

# Device/User Interface Software Requirements For AVTEC 1001 Programmable Telemetry Processor

Version 1.0

October 1, 1997

**Submitted by:** \_\_\_\_\_

Software Engineer

Date

**Approvals:** \_\_\_\_\_

Hardware Engineer

Date

Operations

Date

Project Lead

Date

*Final and Signed  
11/19/97*

---

## Table of Contents

Table of Contents .....	i
1.0 Introduction .....	1
2.0 Required Functionality .....	1
3.0 Parameter Ranges .....	1
4.0 Communications Protocol .....	1
5.0 GUI Functionality .....	1
6.0 Command Scripting .....	1
7.0 High-level Status .....	1
8.0 Replacement Algorithm .....	3
Appendix A: Graphical User Interface Requirements .....	4
Appendix B: Scripting Requirements .....	5

## 1.0 Introduction

This document provides device and user interface requirements for the Avtec 1001 Programmable Telemetry Processor.

## 2.0 Required Functionality

The AVTEC Programmable Telemetry Processor or PTP (a PC-based, multi-channel telemetry and command processing system) is a device within the *Recorder* node. The PTP's unique data handling capabilities are ideal for a telemetry front-end system that performs data acquisition, real-time network transfer, and store and forward operations. Support for both time-division multiplexed TDM and CCSDS telemetry formats provides the flexibility to support multiple spacecraft with a single front-end system.

The PTP acts as a gateway that accepts multiple telemetry streams and outputs time-tagged frame or packet data over a network to workstations in a distributed satellite control and analysis system. The PTP also includes a command gateway that accepts input from the network and outputs serial commands to the uplink. Key elements of the PTP system are AVTEC's MONARCH-E PCI-based CCSDS/TDM Telemetry Processor/Simulator board, a serial input AT-HSIO2 card, a network-based, distributed computing architecture, and the WindowsNT operating system.

## 3.0 Parameter Ranges

The range of status parameters is to-be-determined through operator testing.

## 4.0 Communications Protocol

Unlike other RS-232 dependent devices, the PTP can be controlled locally or remotely via the network using a TCP/IP socket interface. The PTP\_NT remote control and monitoring interface consists of two network sockets. A TCP/IP socket server provides a bi-directional byte stream connection between a single controller and the PTP system. An IP multicast socket is used for periodic status broadcast so that multiple host machines can receive from the multicast address to monitor the status of the PTP system.

The PC-based system runs under the WindowsNT operating system and contains PCI and ISA I/O busses as well as the standard PC peripherals and application specific interfaces. The PTP software controls and monitors all aspects of the PC configuration.

## 5.0 GUI Functionality

See Appendix A: Graphical User Interface Requirements.

## 6.0 Command Scripting

See Appendix B: Scripting Requirements.

## 7.0 High-level Status

The PTP is capable of displaying status from components operating within three different module

types, *auxiliary I/O*, *data I/O* and *Data Processing*. The following status parameters will be made available for monitoring and control:

Auxiliary I/O: None

Data I/O:

Serial Input (AT-HSIO2 or MONARCH-E Frame Synchronizer)

- Receiver state
- Count of frames received
- Count of dropouts

Socket (UDP, UDP multicast, TCP client, TCP server)

- Number of bytes read
- Number of read errors
- Number of bytes written
- Number of connections
- Number of current users
- Number of connection retries

IPDU Receiver

- Number of bytes read
- Number of read errors
- Number of bytes written
- Number of write errors
- Number of connections
- Number connection retries
- Number of headers received
- Number of headers accepted
- Number of data units sent
- Number of echoes sent

Data Processing:

Bit Error Rate Tester

- Injected error count
- Good frames received since last error
- Frames checked good
- Frames dropped during sync process
- Total number of bit errors
- Number of frames with bit errors
- Frames received with errors
- Address of bad buffer containing bit errors
- Bit number of first bit error

CCSDS Virtual Channel Processor

- Total transfer frames received
- Total transfer frames accepted and transmitted
- Total number of frames with VC sequence errors
- Total number of frames with CRC errors
- Total number of frames with correctable RS errors
- Total number of frames with uncorrectable RS errors

CCSDS Virtual Channel Simulator

- Frame count on each virtual channel ID for VCs 0-7 and 63

**Packet Processor**

- Packet Application ID (APID)
- Packet length of last packet with specified APID
- Sequence error count in last packet with specified APID
- Packet count containing specified APID
- Last packet ID received

**8.0 Replacement Algorithm**

To-be-determined through operator testing.

## Appendix A: Graphical User Interface Requirements

The user will be able to access the following remote controllable features.

### **ConnectToServer**

Establishes socket connection to a PTP server

### **DisableAllModules**

Disables all loaded modules

### **EnableAllModules**

Enables all loaded modules

### **DisableModule**

Disables the selected loaded module

### **EnableAllModules**

Enables the selected loaded module

### **GetConfiguration**

Retrieves the current PTP configuration

### **GetModuleState**

Retrieves the current state of a module

### **ListServers\***

Retrieves a list of live PTP servers

### **LoadProfile**

Loads a previously saved configuration from the PTP's disk

### **NoOp**

Places call to PTP server to verify socket and server still alive

### **QuerySavedProfiles**

Retrieves a list of previously saved configurations

### **ZeroAllModuleCounters**

Zeroes the status counters in all modules

### **ZeroModuleCounters**

Zeroes the status counters in a particular instance of a module

**Appendix B: Scripting Requirements**

<b>Master</b>	<b>Node</b>	<b>Comments/Error Handling</b>
Resource Request Specific Parameter: unit number	Start  Check allocation table for unit number Connect to Server Load Desktop Check connection  If available then Mark unit as assigned to this Master Reply " Unit # assigned" Open log file Retrieve configuration file from this Master  Else Reply " Unit # not available"  End  Stop	
Resource Request General	Start  Check allocation table for an available unit using the least recently used method Connect to Server Load Desktop Check connection  If available then Mark unit as assigned to this Master Reply " Unit # assigned" Open log file Retrieve configuration file from this Master  Else Reply " No units available"  End	

Master	Node	Comments/Error Handling
	Stop	
Setup Parameter: unit number	Start  Verify possession of unit by this Master  If not assigned to this Master then Inform this Master Stop End  Load and Verify configuration file  If configuration file error then Inform this Master Stop End  Stop	>> Operator intervention required          >> Operator intervention required
Start Support Parameter: unit number	Start  Verify possession of unit by this Master  If not assigned to this Master then Inform this Master Stop End  Poll for connection, serial input state  If status changes then Report status change End  Stop	>> Operator intervention required
Stop Support Parameter: unit number	Start  Verify possession of unit by this Master	

<b>Master</b>	<b>Node</b>	<b>Comments/Error Handling</b>
	If not assigned to this Master then Inform this Master Stop End  Stop	>> Operator intervention required
Takedown Parameter: unit number	Start  Verify possession of unit by this Master  If not assigned to this Master then Inform this Master Stop End  Mark unit as not assigned Close log file Send log file to this Master  Stop	>> Operator intervention required