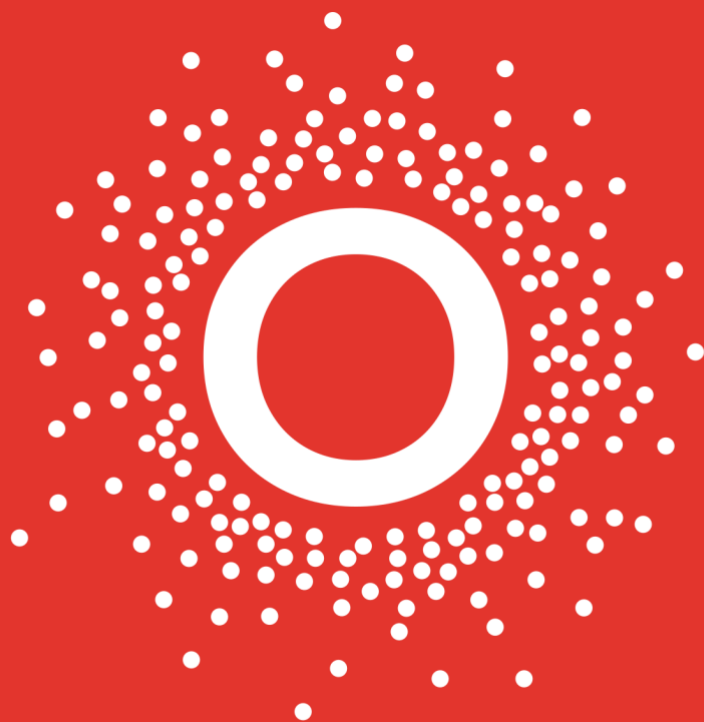


USER MANUAL

BCM-RF-DAQ

Beam Charge Monitor – RF Digitizer



www.bergoz.com

bergoz[™]
INSTRUMENTATION

More than 40 years of experience recognized in the world of particle accelerators

Record of updates

Version	Date	Updates performed
1.0	07/2024	First public release

DISTRIBUTORS

U.S.A.

GMW Associates

GMW Associates
www.gmw.com
sales@gmw.com

Japan

HR HAYASHI-REPIC

Hayashi-Repic Co., Ltd.
www.h-repic.co.jp
sales@h-repic.co.jp

India

GEEBEE
INTERNATIONAL

GEEBEE International
www.geebinternational.com
info@geebinternational.com

China

CONVe-YI 北京科维泰信

Beijing Conveyi Limited
www.conveyi.com
sales@conveyi.com

South Korea


SEYOUNG

Seyoung Co., Ltd
www.seyoungsys.com
apark@seyoungsys.com

TABLE OF CONTENTS

INITIAL INSPECTION	2
WARRANTY	2
ASSISTANCE	2
SERVICE PROCEDURE.....	2
RETURN PROCEDURE	3
SAFETY INSTRUCTIONS	3
GENERAL DESCRIPTION	4
WIRING AND SETUP OF THE DAQ-BCM-RF	4
Front panel.....	4
Rear panel	5
Hardware tips	5
HOW TO USE THE DAQ-BCM-RF?	6
Use case 1: Serial SCPI communication.....	6
Use case 2: Python 3 library	8
Use case 3: EPICS - Bergoz Instruments PV Access software (BIPVA).....	10

INITIAL INSPECTION

It is recommended that the shipment be inspected immediately upon delivery. If it is damaged in any way, contact Bergoz Instrumentation or your local distributor. The content of the shipment should be compared to the items listed on the invoice. Any discrepancy should be notified to Bergoz Instrumentation or its local distributor immediately. Unless promptly notified, Bergoz Instrumentation will not be responsible for such discrepancies.

WARRANTY

Bergoz Instrumentation warrants its beam current monitors to operate within specifications under normal use for a period of 12 months from the date of shipment. Spares, repairs and replacement parts are warranted for 90 days. In exercising this warranty, Bergoz Instrumentation will repair, or at its option, replace any product returned to Bergoz Instrumentation or its local distributor within the warranty period, provided that the warrantor's examination discloses that the product is defective due to workmanship or materials and that the defect has not been caused by misuse, disassembly, neglect, use of faulty part, accident or abnormal conditions, repair made by the customer, or operations. Damages caused by ionizing radiations are specifically excluded from the warranty. Bergoz Instrumentation and its local distributors shall not be responsible for any consequential, incidental or special damages.

ASSISTANCE

Assistance in installation, use or calibration of Bergoz Instrumentation beam current monitors is available from Bergoz Instrumentation, 01630 Saint Genis Pouilly, France. It is recommended to send a detailed description of the problem by email to info@bergoz.com.

SERVICE PROCEDURE

Products requiring maintenance should be returned to Bergoz Instrumentation or its local distributor: The purchaser/customer must ask for a RMA (Return Material Authorization) number to Bergoz Instrumentation or its local distributor before return of goods. Bergoz Instrumentation will repair or replace any product under warranty at no charge.

For products in need of repair after the warranty period, Bergoz Instrumentation will assess the technical issue and send a quote to the purchaser/customer. The purchaser/customer must provide a purchase order before repairs can be initiated. Bergoz Instrumentation can issue fixed price quotations for most repairs.

RETURN PROCEDURE

All products returned for repair should include a detailed description of the defect or failure as well as name, phone number and email of a contact person to allow further inquiry. Contact Bergoz Instrumentation or your local distributor to determine where to return the product. Returns must be notified by email prior to shipment.

The shipment of a product under warranty or out of warranty back to the factory is paid by the user/customer, including the customs fees. The return of this repaired product under warranty back to the customer is paid by Bergoz Instrumentation.

Return of product out of warranty should be made prepaid or will be invoiced. Bergoz Instrumentation will not accept freight-collect shipments. Shipments should be made via UPS, FedEx or DHL. Within Europe, the transportation services offered by the national Post Offices can be used. The delivery charges or customs clearance charges arising from the use of other carriers will be charged to the customer.

SAFETY INSTRUCTIONS

This instrument is operated from the mains power supply. For safe operation, it must be grounded by way of the grounding conductor in the power cord. Use only the fuse specified. Do not remove cover panels while the instrument is powered. Do not operate the instrument without the cover panels properly installed.

Chassis originally shipped to U.S. or Canada feature AC mains power entry modules where the Phase is fused and the Neutral unfused, as is the rule.

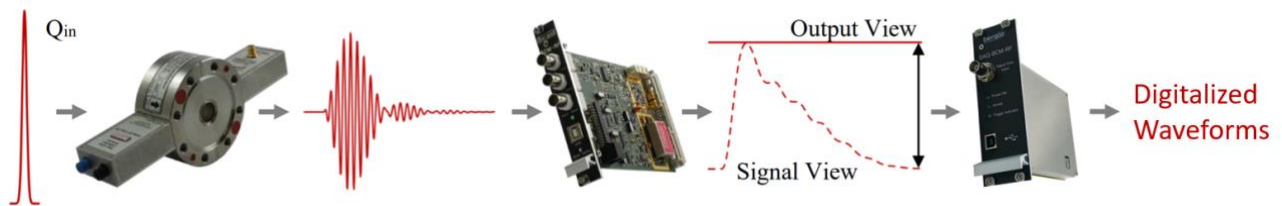
Chassis to other destinations but U.S. and Canada feature AC mains power entry modules where both Phase and Neutral are fused.

When a chassis with unfused Neutral shall be used outside the U.S. and Canada, fuse configuration must be modified so that both Phase and Neutral will be fused:

The Power entry module must be opened, the Phase fuse must be removed, the fuse holder must be flipped; its reverse side presents two slots where two new fuses must be inserted, one in each slot. The fuses rating must be same as the Phase fuse that was removed.

GENERAL DESCRIPTION

The DAQ-BCM-RF is a digitizer specifically designed to operate with the BCM-RF-E from Bergoz Instrumentation. Please refer to the Turbo-ICT and BCM-RF-E manual for more information on the BCM-RF-E.



The DAQ-BCM-RF has been specifically designed to acquire both “BCM Output” and “Signal View” signals from the BCM-RF-E electronics. It allows to perform digital measurements of the charge, mean current, and make acquisitions of the signal waveform as well. Here are its characteristics:

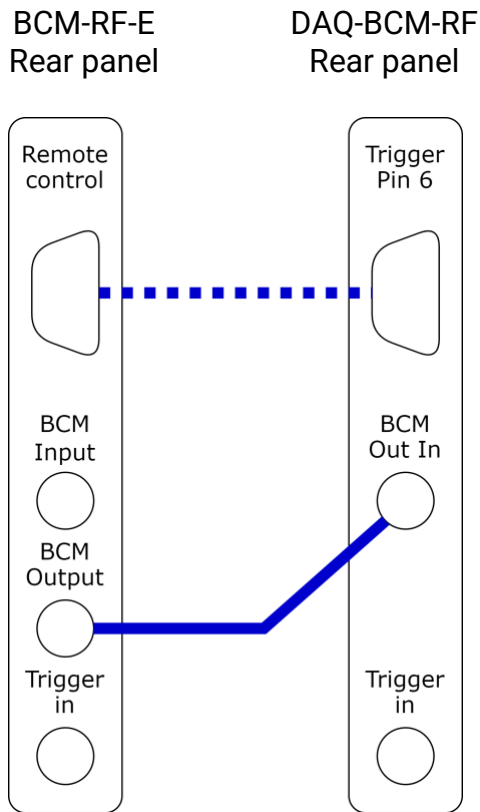
Number of inputs	2
Front input full scale	1V (50 Ω)
Rear input full scale	5V (High Z)
Resolution	16 bits on each input
Sampling rate	100 MS/s, simultaneous acquisition of each input
Acquisition length	Up to 100 μ s
SNR	65 dB typical
Digital interface	USB type B
Trigger inputs	DB9 (from the BCM-RF-E), SMA (external)

WIRING AND SETUP OF THE DAQ-BCM-RF

Front panel

- Use a BNC-to-BNC cable to connect the BCM-RF-E “Signal View” output to the DAQ-BCM-RF “Signal View” input (50 Ω / 1V full scale).
- Use a USB type B cable to connect the DAQ-BCM-RF to a computer.

Rear panel



When the BCM-RF-E is in Sample & Hold mode (charge measurement), a trigger is sent from pin 6 of the “Remote control” connector once the BCM Output is ready to be acquired. To use this trigger, a female-female Sub-D 9 cable can be used. If not used, you can plug a metallic Sub-D 9 dust cover on it.

“BCM Out In” is the calibrated input of the DAQ-BCM-RF matched to the BCM-RF-E output full-scale. Use a well shielded SMA cable as short as possible to connect them.

The SMA trigger input is used when the BCM-RF-E is in Track-Continuous mode. It is a high input impedance, LOW and HIGH thresholds are 1,5V and 3,5V respectively. The maximum input voltage is 5V.

Connect a 50 Ω load if not used.

Hardware tips

There are two ways to trigger an acquisition. Either through the DB9, when the BCM-RF is configured in Sample & Hold mode, or through the dedicated SMA which is left available to the user's choice. A minimum of 5 samples preceding the trigger are also acquired.

The DAQ-BCM-RF (like the BCM-RF-E) must be connected to a computer to be controlled and to acquire signals. For example, it is possible to use a rack-mountable computer located under the Bergoz Instrumentation chassis. In this case, always leave a few centimeters spacing above and below the chassis to allow an air flow cooling down the electronic components.

Prefer the use of short cables with good shielding, and do not hesitate to add chokes on cables to prevent common mode noise.

We recommend connecting a 50 Ω load or a short on all unused inputs.

HOW TO USE THE DAQ-BCM-RF?

Use case 1: Serial SCPI communication

To interact with the DAQ-BCM-RF, use a USB cable and the following communication protocol inspired by SCPI (Standard Commands for Programmable Instruments). Here is the list of commands that the DAQ can interpret:

IDN?	Ask for the DAQ-BCM-RF identification number.
ARM	Arm the DAQ-BCM-RF.
ARM?	Ask if the DAQ-BCM-RF is armed (0 or 1, return to 0 after an acquisition).
OUT? SIVI?	Ask for the last BCM Output waveform (or "Signal View" waveform for the "SIVI?" request). This is a sequence of raw samples <u>in hexadecimal format</u> , with a coma after each sample (","). The number of samples returned depends on the "acquisition length" setting which can varies from 99 to 99,999. To obtain the corresponding voltages (or charges and currents), refer to the Look Up Tables (LUTs) on the USB storage supplied with your device.
TIME?	Ask for the timestamp of the last acquired waveform (in microsecond (μ s) unit from power-on).
TRIG?	Ask for the trigger number of the last acquired waveform. Note that this value will not be equal to zero when the DAQ-BCM-RF is powered on.
LEN:=XXX	Set the length of the next waveform to acquire in microsecond unit (001..100, always send 3 digits).
LEN?	Ask for the current acquisition length (in μ s unit).
FAKE	Force an acquisition by generating an artificial trigger (not counted).

More information:

- Each command sent must end with a "line feed" (10th ASCII character, i.e., '\n').
- All DAQ responses end with a line feed.
- Otherwise noted, each value is in decimal format (base 10).
- The acquisition repetition rate is limited by the USB communication speed of the microcontroller. Considering the sampling rate and resolution, the transmission of the largest possible waveform can take almost a second.

Communication example		
<i>Message sent to the DAQ</i>	<i>Comment</i>	<i>Message returned by the DAQ</i>
IDN?\n		BCM-RF-DAQ:XX-XXX\n
	<i>“XX-XXX” is the serial number of the DAQ</i>	
LEN?\n		100\n
LEN:=026\n		
LEN?\n		26\n
ARM\n	<i>The “Armed” LED Lights up</i>	
ARM?\n		1\n
FAKE\n	<i>Simulates a trigger, without incrementing the trigger counter, the “Armed” LED goes off</i>	
ARM?\n		0\n
TRIG?\n		1708\n
TIME?\n		70681659\n
OUT?\n		31FC,3256,3272,[...],3229,31F4,\n
	<i>“[...]” represents the rest of the 26μs x 100 Samples/μs.</i>	

Use case 2: Python 3 library

With your device we provide you with a USB key containing Python 3 libraries to simplify the use of the DAQ-BCM-RF. The dependencies of these libraries are numpy and pyserial.

To start using it, simply import it and instantiate a DAQ_BCM_RF object with the COM port it is connected to as an argument:

```
>>> from DAQ_BCM_RF import DAQ_BCM_RF
>>> my_bergoz_daq = DAQ_BCM_RF("COM19")
```

Here is the list of main methods:

setAcquisitionLength	In: int : length	Out: None
Description	<i>Set the acquisition length in μs, between 1 and 100.</i>	
Example	>>> my_bergoz_daq.setAcquisitionLength(2)	

getAcquisitionLength	In: no args	Out: int
Description	<i>Ask for the current acquisition length setting in μs.</i>	
Example	>>> my_bergoz_daq.getAcquisitionLength() 2	

arm	In: no args	Out: None
Description	<i>Arm the DAQ.</i>	
Example	>>> my_bergoz_daq.arm()	

isArmed	In: no args	Out: boolean
Description	<i>Ask if the DAQ is armed.</i>	
Example	>>> my_bergoz_daq.isArmed() True	

doFakeTrigger	In: no args	Out: None
Description	<i>Simulates a trigger, without incrementing the trigger counter.</i>	
Example	>>> my_bergoz_daq.doFakeTrigger()	

getLastTimestamp	In: no args	Out: int
Description	<i>Ask for the last acquisition timestamp in μs.</i>	
Example	>>> my_bergoz_daq.getLastTimestamp() 1150497076	

getLastTrigger	In: no args	Out: int
Description	<i>Ask for the last acquisition trigger number.</i>	
Example	>>> my_bergoz_daq.getLastTrigger() 3797	

getRawBcmOut	In: no args	Out: list of int
---------------------	-------------	------------------

getRawSignalView	
Description	<i>Ask for the last acquired "BCM Out In" or "Signal View Input" waveform in raw ADC values.</i>
Example	<pre>>>> my_bergoz_daq.getRawBcmOut() [12836, 12842, 12888, ...] >>> my_bergoz_daq.getRawSignalView() [12836, 12842, 12888, ...]</pre>

To convert the ADC raw values into the corresponding voltages, you can use the Look Up Tables (LUTs) present on the USB key coming with your device.

You can then also convert the voltage read at the BCM-RF output into the corresponding charge or current flowing through the Turbo-ICT using the equations provided with the calibration report, or the other LUTs.

The following methods allow you to directly apply the voltage LUTs during an acquisition:

setBcmOutLUT	In: list of numbers: adc_values, list	Out: None
setSignalViewLUT	of numbers: corresponding_voltages	
Description	<i>Stores the LUTs in the Python DAQ_BCM_RF object in order to automatically convert the ADC raw values to voltages.</i>	
Example	<pre>>>> import pandas >>> full_lut = pandas.read_csv("path_to_lut", sep="\t", header=0) >>> adc_vals = full_lut[full_lut.columns[0]] >>> volt_vals = full_lut[full_lut.columns[1]] >>> >>> my_bergoz_daq.setBcmOutLUT(adc_vals, volt_vals) >>> # Same process for setSignalViewLUT</pre>	

getBcmOut	In: no args	Out: numpy array of float
getSignalView		
Description	<i>Ask for the last acquired "BCM Out In" or "Signal View Input" waveform in volts. The answer is valid only if the corresponding LUT has been previously set.</i>	
Example	<pre>>>> my_bergoz_daq.getBcmOut() array([0.19586481, 0.19595636, ...]) >>> my_bergoz_daq.getSignalView() array([0.18069734, 0.18054475, ...])</pre>	

Use case 3: EPICS - Bergoz Instruments PV Access software (BIPVA)



The BIPVA software developed by Bergoz Instrumentation allows to make a "plug and play" interface between an EPICS 7 network and the DAQ-BCM-RF and its BCM-RF-E.

Installation procedure

To be able to run BIPVA, you must install Python 3 and some libraries. You can find installation instructions of Python on its official website: <https://www.python.org/>

The required Python libraries are:

- Numpy
- Pandas
- Serial (pyserial)
- Pvaccess (pvapy)

You can install them by using Pip in a terminal:

```
pip install numpy pandas serial pvapy
```

To run BIPVA, enter the following command in a terminal:


```
python ./BIPVA.py
```

Depending on your python installation, you may have to write "python3" instead of "python".

To stop it, use the keyboard shortcut [ctrl]+[c].

BIPVA must be launched after any modification of the "setup/BI.xml" file for it to be taken into account.

Here is a description of the elements present in the BIPVA folder provided on the USB key coming with your device. You can copy the BIPVA folder to your computer, making sure its content is not dissociated:

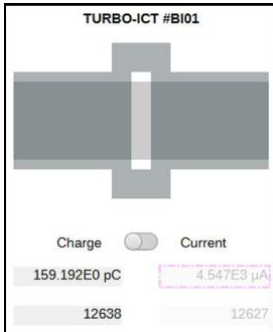
	<p>Bipva_src: Folder containing all the BIPVA subfunctions.</p> <p>Setup : Folder containing "BI.xml", a description of the Bergoz Instrumentation devices that are connected to the computer allowing the BIPVA to detect them. The LUTs of the instruments are usually also placed in this folder.</p> <p>BIPVA.py : Main program. After configuring the "setup/BI.xml" file, just run it with Python 3.</p>
--	---

On the following pages are presented two ways of using a Turbo-ICT, BCM-RF-E and DAQ-BCM-RF set with BIPVA.

- The first one is used to measure charges and average currents with a standard BCM-RF-E configuration.
- The second one is used to acquire current waveforms, e.g. to measure a charge when the accuracy of the trigger cannot be guaranteed for the BCM-RF-E. Please refer to the document "MEASURING CHARGE AND AVERAGE BEAM CURRENT USING TURBO-ICT AND BCM-RF-E IN TRACK-CONTINUOUS MODE" written by F. Stulle, Bergoz Instrumentation.

You can check the correct operation with the standard EPICS tools such as pvget, pvput and pvmonitor. To design a graphical interface, we recommend using CSS Phoebus.

TICT-BCM-DAQ_simple – Measure a charge or an average current



If you want to measure the charge of a single pulse, or the average current of a CW, you can configure a "TICT-BCM-DAQ_simple" to get an automatic measurement at each trigger sent to the DAQ.

The screenshot on the left is an example of implementation in an accelerator control room. The cross-section of a Turbo-ICT is schematized and a controllable switch allows the BCM-RF-E configuration. The last charge (or current) measurement and the corresponding trigger count are displayed.

To set up such an interface, add a "TICT-BCM-DAQ_simple" section to the "setup/BI.xml" file as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<bergoz_systems>
  <TICT-BCM-DAQ_simple
    pv_root="TICT-BI01"
    BCM-RF-E_idn="IDN=Bergoz,BCM-RF-E,SN BI01,FW 3.4"
    DAQ-BCM-RF_idn="DAQ-BCM-RF:BI01 - 228.1 - rev 0"
    LUT-SH="/home/user/BIPVA/setup/LUT-BCM-RF-E.SH.BI01.txt"
    LUT-TC="/home/user/BIPVA/setup/LUT-BCM-RF-E.TC.BI01.txt"
    LUT-DAQ-BCM-OUT="/home/user/BIPVA/setup/LUT-DAQ-BCM-RF.out.BI01.txt"
  />
</bergoz_systems>
```

Field	Description
pv_root	Root of the process variable names. In this example, the PVs will be: <ul style="list-style-type: none"> - TICT-BI01:tc_mode - TICT-BI01:charge - TICT-BI01:current
BCM-RF-E_idn DAQ-BCM-RF_idn	Identification numbers of the serial interfaces of the BCM-RF-E and the DAQ-BCM-RF.
LUT-SH LUT-TC LUT-DAQ-BCM-OUT	Look up tables of the BCM-RF-E in Sample and Hold mode, the BCM-RF-E in Track Continuous mode, and the BCM-OUT input of the DAQ-BCM-RF.

If you have properly configured the "setup/BI.xml" file, the following PVs will be accessible from your EPICS network after running BIPVA.py:

- The PV "[pv_root]:tc_mode" is a Boolean value that switches the system to current measurement (when set to True) or to charge measurement (when set to False).
- The PV "[pv_root]:charge" is a float that corresponds to the last charge measured, in pico-coulomb. This PV contains an alarm field which indicates if the measured value is out of specification, or if the PV tc_mode is not in the right state for this measurement.
Moreover, this PV contains a 'trigg' field (unsigned long) which indicates the trigger number of the last acquired value.
- The PV "[pv_root]:current" is structured and works in the same way as the "[pv_root]:charge" PV, but for current measurement. Its unit is the micro-ampere.

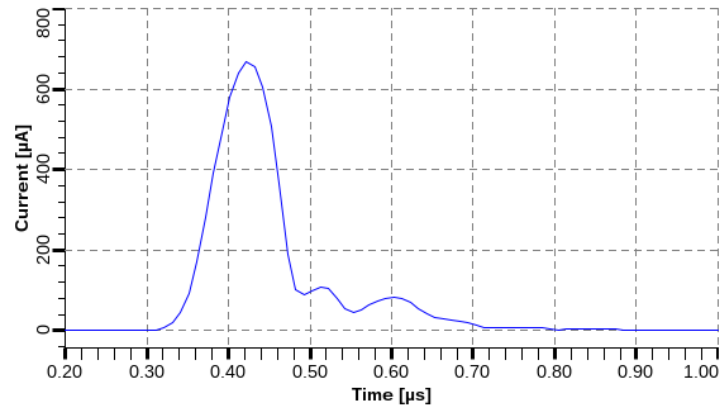
Structure of the "[pv_root]:tc_mode" PV

value (<i>boolean</i>)	
timeStamp (<i>structure</i>)	secondsPastEpoch (<i>long int</i>) nanoseconds (<i>int</i>) userTag (<i>int</i>)

Structure of the "[pv_root]:charge" and "[pv_root]:current" PVs

value (<i>double = float64</i>)	
alarm (<i>structure</i>)	severity (<i>int</i>) status (<i>int</i>) message (<i>string</i>)
timeStamp (<i>structure</i>)	secondsPastEpoch (<i>long int</i>) nanoseconds (<i>int</i>) userTag (<i>int</i>)
display (<i>structure</i>)	units (<i>string</i>)
trigg (<i>unsigned long int</i>)	

TICT_BCM_DAQ_TC_WF – Acquire a current waveform



pva://TICT-BI01:TC_WF/current



Trigger number

1156688

Example of the current waveform of a single pulse in a control system
(For laser-plasma charge measurement)

This interface provides a single PV containing an array of current values, an array of times, and a trigger number. To set up such an interface, add a "TICT-BCM-DAQ_TC_WF" section to the "setup/BI.xml" file as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<bergoz_systems>
  <TICT-BCM-DAQ_TC_WF
    pv_root="TICT-BI01"
    BCM-RF-E_idn="IDN=Bergoz,BCM-RF-E,SN BI01,FW 3.4"
    DAQ-BCM-RF_idn="DAQ-BCM-RF:BI01 - 228.1 - rev 0"
    LUT-TC="/home/user/BIPVA/setup/LUT-BCM-RF-E.TC.BI01.txt"
    LUT-DAQ-BCM-OUT="/home/user/BIPVA/setup/LUT-DAQ-BCM-RF.out.BI01.txt"
  />
</bergoz_systems>
```

Field	Description
pv_root	Root of the process variable name. In this example, the PV will be: - TICT-BI01:TC_WF
BCM-RF-E_idn DAQ-BCM-RF_idn	Identification numbers of the serial interfaces of the BCM-RF-E and the DAQ-BCM-RF.
LUT-TC LUT-DAQ-BCM-OUT	Look up tables of the BCM-RF-E in Track Continuous mode, and the BCM-OUT input of the DAQ-BCM-RF.

If you have properly configured the "setup/BI.xml" file, the following PV will be accessible from your EPICS network after running BIPVA.py:

Structure of the "[pv_root]:TC_WF" PV

current (double array = array of float64)	
time (double array = array of float64)	
alarm (structure)	severity (int) status (int) message (string)
timeStamp (structure)	secondsPastEpoch (long int) nanoseconds (int) userTag (int)
display (structure)	units (string)
trigg (unsigned long int)	

More information and latest manuals revisions can be found on our website

www.bergoz.com

If you have any questions, feel free to contact us by e-mail

info@bergoz.com

