Online Data Conversion for the LHCb Active Radiation Monitor System (ARMS)

Julia Farrugia (University of Malta)
Supervised by Matthias Karacson (CERN)
Co-supervised by Luis Granado Cardoso (CERN)

Active radiation monitors are distributed in and around the LHCb experiment in order to monitor the evolution of radiation dose and 1 MeV equivalent neutron fluence. The active monitors are connected to an online network and their raw voltage measurement can be read out via WinCC panels. However, the raw voltage measurements must be corrected before conversion into correct dose or fluence. Previous work by [1] included a signal correction algorithm that corrects the values offline. The aim of this work is to understand how the signal correction algorithm works, archive previously corrected values into a new database system that will be integrated to a WinCC graphical interface and modify the algorithm so that it may process new values online while using less time and computing resources.

I. INTRODUCTION

Studying the radiation field in a high energy physics experiment such as LHCb is very important since the performance of all detectors, sub-detectors and related electronics is sensitive to radiation. Hence, an understanding of the evolution of the radiation environment is important for maintenance and operation of the detectors. To study the radiation field, the following dosimetric quantities are measured:

- Dose [Gy] – energy released in J by radiation in 1kg of matter
- 1 MeV neutron equivalent fluence [cm^{-2}] – fluence of particles that is equivalent to the fluence 1MeV neutrons crossing an area of $1cm^2$

These quantities are measured through 28 active radiation monitors, which are located in the LHCb Cavern and inside the detector. There are 4 types of sensors in each active radiation monitor, two which measure the dose (REM, LAAS) and two which measure the fluence (CMRP, BPW). Additionally, each PCB features an NTC temperature sensor which is used for temperature corrections of the sensor data. For a more detailed description of the monitors, see [2].

Previous work done by [1] includes a signal correction algorithm that corrects the raw voltage values in order to get the correct dose or fluence reading.

The first section of the report shall describe the correction algorithm in detail, and the second section describes the work carried out by the summer student. The latter section is subdivided into four sections; the extraction of the previously processed values, considerations that needed to be taken into account for the proposed system, a description of the proposed method for online correction, and the testing results. A final section will describe the future work that may be carried out.

II. SIGNAL CORRECTION ALGORITHM

The signal correction algorithm is an offline routine that is implemented using ROOT. In order to achieve the best results, the original algorithm needs to run on the whole dataset available, which is time and computationally inefficient. The raw voltage values and luminosity values are read in from text files, and additional parameters are also stored in this manner. The values which this algorithm corrected are from the first data taken in 2010 until the beginning of LS1.

The first part of the algorithm divides the date and the sensor data into different files, titled (date.txt) and (values.txt). According to the sensor number, temperature correction parameters, drift correction and smoothing parameters, and time parameters are loaded. In the case of sensor number 22, there are no drift correction and smoothing parameters because it is considered defective.

The rest of the signal correction algorithm can be loosely divided into 5 sections:

A. Loading of luminosity values
B. Temperature and offset correction
C. Conversion into dose and fluence
D. Noise and spike correction
E. Drift correction

A. Loading of luminosity values

Before the luminosity values are loaded into the system, the timestamps are checked to see if they are too near, or not in the correct order. If the timestamp of the current value is greater than the previous timestamp of the read-in value, then the point is discarded. This will not be an issue when considering the new luminosity values for the new WinCC implementation since they have already been filtered.

The luminosity values are then plotted on a graph using ROOT, and will be made use of during the drift correction part of the algorithm.

B. Temperature and offset correction

The temperature is corrected using the temperature parameters that are loaded from an existing file, and the mean and sigma computed from the values themselves. Once the spikes and noise have been removed from the temperature, it can be used for the temperature and offset correction of the other voltage sensor values (REM, LAAS, CMRP, BPW).

From the time parameters loaded, an upper limit date for offset calculation is found. This date is then used to load the values such that the offset of each sensor can be calculated up to this value. This period is usually within the first days of

data readings, when the radiation field effects are still below the sensitivity of the sensors. It is necessary to calculate the offset since the signal starts from an initial value that is not zero. Converted values between 0 and 26 are considered to be good values that can be used to calculate the offset. These offset values are unique for each sensor.

In the case of sensors 12, 13, 37, 38 and 39, the luminosity values of 2010 need to be taken into account when calculating the offset. In addition, the LAAS offset of sensors 15 and 30 were corrected by hand.

After the offset has been calculated, the sensor voltage values are normalised to 20°C and are corrected using the temperature correction parameters.

C. *Conversion into dose and fluence*

After the raw voltage values have been corrected for temperature, they are then converted into dose and fluence according to parameters defined in [2].

D. *Noise and spike correction*

The noise and spike correction are dependent on the step size S, and the number of "tolerated" standard deviation $N\sigma$. These values were found through empirical analysis in [1]. To begin the correction, the mean of the first S converted values are calculated, and the first S values are then set to this calculated mean as a start off point. The noise and spike correction can begin after this step based on these initial values.

To correct a given point (t_i) , the mean μ and the standard deviation σ is calculated within the range $P(t_{i-S/2})$ up to $P(t_{i+S/2})$. After the mean and the standard deviation are obtained, the following conditions are checked:

- If $|P(t_i) - \mu| < N\sigma$, the point $P(t_i)$ is considered to be a "good" point so it is saved
- If $|P(t_i) - \mu| > N\sigma$, the point $P(t_i)$ is considered to be a "bad" point so it is flagged

After all the "good" points have been distinguished, a simple linear interpolation is used to correct the flagged values. The interpolation needs a previous good value and the next good value for a successful interpolation. If a next good value is not found, interpolation does not take place and all the values are set to zero.

E. *Drift Correction*

The drift correction corrects for the annealing effects of the sensors using luminosity data to distinguish between beam ON/OFF periods. For this method of drift correction it is assumed that the annealing only occurs significantly during periods without collisions. Unfortunately, it has been found that the annealing also influences the measurements during collisions. However, there is no accurate solution to account for these effects while the measurement signal increases, so it is left uncorrected at this time. Fig.1. describes the flow of the algorithm.

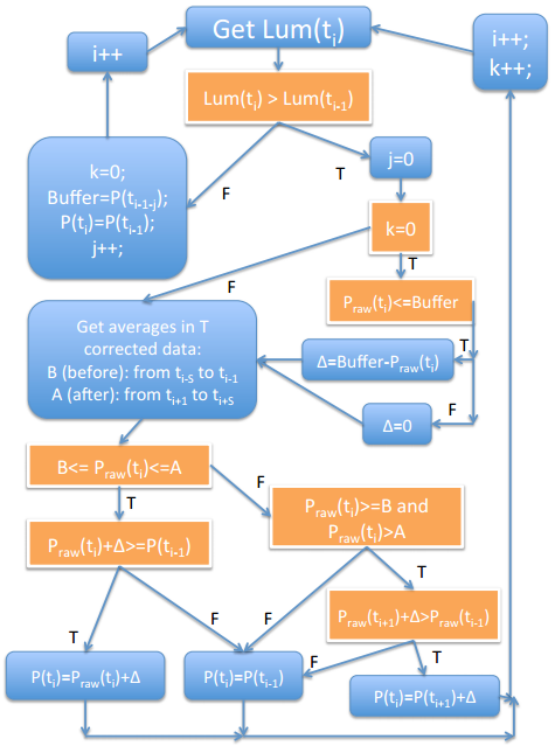

Fig.1. The starting point is "$Get\ Lum(t_i)$", i.e. the delivered luminosity at time t_i. $P_{raw}(t_i)$ is the temperature corrected data point (dose or fluence), while $P(t_i)$ is the drift and drops corrected point. Δ is the loss of signal during beam off period; "Buffer" is the stored value of the signal before the beam is switched off [1].

III. Extraction of Historical Values

The previously corrected values were saved as plots in ROOT. Hence, a short function was implemented using ROOT to extract the values from the plots that are stored in the ROOT file with the final corrected values. The extracted values were saved to a text file. The values included different calibration curves based on the same initial voltage measurements: the REM($< 40Gy$), REM($> 40Gy$), LAAS(HEH), LAAS(PPh), CMRP, BPW and the NTC. In addition, the corresponding dates and the times of measurements were also saved. The values in the text files were then loaded into WinCC to be stored in a datapoint of a database system provided by CERN.

In the future, it may be needed to display a custom calibration value for the LAAS sensors. At the moment, a value is set to the calculated average between the LAAS (HEH) and the LAAS(PPh) values, which can be modified easily as soon as a fitting calibration is found. To achieve this, a WinCC panel was created so that it could display the average at the time displayed in the plot.

IV. Considerations for Online Conversion

The proposed method will work in WinCC to take advantage of its capability of storing data in datapoints, thus removing the dependability from text files. The challenges of implementing an online algorithm were mainly due to the following factors:

- There is approximately a 20 minute delay between readings from the sensors. Since the correction algorithm works on a range of values, it is necessary to wait until the end of a day so that there are sufficient values for the algorithm to work with.
- Within the noise and spike correction algorithm, to correct a given point (t_i) , the mean σ and the standard deviation μ is calculated within the range $P(t_{i-S/2})$ up to $P(t_{i+S/2})$. Hence, it was necessary to find a method to fetch previous values from already corrected datapoints so that they can be used for the correction of a given point.
- The spike and noise correction is dependent on the step size S which was found through empirical analysis in [1]. The step sizes range from a step size of 40 to a step size of 5000. In a day, approximately 70 values are gathered by the sensors, so a different solution will need to be found for sensors which need a step size of 5000.
- The drift correction relies on a very good noise and spike correction. Hence, it is necessary to wait for a "good" point before correction can take place, so that the drift correction is not affected. The probability of finding a "good" point to work with depends on the sensor – for the REM and the LAAS sensors it is estimated that at least one good value is found within a day. However, for the CMRP and the BPW it may be necessary to wait for a longer period of time.
- In addition, the drift correction relies on past T corrected values for correction. Again, it was necessary to find a method to fetch previously corrected values so that they can be used for the correction of a given point.

In light of the above points, the proposed system will require that the data between 2010 and LS1 is processed so that the relevant past values can be saved, and new values can use this past data for processing.

V. Proposed System for Online Conversion

The proposed system eliminates the dependency on text files by storing the data in datapoints. For each text file, a new datapoint was created and stores the values in the same structure as the text file. In addition, the offset value is calculated once and stored in a datapoint. If the user would like to change a value and still remember the past value, then archiving must be set so that the past value is remembered. A table with the list of datapoints and their description may be found in Table 1.

With these parameters in the datapoints, the correction algorithm can begin. Raw voltage values are loaded from a last determined "good" point until the end of the current day. If a "good" point is not found, the correction algorithm cannot continue, so the algorithm will have to wait until more values come in before it can begin correction. Hence, the conversion is dependent on whether a "good" point is found in a day.

After the values have been loaded, the interpolation can take place, and following this the drift correction can begin. Due to the way the drift correction algorithm works, if N values have been T corrected, then the drift correction will correct $N - 5$ values. These five values will then be corrected when the correction algorithm is activated again. Four of these values will be loaded from a data point, while the fifth value is the "good" value from which the correction algorithm begins loading.

Datapoint type	Datapoint name	Description
lbRadmonTparam	<snNo>/Tparam	T correction values for: -REM -LAAS -BPW -CMRP - Nσ NTC
lbRadmonSmooth	<snNo>/smoothParam	Smoothing parameters: –Nσ REM - Nσ LAAS - Nσ BPW - Nσ CMRP - step size S
ldRadmonOffset	<snNo>/Offset	Offset values for -REM -LAAS -CMRP -BPW In addition, there are the dates between which the offset was calculated

Table 1. Datapoints used to start the correction algorithm.

The proposed system makes use of a datapoint which stores the time at which the points should be loaded, as well as parameters that are relevant to the drift correction.

The datapoint has the following structure:

- lastGoodTime_center: this is a time field which stores the time at which the values must the fetched. Since the interpolation is triggered by a "good" point, the time corresponds to the time of $P(t_{i-S/2})$, such that the point $P(t_i)$ is corrected and hence, a "good value".
- lastGoodTime: this is a time field which stores the date of the "good" point $P(t_i)$

The following fields in the datapoint are vital for the drift correction:

- buffer: this contains the last drift corrected value before the beam off period.
- delta: this contains the loss of signal during the beam off period
- j: this is a counter which keeps a reference to the location of the value that is in the buffer
- k: this is a flag that signals when the beam has just been turned off
- lastVal: this the last drift corrected value. It is needed since the algorithm may need to look back at the previous drift corrected value

- ValAV.1 – 9: these datapoints contain the last 9 T and spike corrected values. Points 1 to 5 are required in case the beam is on and the average before the first point needs to be calculated (see Fig.1.). Points 4 to 9 are T and spike corrected values that need to be corrected in addition to the last "good" point.

With each iteration, the datapoint updates itself with new values. It is for this reason that it is necessary to process the values between 2010 and LS1 once more with the newly implemented WinCC routines, so that the relevant past values can be saved, and new values can use this past data for correction.

VI. Testing and Results

Testing of the WinCC routines was carried out on the 2010 – LS1 values, and the values were cross-checked with the values from the original correction algorithm in ROOT to ensure that the value which is being calculated matches. Since testing is a time-consuming process due to the large amount of values that need to be drift corrected, three sensors were used for testing: sensors 11, 13, and 27. Sensors 11 and 13 have a step size of 40, while sensor 27 has a step size of 50.

Sensor 11 matched the original values of the correction algorithm accurately; however sensor 13 yielded a small deviation during the correction of the BPW values. This deviation soon corrected itself, so the error is most likely due to the difference in the way the luminosity is calculated on both platforms. In the original ROOT script, the luminosity is determined through a native ROOT function called Eval. However in WinCC this functionality does not exist, therefore this it had to be implemented using WinCC tools.

Sensor 27 yielded accurate results for all instances except one. For the BPW correction, an unforeseen problem was revealed in the proposed system – no "good" values were detected because the sensor seems to be defective. Consequently, the datapoint in the WinCC routine was not loaded with the necessary parameters, which means the correction algorithm could not take place. In the original ROOT script it was found that in such a scenario, all the values are set to 0.

VII. Future Work

From the test results, it may be deduced that the proposed system requires further testing before it can be implemented. Since the project was conceived to work as an online system, the possibility of using a smaller step size during the spike and noise correction must be investigated.

In addition, a scenario such as no "good" points found in a day hinders the working of the proposed system, hence a way to mitigate this problem must be found. One possible solution might be to widen the definition of a "good point", meaning that a little more noise is deemed acceptable, without inducing any significant loss on the overall accuracy.

VIII. Conclusion

This work was successful in extracting the converted values between 2010 and LS1, and the original ROOT correction algorithm was successfully implemented in WinCC. The routine in WinCC retains the same functionality as the ROOT script, despite the fact that certain functions that were available in ROOT were not available in WinCC. In addition, this work has shown that it is possible to use the existing correction algorithm for online data conversion. However, further testing must be carried out, so that all possible known scenarios can be investigated and mitigated.

References

[1] V. Battista, "Active Radiation Monitors in LHCb", CERN-SUMMER-STUDENTS-2012. 28th Sept. 2012.

[2] F. Ravotti, "Development and Characterisation of Radiation Monitoring Sensors for the High Energy Physics Experiments of the CERN LHC Accelerator", CERN-THESIS-2007-013. 17th Nov. 2006.