244 1472
MNP 018B
NEWR 03A3
ODTH 01E2
OCR 0331 1655 1670
PIC 0040 0030
PORT 1080 1233
PROM 0100 0130
QTCLR 0345
REINT 0251 1321
RESET 033C 1690
RSTR 01F2 1230
RSTK 0380
STK 0254
STUP 0295
START 0000 0040 0060 0080 0100 0120 0190 0398 0630 1299 1339 1357
1400 1500 1800 2091 2105
THIS PROGRAM IS USED FOR REPROGRAMMING

* IX FROMS AFTER REASSEMBLY
* ALL UNCHANGED LOCATIONS ARE SET TO DDR
* ALL CHANGED LOCATIONS ARE SET TO NEW VALUE
* LOAD OLD OBJECT CODE AT 8000H
* LOAD NEW OBJECT CODE AT A000H
* EDITED CODE IS LOADED AT B000H
* EXEC 2028
APPENDIX D

Data Analysis
<table>
<thead>
<tr>
<th>Step</th>
<th>Time (sec)</th>
<th>Data 1</th>
<th>Data 2</th>
<th>Data 3</th>
<th>Data 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.245</td>
<td>5.67</td>
<td>8.901</td>
<td>1.67</td>
<td>8.901</td>
</tr>
<tr>
<td>2</td>
<td>1.245</td>
<td>5.67</td>
<td>8.901</td>
<td>1.67</td>
<td>8.901</td>
</tr>
<tr>
<td>3</td>
<td>1.245</td>
<td>5.67</td>
<td>8.901</td>
<td>1.67</td>
<td>8.901</td>
</tr>
<tr>
<td>4</td>
<td>1.245</td>
<td>5.67</td>
<td>8.901</td>
<td>1.67</td>
<td>8.901</td>
</tr>
</tbody>
</table>
Data Analysis: Vehicle failure #3
(based on data outputed 10/30/18)