ALPHA DATA
BEAM

CERN Trajectory Measurement System

TmsPuServer

Project	CERN-TMS
Date	2008-01-222
Reference	Cern-tms/TmsPuServer
Version	1.2.5
Author	Dr Terry Barnaby

Table of Contents

1. Introduction

This document covers the operation of the TmsPuServer program that runs on the Module Controllers of the TMS. The program provides control and allows data to be read from the individual, FPGA based, Pick-up processing engines. The TmsPuServer program runs with real-time process priorities and is normally started at boot time by a system start-up script.

2. Starting the TmsPuServer program

Normally the TmsPuServer program is started on the Module Controller by the init program. The init configuration file "/etc/inittab" is used to perform this. The system is configured to keep the TmsPuServer program running.

For debug purposes it can be disabled by editing the /etc/inittab file for the Module controller concerned. Note that this will need to be done on the System Controller as the individual Module Controllers mount their root files systems read only. The root file systems for the module controllers are in the directories: /usr/tms/rootfs-?.

Once the system has been configured to not restart the TmsPuServer on quit, you can start the TmsPuServer manually from the command line. The TmsPuServer program also provides the ability to start the program from the command line with various debug options set. When started from the command line the following command line options are provided:

-f	Starts the program in the foreground. By default the program is started as a background task.
-d 0x03	Sets the debug options. The debug options are a bit mask of possible debug enables. These are listed below.

Debug bit values

0x000001	Standard. Provides basic debug print out.
0x000002	Commands. Prints out all of the commands received.
0x000004	Cycles. Displays all events related to a processing cycle

0x000008	Pupe. Dispalys debug for the PUPE board interface
0x000010	Event debug
0x000100	Resource debug
0x001000	Threads. Threads usage.
0x010000	Fpga access
0x020000	Fpga timing debug
0x040000	Interrupts debug

3. Configuration

The TmsPuServer has a simple ASCII configuration file, /etc/TmsPuServer.conf to configure its default state. This file can be edited using a conventional text editor. Its contents are as follows:

Parameter	*Default*	*Description*
TmsServer:	tmssc.tmsnet	The BOAP Servers host name. Normally the System Controllers host name on the TmsNet.
ModuleControllerNumber:	1	The number of the Module Controller
SimulateFpga:	0	If set to 1, the TmsPuServer will simulate the PUPE FPGA boards internally. This is useful for debug without using any PUPE engine boards.
SimulateTiming:	0x00	Simulate Timing signals in software. Bit mask (0xFF all timing signals)
Watchdog:	120	Sets a hardware watchdog timer to reset the system if the TmsPuServer or the system hangs. If this is set to 0 the Watchdog is disabled. If the SIGINT and SIGTERM signals are used to abort the TmsPuServer the Watchdog timer will be stopped. Otherwise the Watchdog timer will continue and if not pinged, will reset the system.
FpgaFirmwareFile:	/usr/tms/fpga/tms-fpga.bit	This is the path name for the FPGA bit file to use for the PUPE boards.
FpgaLclk	50	The is the PUPE LCLK frequency to use
FpgaMclk:	125	The is the PUPE MCLK frequency to use
PupeNumber:	5	Defines the number of PUPE boards
PupeMaster:	5	Defines the PUPE board that has the master timing inputs
PupePhysicalOn:	1	Use physical slot locations
PupePhysicalDevices:	10,11,12,13,14	The list of PCI device numbers

4. Overall Operation

On startup the TmsPuServer will initialise its internal state and then publish its BOAP API to the Boap server running on the system who's host name is given in the configuration files “TmsServer” parameter. It will then attempt to initialise all of the PUPE boards found on its local bus. It will then connect to the

master TmsServer which will initiate and initialisation of the State/Phase tables etc.

The TmsPuServer will now listen for API requests from the TmsServer and perhaps other clients and service them as required.

The TmsPuServer programs can detach and re-attach themselves to the TmsServer as required. This allows the system to power up in any sequence and also allows the Module Controllers to be re-booted if required. Any access to a PU channel that is not currently available, will return an appropriate error.

5. Errors

The TmsPuServer will handle errors in one of two ways.

- The individual API calls will return an errors as appropriate.
- The TmsPuServer will send an error event to the client.

6. Logging

While in operation all warnings and notices will be written to the systems standard logging daemon. The standard logging daemon has been configured to send these messages to the System Controller. These messages will thus normally appear in the /var/log/messages file on the System Controller.