SINGLE POINT USER-INTERFACE SOFTWARE DESIGN for TELEMETRY STATIONS

Brian A. Corbin
Computer Engineer
NASA

KEY WORDS

User interface, Shell

ABSTRACT

As the repetitive processes in the manufacturing community are being automated to reduce operating cost, the satellite tracking station (being for the most part a repetitive operation) can also reduce operating cost by automation. A conventional satellite tracking station requires personnel to setup, monitor, and adjust a variety of equipment, coordinate data collection, and archive collected data. By automating, the above duties in addition to managing the station can be done through a single-point user interface. This paper presents the methods used to design the user-interface software including graphical user interface (GUI) types of hardware I/O for data acquisition and control, and remote control.

INTRODUCTION

Although there are no shortages of data-acquisition hardware and software for both multitasking and sequential (Von-Neumann architecture) computers, this paper discuses one cost effective approach that has been implemented on NASA’s Transportable Orbital Tracking Station (TOTS).[1]

Early design decisions based on programming cost, software maintenance cost, and operational simplicity, led to the following system design.(figure #1)

A 80486 PC running at 33mhz with multi serial ports (RS-232) and multi parallel ports was selected to be the platform from which the user-interface software would run. This master or control computer would communicate through the

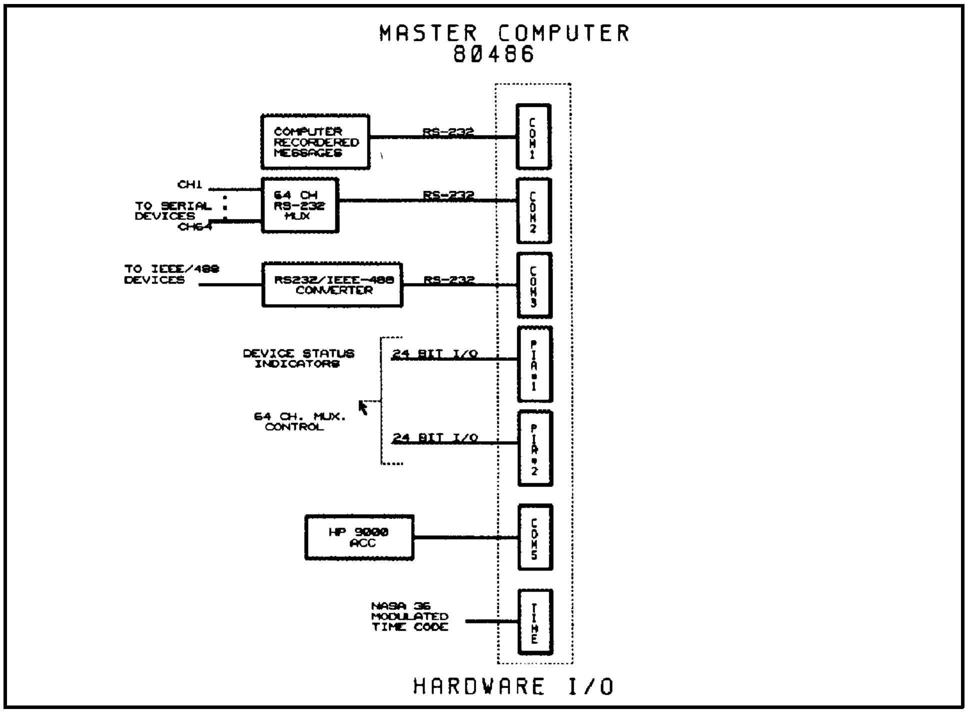

figure #1

serial and parallel ports with the test and operational equipment and other computers in the tracking system. The software would be written in **"C"** using windows and menus to exercise various station systems and equipment. In addition a graphics display or GUI for system operational status would be used to centralize information on station performance.

Although there are many components that contribute to a user friendly interface, this paper will concentrate on the design of the application SHELL program used in the above configuration.

DESIGN GOALS

The design goals of the user-interface software were to provide a single point in the entire tracking station where a user could interface with all equipments and sub-systems. This point would allow the operator to :

1. Exercise all equipment from a set-up or operational mode.
2. Monitor all equipment and systems through a GUI during the operational (tracking) mode.
3. Provide a communications port for other computers in the system
4. Provide station status to a remote user.

DESIGN

The user-interface software would consist of a user friendly menu driven main program (SHELL) that would orchestrate both the RF and data system. In particular it would provide:

1. A mechanism for running all device or equipment drivers. (figure #2)
2. A method to automatically control prepass, pass, and postpass operations.
3. A method for viewing system status from a remote site and remote control.

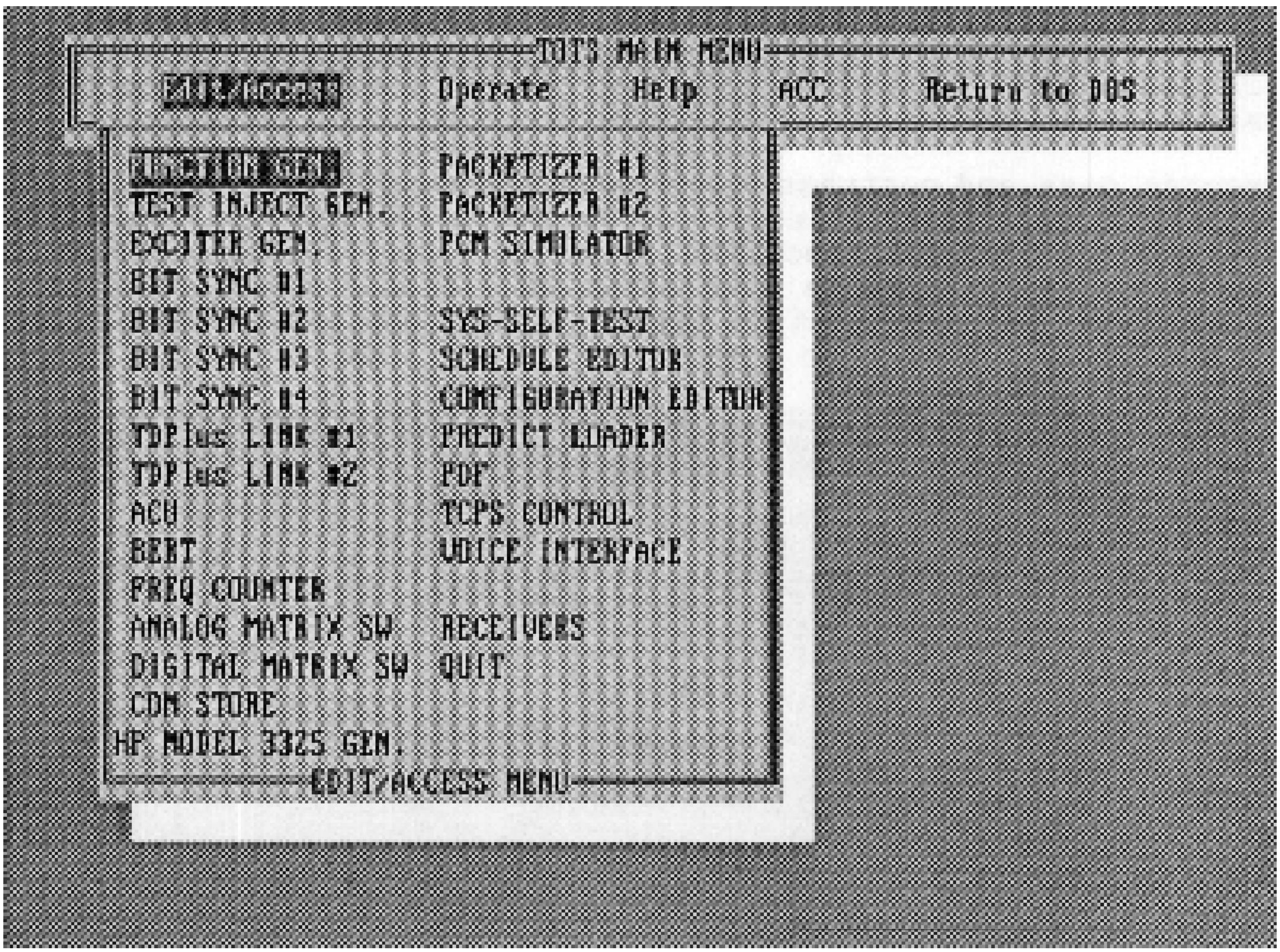

figure #2

Device drivers are functions or complete programs that interact with each instrument or piece of equipment (device). Device drivers can be purchased from the device manufacture or custom made for each application or device. These device drivers accomplish too purposes, one is to control the interface weather it be IEEE-488, RS232, or some other type. The other function is to provide a method to setup and receive status from the device.[2] Even thought device drivers are an important part of the user-interface the emphasis here is how these drivers are integrated into the shell.

The approach taken was to make all device drivers executable programs. This would make the driver maintenance less costly by not being apart of the main program (SHELL) and make the SHELL more independent by not having to be recompiled it each time a driver was changed or updated. By using process control functions such as “spawn” and “system” a child program (in this case a device driver) can be executed without terminating the parent (the SHELL). The parent then resumes execution where it left off before the spawn or system call was made. The disadvantage to this approach is memory limitations. But in most cases device drivers are relativity small (30 to 60K bytes) and if the parent program is less than 640K bytes minus driver size and DOS overhead, no extended memory managing programs will be needed. By using this technique (with arguments) and running all programs (SHELL and drivers) from virtual disk, multiple device drivers are accessible by parent/child switching.

Prepass, pass, and postpass operations are handled by both time based sequences and physical events. By using the station clock (NASA 36) and time events (received from the Antenna Control Computer) the time of occurrence and duration of all operations can be controlled. In addition to the automatic operation, events can be initiated by the station operator with all results logged to disk.

During the operational mode the SHELL would provide a graphics display (GUI) that would be updated by polling all pertinent parallel and serial lines. (figure #3) The GUI is displayed in two phases. All background or static information are displayed first. Then the indicators are updated as their respective I/O device are polled. To keep the update or polling sequence efficient (have sufficient samples) the two slowest operations screen updates and reading the serial interface are designed such that the screen indicators are serviced only when a change in status occurred (i.e. lock to loss or loss to lock) and the 19.2 K BAUD serial link serviced only when a command is being received. By using these techniques and filtering the amount of data that is logged to disk satisfactory performance can be achieved.

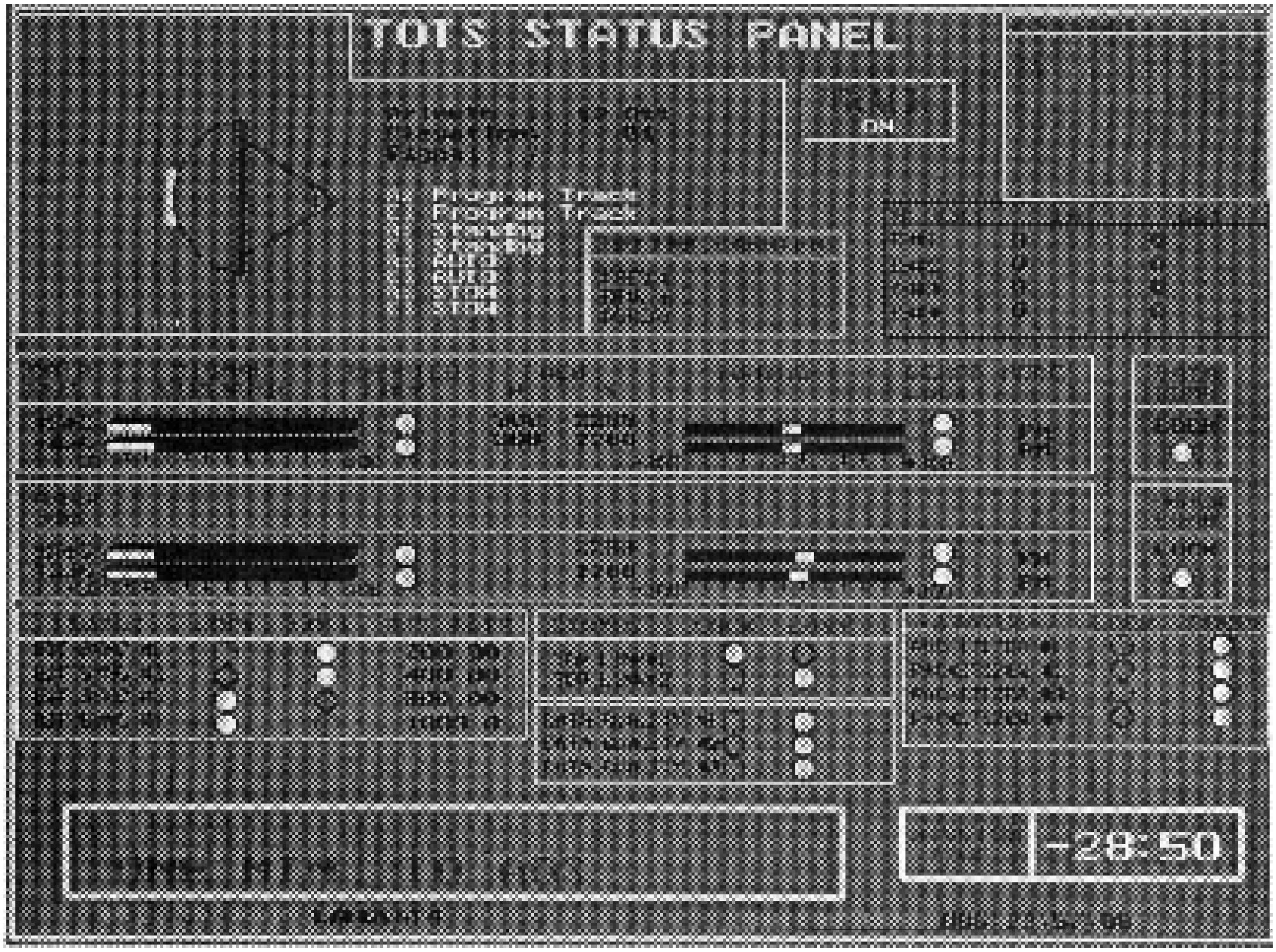

figure #3

Many GUI’s are available but only a few are oriented toward instrumentation. The GUI’s come with packaged indicators, such as gages, slide bars annunciators and seven segment displays. With these functions there is still a fair amount of special design to create a custom display. All graphics in figure #3 was done by a fairly low level GUI, meaning all figures were made from basic geometric figures.

The third major function was to provide a method for viewing system status off site (a remote location) with the same detail and control as when operating the station locally. Several commercial packages are available that provide remote control and file transfer functions. The one chosen advertised graphics capability but early testing revealed that trying to remote the graphics screen (figure #3) at 2400 BAUD (about the best you can expect from commercial lines) was unacceptable. A second status screen (figure#4) using character based graphics was designed to do all remote

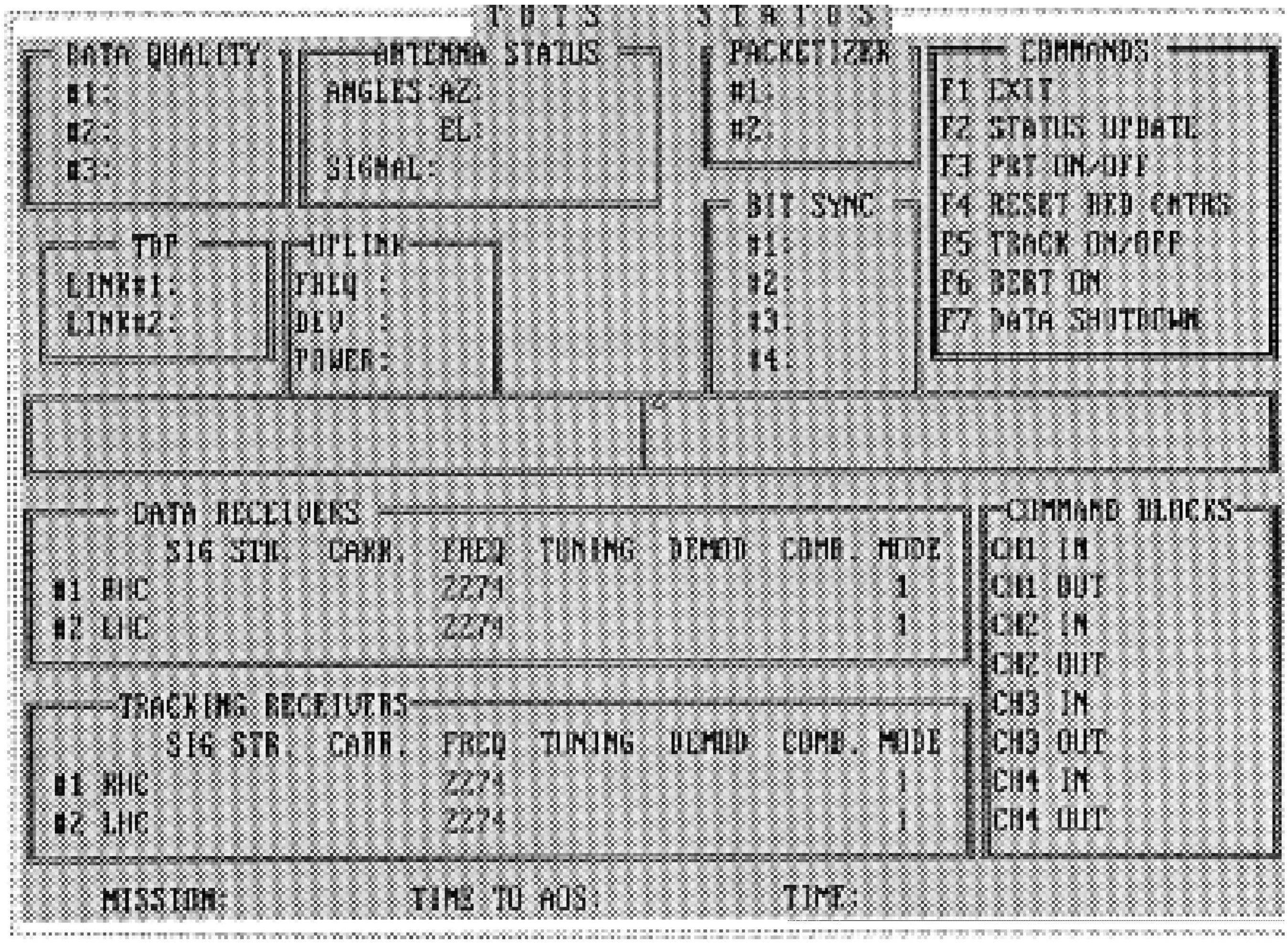

figure #4

operations. This method although restrictive to the type of information that can be put on a screen proved to be satisfactory allowing a remote operator complete control without disrupting or interfering with any functions or operations on site.

CONCLUSION

Although this paper has touched on only a few of the functions of the user-interface and being this is version 1.0 other functions will be added and or deleted when the software is beta tested. This being the case due to the modular construction of the software, upgrades will be a minor task. On line help menus will be changed to reflect these updates thus eliminating the need to print new manuals or errata sheets. The user-interface as it implies is where users will have the most interaction with the tracking system. This one program (the SHELL) will determine, friendliness, performance and acceptability of the system.

ACKNOWLEDGEMENT

I would like to thank Dave Massey of NASA/Goddard Space Flight Center for his helpful suggestions on this project.

REFERENCES

1. Morris, R.A., Powell, W.R, and Bundick, S.N.,”A Precision Transportable Tracking Telemetry System”, International Telemetering Conference, Vol. XXVII, pp ll9-127, 1991.

2. Romanchik D.,”Rolling Your Own Instrument Drivers”, Test & Measurement World, December 1991, pg 51.