Salamander: Analysis Software for the slow control information from the TT burn-in stand

M. Needham
Physik-Institut der Universität Zürich

July 26, 2005

1 Introduction

All slow control information from the TT burn-in stand is written to one file once per 'run'. This short note documents the data format and corresponding analysis software for this information.

2 Data Format

The file written out by labview is a plain ASCII file with a csv format. The format is given below. All the data for each run is contained between a header (**sob**) and a footer (**eob**). A timestamp (time since the start of the unix epoque) marks when the data was taken. Each 'entity' read out is given a tag that identifies what it is (for example a temperature sensor). This is followed (if necessary) by a unique identifer and then the data that is to be stored. The tags and values are separated by commas.

```
sob
timestamp,  [Seconds from start of the Unix epoque]
tprobe, [id] ,[Temp], [Humidity], [box], [location]
hv,  [Module], [Channel] , [Voltage] , [Current]
```

```
times, [FineDelay1], [FineDelay2]
beetle, [id], [vfs], [latency]
thybrid, [id], [temp]
chiller, [temp]
eob
```

The sob, timestamp and eob are required, all other tags are optional.

3 Analysis Framework

For analysis a set of C++ classes has been developed. Online doxygen documentation can be found at *http://ckm.physik.unizh.ch/tt/readout/salamander/*. The analysis software is located in:

```
/home/hep/lhcb/burn/salamander
```

together with a Makefile for compilation. By default the Makefile produces an executable called salamander. This can then be used to process the data. The executable has a built in command line parser [1]. To find out the possible arguements type:

```
./salamander --help
```

Two possibilities are forseen for data access: reading from a file and reading a list of files. To read a simple file and write out a root summary file:

```
./salamander -f run.csv -o output.root
```

To read from a list:

```
./salamander -l run.list -o output.root
```

The root file contains a set of summary plots.

4 Code structure

The main processing work is done by the **runManager**. This class builds a list of runs to be processed and then makes a set of **scRun** objects. Each

[1] tclap available from www.sourceforge.net

scRun object contains all the objects, corresponding to tags in the csv file, found for that run. From these objects the summary plots are created. It should be noted that system is totally dynamic: objects are built that correspond to the tags actually found in the input files, only the header, footer and timestamp tags are required.

5 Summary plots

The code makes a set of summary plots (root **TGraphs**). Currently, four sets of plots can be made:

IV Current vs voltage for each HV channel

Temperature Temperature and humidity for each probe

ITime Current drawn by each HV channel as a function of time

ITemperarature Current drawn by each HV channel as a function of the ambient box temperature and humidity

Which sets of plots are made is configured using the file 'slow.cfg'. If the tag used above is present and not commented out with a C++ style leading // then the plots will be produced. For example the file below would produced the IV and temperature plots.

```
IV
Temperature
//ITime
//ITemperature
```

A Data class description

All the data classes corresponding to tags derive from a common interface SObject. The interface has defines one pure virtual function 'streamer' that each of the derived classes should implement.

class: beetle

Derived from: SObject

Purpose: Beetle chip settings

class: beetle

Derived from: SObject

Purpose: Beetle chip settings

class: chiller

Derived from: SObject

Purpose: Chiller parameters

class: hvChannel

Derived from: SObject

Purpose: Information for one High voltage channel

class: tProbe

Derived from: SObject

Purpose: Temperature probe

class: TTC

Derived from: SObject

Purpose: TTC system parameters

The above objects are managed by the class **scRun**.

class: scRun

Derived from: —

Purpose: Manage the slow control objects