ARA Smart Power System
User Manual

September 20, 2017

1 Introduction

2 Power Box

3 DAQ Box

3.1 Main control page

The main ASPS-DAQ microcontroller presents a web page that allows monitoring/control of all outputs in the DAQ box, with the exception of the fiber transceiver, for obvious reasons. The fiber transceiver *can* be disabled through the USB serial command interface.

3.2 Ethernet Serial Servers

The main ASPS-DAQ microcontroller presents serial servers on several ports.

- Port 23: ASPS-DAQ Heater debug/reprogramming port
- Port 24: ASPS-Power serial port
- Port 25: SBC serial port
- Port 26: ASPS-DAQ Heater command/monitoring interface

To access any of these ports simply telnet to them.

Note, however, that many Telnet clients automatically echo normally, and so therefore some care is needed to avoid any hassles. In addition, the Console Redirection in the SBC's BIOS uses VT100-style function keys, so some configuration is needed.

At the current time, the most appropriate Telnet client for ASPS-DAQ is PuTTY. In this case, the following options need to be set:

- Under "Terminal," select "Force off" for Local echo.

- Under "Keyboard," select VT100+ for the Function keys and keypad setting.

With these settings, the complete boot process for the SBC can be viewed on port 25, and remote BIOS access is possible.

3.2.1 Resetting serial servers

Only 1 user can be connected to a serial server at a time. The status of the serial servers can be seen under the "serial.html" page (e.g. http://ip.address.here/serial.html), and a serial server which is connected to an unknown client (or a client which failed to close the connection somehow) can be forcibly disconnected.

3.3 ASPS-DAQ Heater

The heater section of ASPS-DAQ serves as a temperature watchdog. At power on, it holds off activating the remainder of the system until the temperature has reached a specified target (typically -40 deg C) via the use of an onboard adjustable heater. It is the only section of the ASPS-DAQ board which is required to operate to -55 deg C.

Communication with the ASPS-DAQ heater is done via a serial connection through the main ASPS-DAQ microcontroller, which presents it on a specified TCP port. It should be noted it is *not* possible to communicate with the ASPS-DAQ heater until the main power has been activated. Therefore, any changes to the "autonomous behavior" parameters should be done with *extreme* caution.

All communication is done via JSON packets.

There is *also* a secondary serial port interface to the main ASPS-DAQ microcontroller, used for debugging and bootstrap reprogramming. This is also presented on a specified TCP port. Entering the device into bootstrap mode is done via the Web interface.

3.3.1 LED behavior

At power-on, the red LED by the heater will blink 2 times, indicating power-on cycle behavior. This is a useful thing to note if seen at any other time, because at power-on, the heater initially *disables* the main +15 V rail, which shuts down the station completely, before running through the decision tree as to whether or not to power on. (A normal reset does not cause this behavior - it is only caused by an initial power-on).

At any other time, the red LED indicates that the *PID controller is working to find the proper current.* The PID controller is extremely fast, which means that the red LED will only be on briefly once the current starts to ramp up if the heater is on.

The green LED indicates that the target temperature has been reached and everything is OK.

3.3.2 Monitoring

The ASPS-DAQ heater puts out a constant stream of monitoring data. Each JSON key contains a specific group of data.

PID data ("pid" key) This key contains an array of (in order) the setpoint, input, and output of the heater current PID loop. These values are proportional to the current flowing through the heater. The scale is roughly 1 mA (e.g. setpoint $500 \simeq 500\,\mathrm{mA}$).

Temperature data ("temps" key) This key contains an array containing only the local (microcontroller) temperature at the current moment. Remote temperature will be added in the future.

Voltage data ("volts" key) This key contains an array containing the input voltage, and the +15 V voltage. Note that these two should be close to identical if the system is on.

3.3.3 Current control

The setpoint for the heater (which returns to 0 once the system is above temperature for the wait period) can be modified via a "current" key- that is, a JSON packet with a key of "current" and a value equal to the desired setpoint.

3.3.4 Heater parameters

The "autonomous behavior" parameters can be queried and altered with the "heaterparams" key. Querying is done by sending a JSON packet with a "heaterparams" key and an empty value (or an array with less than 2 entries). The heater parameters are

Index	Description	Units	Default
0	Minimum turn-on temperature relative to $-60\,\mathrm{degC}$	deg C	20
1	Heater current when below temp	mA	500
2	Maximum wait time below temp	sec	7200
3	Wait time after target temp reached	0.5 sec	600
4	P-term in PID	128	
5	I-term in PID	64	
6	D-term in PID	0	

Altering these values is done by sending a JSON packet with a key of "heaterparams", and an array containing the index to alter (0-6), followed by the new value. Note that changing the P, I, or D terms could result in very bad behavior, and these changes are preserved over power cycles!