Technical Note

Using
CERN Power Converter Controls
with EPICS and TANGO

AUTHOR:
Quentin King, CERN TE-EPC

ABSTRACT:
CERN has developed a Power Converter controller known as a Function Generator/Controller (FGC) which can be deployed at sites that use the EPICS or TANGO frameworks. This paper includes an overview of the latest version of the FGC hardware and firmware. It also presents the hardware and software components needed to integrate FGC-based power converter controls into EPICS and TANGO.
## HISTORY OF CHANGES

<table>
<thead>
<tr>
<th>REV. NO.</th>
<th>DATE</th>
<th>PAGES</th>
<th>DESCRIPTIONS OF THE CHANGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2018-07-11</td>
<td>ALL</td>
<td>First version</td>
</tr>
<tr>
<td>1.1</td>
<td>2018-07-22</td>
<td>Various</td>
<td>Minor improvements for readability</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>Update figure 16 to show PowerSpy and Terminal clients</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

1. Executive Summary ........................................................................................................... 4
2. FGC History ....................................................................................................................... 5
3. Power Converter Integration ............................................................................................. 5
4. FGC3.1 Hardware .............................................................................................................. 6
   4.1 Network interface ......................................................................................................... 7
   4.2 Analog interface ........................................................................................................... 7
   4.3 USB interface .............................................................................................................. 8
   4.4 SCIVS and SPIVS buses .............................................................................................. 9
   4.5 Diagnostic Interface .................................................................................................... 9
   4.6 Maxim (Dallas) 1-wire Identification buses ................................................................. 10
   4.7 Synchronization .......................................................................................................... 12
5. FGC3.1 Firmware ........................................................................................................... 14
   5.1 Boot Program .............................................................................................................. 14
   5.2 Main Program ............................................................................................................. 14
6. FGC Ethernet Gateway ................................................................................................. 20
   6.1 Device directory – the FGC name file ....................................................................... 21
   6.2 TCP communication ................................................................................................. 21
   6.3 UDP publication ........................................................................................................ 21
   6.4 FGC remote terminal ............................................................................................... 21
   6.5 FGC firmware updates .............................................................................................. 22
7. FGC Applications ........................................................................................................... 23
   7.1 FGC web server ......................................................................................................... 23
   7.2 Post mortem manager ................................................................................................. 25
   7.3 Configuration manager (planned in 2019) ................................................................ 25
8. Integration in EPICS ......................................................................................................... 26
9. Integration in TANGO ..................................................................................................... 27
10. Portable FGC3 crate ...................................................................................................... 28
11. References .................................................................................................................... 30
1. Executive Summary

The FGC3.1 is an embedded power converter controller developed by the CERN Electrical Power Converters group (1). Since 2012, more than 2000 have been manufactured and hundreds have been deployed into operation at CERN.

![FGC3.1 module](image)

Figure 1 - FGC3.1 module

In 2016, a project was started to develop the software required to use FGC3 controllers outside CERN. In 2018, the first FGC3 was deployed into operation (2) at the TRIUMF laboratory, controlling the new 20 kA power converter for their main cyclotron magnet. TRIUMF uses the EPICS framework, so they developed a simple binding between EPICS and the FGC. By the end of 2018, a more advanced binding will be available that will support a faster refresh rate for the power converter status data.

More FGC3s will be deployed in 2018 at ESRF (3) in Grenoble and FREIA (4) in Uppsala. FREIA uses EPICS while ESRF uses TANGO, so a similar binding will be developed for FGC3s within the TANGO framework.

The FGC3.1 controller can manage fast-pulse capacitor discharge converters, switch-mode converters and thyristor converters. It operates at 10 kHz and with a fast switched-mode converter, it can regulate the circuit current or magnetic field with a bandwidth of up to 1 kHz. It can support redundant current or field measurements.

As well as regulating the current or field, the FGC3 can manage the power converter state, optional polarity switch and first-fault diagnostics. It has a sophisticated function generator included in the firmware and advanced digital and analogue signal logging in circular buffers.

Communication with the FGC3s uses a dedicated switched Ethernet fieldbus via a Linux-based gateway system. FGC3s cannot communicate directly with client applications using TCP/UDP because they use a simpler communication protocol over raw Ethernet. Each gateway can manage up to sixty four FGC3s using two network interfaces, one for the FGC fieldbus and the other for the accelerator controls LAN. The gateways are also responsible for the synchronisation of the clocks inside the FGC3s and FGC3 firmware updates.
FGC3s can exchange information via the Ethernet fieldbus when required. For example:

- Multiple FGC3s can work together in a master-slave(s) configuration if a circuit is powered by multiple converters.
- If circuits are coupled electrically or magnetically, then the FGC3s can share information to allow decoupling by the FGC3 firmware.
- If an analogue signal needs to be acquired far from the power converter, then an FGC3 can be used as a remote measurement device. It can send the measurement over the FGC Ethernet fieldbus, avoiding long analogue signal cables.

Advanced expert tools are provided to allow power converter experts to manage the power converters. These include a high performance web-based graphing tool called **PowerSpy** that can display and analyse logged signals. In the event of a trip, a post mortem server will automatically read out and save the important logged signals in files. These post mortem files can be analysed later using PowerSpy.

The automatic configuration of the circuit related parameters is assured by a configuration manager program\(^1\). This means that an FGC3 can be replaced by a spare and the new FGC3 will be set up to work identically without human intervention.

## 2. FGC History

The FGC project started in 1998 with the development of the FGC1. This first version was used for about ten years in the CERN Large Hadron Collider (LHC) magnet test benches. In 2002, version 2.3 was finalised for use with the 1750 power converters in the LHC. These began operation in 2008. At the same time, the FGC3 project was started to develop a smaller, faster and cheaper controller for the other accelerators at CERN. The first FGC3.1s were deployed into operation in 2012. They are used in CERN’s new Linac4 and HIE-ISOLDE accelerators and the ELENA decelerator as well as for the upgrade of obsolete power converter controls in CERN’s older accelerators and transfer lines.

## 3. Power Converter Integration

The FGC2 was designed to work with analogue voltage sources by driving a voltage reference signal with a DAC. The FGC3 supports the same analog interface as well as digital serial links that can drive a digital voltage source. CERN has developed a set of crates and cards to control different types of power hardware including MOSFET, IGBT and thyristor bridges and capacitor discharge topologies. These crates and cards are known as RegFGC3 components. RegFGC3 controls can be bought or licensed from CERN, or the FGC3 can be used alone with an external current source, voltage source or firing source.

This document focuses on the FGC3.1. For more information on the broader RegFGC3 platform, please contact [Nick Zilogas](mailto:nick.zilogas@cern.ch).

---

\(^1\) Due in 2019
4. FGC3.1 Hardware

This chapter provides an overview of the FGC3.1 hardware. A much more complete description is available in this document (5).

The FGC3.1 module is assembled into a protective metal cassette (see Figure 1) with dimensions 3U x 10TE x 220mm. Figure 2 shows an overview of the internal architecture of the FGC3.1.

Table 1 lists the main features of the hardware. Physically, the components are spread across a main board, shown in Figure 3, and two daughterboards, visible in Figure 4.
Table 1 - Overview of the FGC3.1 hardware

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller (MCU)</td>
<td>Renasas RX610</td>
<td>100MHz, 32-bit FP only</td>
</tr>
<tr>
<td>MCU internal SRAM</td>
<td>128 KB</td>
<td></td>
</tr>
<tr>
<td>MCU external SRAM</td>
<td>1 MB</td>
<td></td>
</tr>
<tr>
<td>MCU internal FLASH</td>
<td>2 MB</td>
<td></td>
</tr>
<tr>
<td>MCU Non-volatile MRAM</td>
<td>512 KB</td>
<td></td>
</tr>
<tr>
<td>FPGA</td>
<td>Xilinx Spartan 3-700AN</td>
<td></td>
</tr>
<tr>
<td>Digital Signal Processor (DSP)</td>
<td>TI TMS320C6727</td>
<td>300MHz, 32 &amp; 64-bit FP</td>
</tr>
<tr>
<td>DSP internal SRAM</td>
<td>256 KB</td>
<td></td>
</tr>
<tr>
<td>DSP external DRAM</td>
<td>64 MB</td>
<td></td>
</tr>
<tr>
<td>Digital command lines</td>
<td>8 open collector</td>
<td>For power converter control</td>
</tr>
<tr>
<td>Digital status input lines</td>
<td>16</td>
<td>For power converter and DCCT status</td>
</tr>
<tr>
<td>Interlock input signals</td>
<td>2</td>
<td>Opto-coupler isolation</td>
</tr>
<tr>
<td>Interlock output signals</td>
<td>2</td>
<td>Relay contacts</td>
</tr>
<tr>
<td>Diagnostic buses</td>
<td>2 x 500 kbps FGC Diagnostic Interface Module (DIM) buses</td>
<td>Up to 16 DIMs per bus, 24 digital and 4 analog signals per DIM.</td>
</tr>
<tr>
<td>Dallas 1-wire ID buses</td>
<td>6</td>
<td>For remote reading of barcodes</td>
</tr>
<tr>
<td>RegFGC3 SPI link</td>
<td>10 Mbps</td>
<td>Real-time link to converter control card</td>
</tr>
<tr>
<td>RegFGC3 SCI link</td>
<td>5 Mbps</td>
<td>Non-real-time link to all RegFGC3 cards</td>
</tr>
<tr>
<td>Power requirements</td>
<td>+5V 1.25A</td>
<td>Future FGC3s versions(^2) may consume more than the FGC3.1</td>
</tr>
<tr>
<td></td>
<td>+15V 0.35A, -15V 0.12A</td>
<td></td>
</tr>
</tbody>
</table>

4.1 Network interface

The network interface daughterboard has the RJ45 connector and 100 Mbps LAN controller. It also has the fieldbus address connector. This connector allows a “dongle” encoding the fieldbus address (1-64) to be plugged onto the front panel. If an FGC3 needs to be replaced by a spare unit, the dongle just needs to be plugged onto the new unit for it to be correctly configured to take over the control of the converter.

4.2 Analog interface

The analog interface daughterboard (ANA-103) has four ADCs and two DACs. The analog inputs use \textbf{LTC2378-20} (6) SAR ADCs from Linear Technology. They are sampled at 500 kHz and are filtered in the FPGA with a simple 50-sample accumulator to produce 10 ksps. The 500 kHz samples are only logged when the FGC is managing a fast-pulse capacitor discharge converter.

The DACs are 16-bit \textbf{MAX5541CSAs} (7), oversampled to deliver 20-bits resolution with a bandwidth of about 3 kHz. The first DAC can be used to send the current, voltage or firing reference to an analogue power converter. The second DAC is only used for special applications.

\(^2\) Future FGC3s will be plug compatible, but may consume more power than the FGC3.1
Table 2 - ANA-103 analog interface performance at constant temperature

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Typical</th>
<th>Worst case</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial uncertainty (2 x rms)</td>
<td>5 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Non-linearity</td>
<td>±4 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Stability during 12 hours</td>
<td>5 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Stability during 20 minutes (2 x rms)</td>
<td>1 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Noise &lt; 500Hz (2 x rms)</td>
<td>6 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Repeatability (2 x rms)</td>
<td>2 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Long term fill to fill stability max - min</td>
<td>20 ppm</td>
<td></td>
<td>ppm</td>
</tr>
<tr>
<td>Temperature coefficient ppm/K</td>
<td>±1 ppm</td>
<td>±2.5 ppm</td>
<td>ppm/K</td>
</tr>
<tr>
<td>Warm up time to reach specification</td>
<td>24 hours</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The ANA-103 circuit board includes a temperature regulated zone for the voltage references, including ±10V references used for automatic calibration of the ADCs. Once calibrated, the ADCs can calibrate the DACs. Note that the ANA-103 has no adjustable components. Calibration of the FGC3 is based on the measurement of the calibration errors and the compensation of the errors by the FGC3 firmware. For a more complete description of the ANA-103 interface, please consult this technical note (8).

The performance of the ADCs is given in Table 2.

4.3 USB interface

The FGC3.1 includes an FTDI dual-channel USB1 interface controller connected to a Type-A USB port. This allows basic control of a power converter without needing an FGC Ethernet gateway.

FTDI Drivers are included in Linux, Windows and MacOS, so if the FGC3.1 is connected to a PC, two COM ports will appear. One COM port is for the terminal and the other is for the “Spy Data” channel. The terminal can be used by a human operator (in editor mode) or a program (in direct mode) and it supports a simple ASCII command/response protocol. For legacy reasons, the speed of the terminal is limited to only 960 characters per second. This is very slow, but is adequate for basic control actions.

Figure 5 - Example of the FGC USB terminal in editor mode in a Windows terminal emulator
Figure 5 shows an example of the terminal in editor mode connected to a terminal emulator program running under Windows.

The Spy Data channel allows the FGC3.1 firmware to stream six floating-point signals at 1 ksps. The choice of the six signals can be made from a long list of possible signals. The data received through the Spy Data COM port can be written to a file and then visualised or processed later using PowerSpy or other applications.

4.4 SCIVS and SPIVS buses

The FGC3 has two digital serial buses to communicate with the CERN-designed RegFGC3 digital power converter control cards.

The *Serial Communications Interface for Voltage Sources* (SCIVS) is a non-real-time 5 Mbps bus that also distributes a 5 MHz clock to the RegFGC3 cards, so that all cards will work synchronously with the FGC3’s clock.

The *Serial Peripheral Interface for Voltage Sources* (SPIVS) is a real-time point-to-point link at 7 Mbps that can exchange data between the FGC3 and the voltage regulation and firing control board at up to 10 kHz.

More details are available in (5).

4.5 Diagnostic Interface

A very important requirement for the power converter controller is the identification and recording of the first fault that causes a power converter to trip. This is especially important for large converters created from multiple sub-converters. The largest at CERN supplies the LHC ATLAS experimental magnet with up to 20.4kA. It has 8 sub-converters, each with two input modules and three output modules. If a sub-converter trips, normally the rest can continue to operate, taking over the missing current within a millisecond. However, in some cases, the rest of the sub-converters may also trip in a cascade, resulting in the fast stop of the whole converter. In this situation, it is essential that the FGC can report the first fault on the first sub-converter so that the faulty module can be replaced.

This is possible using a small acquisition card called the FGC Diagnostic Interface Module (DIM). A manufacturer can design their boards to allow one or more DIMs to be integrated into the power converter electronics. Each DIM has 24 digital inputs and one trigger input. Internally there is a clock with a period of 8 μs. When the trigger input is activated, the state of the 24 inputs is latched within 1 μs and the clock is frozen. Provided the trigger is the sum of all the fault signals connected to the DIM, this will allow the first fault to be identified, even if multiple DIMs are triggered.

As an extra service, the DIM has four slow (50 sps) 12-bit ADC channels, which are logged by the FGC3 firmware and which can be viewed using PowerSpy. A DIM is shown in Figure 6 and the design is described in detail in a technical note (9). More details about how the DIM can be integrated into the power converter electronics is available in this engineering specification (10).

The FGC3 has two queued serial peripheral interface (QPSI) buses, A and B. Each bus can have up to 16 DIMs connected in a daisy chain (or 32 can be connected just to bus
A). This allows a total of 768 diagnostic digital signals and 128 diagnostic analog signals to be acquired.

Figure 6 - Diagnostic Interface Module (DIM)

4.6 Maxim (Dallas) 1-wire Identification buses

Identification of the components in the power converter, controls crate and FGC3 itself is very important for three reasons:

1. Automatic configuration of calibration error values for analog components (DCCTs, ADCs and voltage references)
2. Validation that all expected components are present
3. Asset tracking

The technology chosen is the Maxim 1-wire system (11). This is often referred to as the Dallas 1-wire system, as it was originally developed by Dallas Semiconductor Corp.

The FGC3 has six buses, each able to support dozens of 1-wire devices. Two types of 1-wire devices are used, one just has a unique 64-bit ID number, while the other also has a temperature sensor. The buses are:

1. Voltage source bus A – follows DIM bus A and visits DIM cards
2. Voltage source bus B – follows DIM bus B and visits DIM cards
3. Measurement bus A – DCCT A head and electronics and external ADC A (if used)
4. Measurement bus B – DCCT B head and electronics and external ADC B (if used)
5. Internal FGC bus
6. FGC crate bus

The voltage source buses share the same cable as the DIM buses and pass through the DIM cards, which have a 1-wire ID device to identify themselves.

The 1-wire bus standard cannot identify the order of components on a bus. This is why there are two measurement buses. It is essential for the configuration of the FGC that
the identities of DCCT A and DCCT B be known. This would not be possible if they had
to share one bus.

CERN identifies all components with barcodes with a standardised format. For
components that need to be tracked automatically, they also have a 1-wire device and
the association between the 1-wire ID number and the barcode is stored in a database.

Figure 7 - Screenshot of the FGC Barcode Manager web interface

Figure 7 shows a screenshot of the web interface for the FGC Barcode Manager. This is
an application that allows the ID-Barcode associations to be managed. More than 70,000
components are currently tracked in this way, with roughly half being for FGC2-
controlled converters and half for FGC3-controlled converters.

Each FGC3.1 can store up to 40,000 ID-Barcode associations, so the FGC3 firmware can
report the barcodes of all the connected components. For the largest power converter
at CERN, more than 80 components are identified in this way.

Figure 8 – FGC Ethernet fieldbus architecture at CERN
4.7 Synchronization

The FGC3.1 includes a 25 MHz voltage-controlled oscillator (VCXO) as its primary clock. The frequency is controlled by a Phase-locked Loop (PLL) which can be synchronized in two ways. At CERN, a 50 Hz synchronisation signal generated in the accelerator timing interface in the gateway is distributed to the FGC3s through an unused pair inside the Ethernet cable. Using the Ethernet cable in this way reduced the cost of cabling and connectors. This is illustrated in Figure 8.

A “Pulse Injector” was developed to fan-out the synchronisation signal. The pulse injector is paired with a low-cost unmanaged switch to form an FGC Ethernet fieldbus “star-point”. Up to three star-points can be daisy-chained to one gateway, to allow up to 64 FGC3s to be connected. Figure 9 shows a star-point with the 1U-tall switch above the 1U-tall pulse injector. Below there is 1U for the cables to pass to the back of the rack for a total of 3U per star-point.

So at CERN, the 50 Hz synchronisation pulses are the primary source of synchronization, however, if they fail or are not available, then the FGC3’s PLL switches automatically to synchronise using the time of arrival of “time packets” sent by the gateway over the Ethernet every 20 ms.

Figure 9 - FGC_Ether fieldbus star point at CERN

Figure 10 - FGC_Ether architecture outside CERN
Synchronisation using the gateway’s time packets has an increased RMS phase error, because it depends upon the response time of software running in the gateway. Using Linux with the RT_PREMPT patch makes the phase error less vulnerable to disturbances due to other processes running on the gateway machine. It can work quite well with normal (non-real-time) Linux provided the gateway machine has a multi-core CPU and all other activities are minimised. The typical phase error with the 50 Hz synchronisation pulses is less than 40 ns, while the typical phase error for the time packets is 10-20 μs. With non-real-time Linux, the worst case is essentially unbounded, but the FGC3’s PLL is tolerant to occasional late or even lost packets.

So for most applications, an Ethernet-only solution is sufficient and installations outside CERN will normally follow the simpler architecture shown in Figure 10.
5. FGC3.1 Firmware

The FGC3 has two different programs in its Flash memory:

1. Boot program – this always runs first after a reset. It is a bit like the bios on a PC. It provides important low-level services including self-tests, firmware updates and the running of the main program when everything is nominal.

2. Main program – this controls a power converter.

5.1 Boot Program

The boot program is rather basic and supports interaction with a user through a system of numbered nested menus. For example, the root menu has these options:

```
Root menu  
B:A G:1 T:1497109742 F:0200 W:0000 L:0000 ST:FO  
0* RunMain  
1  Run Log  
2  Device status  
3  Manual Tests  
4  Self tests  
5  Codes  
6  Expert  
7  Spy  
8* Build info
```

There is only one class of boot program for the FGC3.1 platform.

5.2 Main Program

There are different classes of FGC3.1 main program dedicated to different topologies of power converter. FGC3.1 main program classes are numbered from 61 to 69:

- Class 62 is used to control fast-pulse capacitor discharge power converters
- Class 63 is used to control switched-mode and thyristor power converters
- Class 64 is used to control static var compensators

Note that Class 61 existed but was replaced by Class 63. Classes 65 to 69 are reserved for future use.

5.2.1 CERN Device-Property Model

For many years, CERN has used the Device-Property model for accelerator controls. In this model, the following terms are defined:

- Device – an instance of a type of controller running a software class to implement a particular functionality. The device has a name and a network address.

- Property – a named data container. It has a type (integer, float, string, enum, bit mask, etc...). It can be scalar or an array and it can be read-only, read-write or even (occasionally) write-only.

Devices only support three commands: set, get and subscribe. All the complexity of the class is encapsulated in the list of properties, which can be numerous. For Class 63,
there are around 1000 properties, though relatively few are important for the operation of the power converter.

FGC property names are hierarchical with point separators: A.B.C, A.B.D, etc... For example: MODE.PC, STATE.PC, STATUS.FAULTS, REF.RAMP.ACCELERATION.

5.2.2 Power Converter State

A primary activity for all main program classes is the management of the state of the power converter. The user or control application can set the desired state using the MODE.PC property and read back the actual state in the STATE.PC property.

The power converter state machine is shown in Figure 11. It combines the states related to starting and stopping the converter with the states related to the reference generator.

![Figure 11 - Power Converter State Machine](image)

The reference generator can operate in three different ways:

1. Unsynchronised ramping to a desired set point (DIRECT)
2. Synchronised execution of a single-use function (IDLE, ARMED, RUNNING)
3. Synchronised execution of a cycling function (CYCLING)

The third option is used extensively in CERN's cycling accelerators, but it requires a timing system. Since different labs use different timing systems, this is not currently supported in the FGC gateway used outside CERN. Options 1 and 2 don’t depend on a timing system, but the quality of the synchronisation for option 2 will depend upon the
accuracy of the clock in the gateway computer. If it uses the ntpd service, this can be good to about a millisecond. To be more accurate will probably need a timing interface.

More information on the reference manager can be found in this ICALEPCS’15 paper (12).

Class 63: Published Data

<table>
<thead>
<tr>
<th>CLASS</th>
<th>[TOP]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address:</td>
<td>0x00000:0</td>
</tr>
<tr>
<td>Size:</td>
<td>40B</td>
</tr>
</tbody>
</table>

Property:

<table>
<thead>
<tr>
<th>Child zone</th>
<th>Offset</th>
<th>Size</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST_FAULTS</td>
<td>0x00000:0</td>
<td>2B</td>
<td>Faults</td>
</tr>
<tr>
<td>ST_WARNINGS</td>
<td>0x00002:0</td>
<td>2B</td>
<td>Warnings</td>
</tr>
<tr>
<td>ST_LATCHED</td>
<td>0x00004:0</td>
<td>2B</td>
<td>Latched status</td>
</tr>
<tr>
<td>ST_UNLATCHED</td>
<td>0x00006:0</td>
<td>2B</td>
<td>Unlatched status</td>
</tr>
<tr>
<td>STATE_PLL</td>
<td>0x00008:0</td>
<td>1B</td>
<td>Phase-locked loop state</td>
</tr>
<tr>
<td>STATE_OP</td>
<td>0x00009:0</td>
<td>1B</td>
<td>Operational state</td>
</tr>
<tr>
<td>STATE_VS</td>
<td>0x0000A:0</td>
<td>1B</td>
<td>Voltage source state</td>
</tr>
<tr>
<td>STATE_PC</td>
<td>0x0000B:0</td>
<td>1B</td>
<td>Power converter state</td>
</tr>
<tr>
<td>ST_ADC_A</td>
<td>0x0000C:0</td>
<td>1B</td>
<td>ADC A status</td>
</tr>
<tr>
<td>ST_ADC_B</td>
<td>0x0000D:0</td>
<td>1B</td>
<td>ADC B status</td>
</tr>
<tr>
<td>ST_ADC_C</td>
<td>0x0000E:0</td>
<td>1B</td>
<td>ADC C status</td>
</tr>
<tr>
<td>ST_ADC_D</td>
<td>0x0000F:0</td>
<td>1B</td>
<td>ADC D status</td>
</tr>
<tr>
<td>ST_DCCT_A</td>
<td>0x00010:0</td>
<td>1B</td>
<td>DCCT A status</td>
</tr>
<tr>
<td>ST_DCCT_B</td>
<td>0x00011:0</td>
<td>1B</td>
<td>DCCT B status</td>
</tr>
<tr>
<td>L_ERR_MA</td>
<td>0x00012:0</td>
<td>2B</td>
<td>Error between current reference and measurement (mA)</td>
</tr>
<tr>
<td>L_DIFF_MA</td>
<td>0x00014:0</td>
<td>2B</td>
<td>Difference in current measurements (mA)</td>
</tr>
<tr>
<td>EVENT_GROUP</td>
<td>0x00016:0</td>
<td>1B</td>
<td>Reference event group</td>
</tr>
<tr>
<td>L_EARTH_PCNT</td>
<td>0x00017:0</td>
<td>1B</td>
<td>Earth fault current (percent of trip level)</td>
</tr>
<tr>
<td>L_REF</td>
<td>0x00018:0</td>
<td>4B</td>
<td>Current reference (A)</td>
</tr>
<tr>
<td>L_MEAS</td>
<td>0x0001C:0</td>
<td>4B</td>
<td>Current measurement (A)</td>
</tr>
<tr>
<td>V_REF</td>
<td>0x00020:0</td>
<td>4B</td>
<td>Voltage reference (V)</td>
</tr>
<tr>
<td>V_MEAS</td>
<td>0x00024:0</td>
<td>4B</td>
<td>Voltage measurement (V)</td>
</tr>
</tbody>
</table>

Def: Wed Jul 18 05:51:47 2018

Figure 12 - Web page documenting the published status data for Class 63

5.2.3 Regulation

The option to regulate the current or field and voltage is included in Class 63. For fast-pulse capacitor discharge converters, the regulation of the flat top of each pulse is implemented in a RegFGC3 board using an FPGA so Class 62 is simply responsible for setting the reference, controlling the timing and acquiring the measurements.
For Class 63, the current and field regulators are implemented using the RST algorithm, while the voltage regulator (used with thyristor converters) is implemented using classical state-space equations.

More information on the RST implementation can be found in this [ICALEPCS’11 paper](#) (13)

5.2.4 Circuit Load Model in Class 63

The circuit load mode in Class 63 is shown in Figure 13. It is a first order model, but it includes a parallel damping resistance.

![Figure 13 – Circuit load model in Class 63](image)

With warm magnets (i.e. not super-conducting), saturation of the iron yoke can be a very significant effect which can reduce the time constant of the load by more than a factor of two. If this is not compensated, it can either make the current regulator unstable or require the regulator performance to be substantially degraded. Class 63 allows the user to define a simple model of the saturation effect on the differential inductance and this is used to hide the change in the circuit time constant from the current regulator.

5.2.5 Status Publication

The communication on the FGC Ethernet fieldbus has a cycle time of 20 ms. The gateway broadcasts a “time packet” at the start of milliseconds 18 and 19. In return, every FGC sends a small “status packet” to the gateway in the middle of millisecond 1. The packet contains 56 bytes of which 40 bytes are available for the developer of the class to define which properties should be published. This is specified in an XML file which is used to generate a C header file declaring the status structure for the class. A web page documenting the status data is also generated (see Figure 12) as well as decoders for Perl, Python, EPICS³ and TANGO⁴.

The gateway can read a UDP subscriptions file on start-up, defining the hostnames, UDP ports and transmission periods for publication of UDP packets containing the status data from every FGC managed by the gateway. The gateway is also an FGC device and can publish its own 56 bytes of status data. The packets also contain a time stamp.

So a client system can receive the status of every FGC and gateway at up to 50 Hz by simply listening to a UDP port.

5.2.6 Logging

The FGC3.1 dedicates nearly 60 MB of memory to the logging of analog and digital time-series signals. Logging is based on multiple circular buffers.

---

³ In development
⁴ In development
All the signals in a buffer share the same time base. For most logs, the sampling rate is fixed, for example, in Class 63 the I_MEAS log is always recorded at 10 kbps. For some logs, the sampling rate may depend upon the configuration of the FGC3. In Class 63 the I_REG log records signals related to the regulation of the current, so it samples the signals at the current regulation rate. This rate cannot be changed while the power converter is running, but it can be changed when it is stopped.

The developers of an FGC software class define all the logs and signals in a spreadsheet. This is processed automatically to configure the logging for the Class when the software is compiled. The developers specify the list of all possible signals for each log and the number of samples in the circular buffer. This obviously defines both the time that each sample will remain in the log before being overwritten, as well as the size in memory that the log take up.

Figure 14 shows a screenshot of the logging definition spreadsheet for Class 63. It is too small to read, but each column defines one log buffer and each of the lower rows with the white background defines a signal. This gives an idea of the number of logs and signals involved.

Even with 60MB, there isn’t enough memory for the FGC to log every possible signal for a reasonable duration, so it is possible to define more signals for a log than can fit in the buffer. For example, with Class 63, the ACQ_FLTR log has space for four signals, but 14 possible signals can be logged. The choice of two of the four signals is fixed while the other two can be selected using a property. This can be done easily with the PowerSpy tool.

All the FGC3.1 main program classes also record an event log. This is a text-based circular log where records are individually timestamped. It reports set commands, changes of state machines and statuses and the first fault in the event of a trip. Figure 15 shows a screenshot of part of an event log for a power converter in the Large Hadron Collider, viewed using PowerSpy.
Figure 15 - Screenshot of part of an Event Log showing that the first fault that stopped the converter in this case, was due to an external FAST ABORT interlock.
6. FGC Ethernet Gateway

At CERN, the FGC Ethernet Gateway program was developed in 2011 and has been operational since 2012. It integrates the FGC controls into the CERN accelerator control system, with links to the CERN timing, alarms, logging, post mortem service and controls middleware. It runs on the real-time version of Linux that CERN uses for all front-end computers in the accelerator controls context.

In 2016, a project was started to write a new, simpler version of the FGC Ethernet gateway that could be used anywhere and could run on any standard Linux distribution. In March 2018, this version started operating at the TRIUMF lab. While it is recommended to use real-time Linux, it will work fine on normal non-real-time Linux, provided the gateway machine has sufficient resources and other activities are minimised. A multi-core CPU is also helpful.

The architecture of the new gateway is presented in Figure 16.

Figure 16 - Architecture of the FGC Ethernet gateway
6.1 Device directory – the FGC name file

The relationship between FGC device names and fieldbus addresses on gateways is maintained in the FGC "name file". This has a very simple syntax with colon separated fields. Here is a sample of lines from the name file at CERN:

- `cfc-115-r1:0:4:CFC-112-R1:0x0000`
- `cfc-115-r1:1:62:RPZEG.112.MOD.AC1:0x0000`
- `cfc-115-r1:2:62:RPZEF.112.MOD.AC1:0x0000`
- `cfc-120-reth1:0:4:CFC-150-RETH1:0x0100`
- `cfc-120-reth1:1:62:RPAFX.150.ITE.RQFN01:0x0100`
- `cfc-120-reth1:2:62:RPAFX.150.ITE.RQDN02:0x0100`
- `cfc-120-reth1:3:62:RPAFX.150.ITE.RQDN04:0x0100`
- `cfc-120-reth1:4:62:RPAFX.150.ITE.RQFN05:0x0100`

The fields are:

- `<gateway hostname>`<fieldbus address>`<device class>`<device name>`<operational mask>`

The name file is read by all gateways, PowerSpy and all other tools that want to interact with FGCs or gateways.

Device names have point separated fields, starting with the device type and are limited to 24 characters. For power converters at CERN, they typically have the format:

- `<Power Converter Type>.<Power Converter Location>.<Circuit Name>`

The Circuit Name is often broken into two fields, separated by a point.

6.2 TCP communication

The FGC gateway listens to a TCP port and will accept connections. Once a client has connected, it can communicate with either the gateway’s own device or FGCs connected to the gateway using the FGC Network Command/Response protocol. This is described in this document (14).

6.3 UDP publication

As mentioned in section 5.2.4, the gateway can send UDP packets containing the status data from the gateway and all the FGCs to UDP ports defined in a file that the gateway reads on start up. The maximum rate of transmission is 50 Hz and for each destination, the period can be defined in the file to be an integer multiple of 20 ms.

6.4 FGC remote terminal

The gateway supports the connection to the FGCs terminal interface, remotely via the FGC Ethernet fieldbus. Multiple connections can be made to the same FGC and the local USB terminal can also be connected at the same time. In this case, all the keyboard characters from all the terminals will be received by the FGC and the characters sent to the terminal by the FGC firmware will appear on all the terminals at the same time.

A command line FGC terminal client is provided in the FGC software package and PowerSpy also provides terminal access to the FGCs from the browser.
6.5 FGC firmware updates

The gateway can read FGC firmware files and will broadcast the contents in a round-robin way in the time packets. It will send 1 KB of code per 20 ms, or 50 KB/s. It also broadcasts the list of all the available firmware packages (known as FGC codes). For each FGC code, the gateway “advertises” the type of the code, the version and the checksum. This allows the FGC boot program to rapidly check if any codes resident in its flash memory need updating. If so, the boot program waits for the round-robin to reach the code or codes that it needs to update. These are then written to flash. In this way, every FGC will ensure that it is fully up to date after every reset. The gateway can be told to re-read the code files if they have been updated by setting a property.
7. FGC Applications

In addition to the gateway systems, an application server is needed to manage a population of FGC3s. This can be a virtual machine but it must run Linux. Figure 17 shows an overview of the hardware and software components needed to integrate FGC3 controls with EPICS or TANGO. Just one gateway is shown in the figure, but there can be as many as needed, given that no more than 64 FGC3s can be connected to a gateway.

7.1 FGC web server

7.1.1 PowerSpy

The PowerSpy web application requires Apache on the application server. The backend of the application is written in Python. The client application is written in JavaScript and uses the open-source Flot charting library (15). This was selected because it is extremely fast and can handle very large datasets. Figure 18 shows a screenshot of the standalone version of PowerSpy displaying a buffer of signals acquired by the controller of the power converter for the CERN Proton Synchrotron main magnets.

PowerSpy can also pop open terminal windows that behave identically to the local terminal that can be used via the FGC’s USB port.

For more information about PowerSpy, please consult this introduction (16). It is possible to try out the client interface using a standalone version of PowerSpy (17) which can read and display data from local files. Video tutorials are available under the Help menu.
Figure 18 - The PowerSpy data viewer showing data from the CERN PS accelerator

7.1.2 FGC Project Web Site

The Apache web server can also serve the FGC project web site. This includes thousands of pages of documentation generated from the FGC XML definition files, including a description of all the properties instantiated by the FGC3 firmware. This is a useful resource for application developers whose software has to interact with the FGC3s. Figure 19 shows a screen shot of the site map of the FGC project web site.

Figure 19 - Screenshot of the site map of the FGC project web site
7.2 Post mortem manager

The FGC post mortem manager program, written in python, can receive and decode the UDP published status packets. It can see when a converter trips completely or starts a “slow abort” to smoothly extract the energy in the circuit before turning off. It is responsible for reading out the logs and storing the data in PowerSpy files in a directory structure that can then be navigated and searched within PowerSpy.

Files are stored with this hierarchy:

```
.../Trips/{Year}/{Month}/{Day}/{Trip_Time}/{Device_Name}/{Buffer_Name}
```

or:

```
.../Slow_Aborts/{Year}/{Month}/{Day}/{Trip_Time}/{Device_Name}/{Buffer_Name}
```

Multiple instances of the post mortem manager can be running on the application server or servers, if the number of gateways is large. The configuration is driven by the UDP subscription files that the gateways read.

7.3 Configuration manager (planned in 2019)

The FGC configuration manager program will be written in python and will also receive and decode the UDP published status packets. It will see when an FGC has restarted and will set its configuration properties from a database.

There are three different types of configuration properties:

- **System properties** – these are associated with the device name. System properties include current limits, circuit load characteristics and regulation parameters.

- **Equipment properties** – these are associated with the barcode for an individual component such as a DCCT or analog interface. Equipment properties usually contain calibration data.

- **Type properties** – these are associated with the type of a component. This is known from the first part of the barcode. Typically, type properties are scale factors for a piece of hardware, for example, the turns ratio of a DCCT head.

If system properties or equipment properties are changed in the FGC, then the user can request the configuration manager to update the database from the FGC. This is not possible for type properties, which can only be set directly in the database, since they will affect every converter that includes that type of component.
8. Integration in EPICS

A first EPICS integration of the FGC3 controls deployed at TRIUMF was made by Rod Nussbaumer using the StreamDevice module (18). This was configured to map the important FGC properties to EPICS records and allows these properties to be polled or set.

It works well, but the polling rate is quite slow (about 1Hz). So a second phase is in development to exploit the published status data via UDP packets using the EPICS asynDriver package (19). This will allow a much faster refresh rate in EPICS and simplify the StreamDevice configuration.

The software needed to decode the UDP packets will be generated from the XML definition of the published status data, so coherence between the format of the UDP status packets and the interpretation of their contents into EPICS records will be easy to maintain.

Full details will be added to an updated version of this document, once phase two is working.
9. Integration in TANGO

Integration in TANGO is under development and will be ready by the start of 2019. A full description will be added to a future version of this document once it is working.
10. Portable FGC3 crate

A 3U, 32TE portable FGC3 crate is available to help with the development of the software that interacts with the FGCs. It can also be used to prototype the integration of an FGC3 into a power converter. The front of the crate is shown in Figure 20. It contains four modules:

- AC-DC power supply unit (PSU) (8TE)
- Extension card (6TE)
- Analog simulator of the voltage source and magnet circuit card (8TE)
- FGC3.1 (10TE)

The PSU can be powered from single phase 110 VAC or 230 VAC. The extension card gives front-panel access to the four ADC inputs and the two DAC outputs and allows the bus type to be defined. The bus type indicates the family of power converters being controlled, so it can influence the behaviour of the FGC3 FPGA and firmware.

![Portable FGC3 Crate](image)

**Figure 20 - Portable FGC3 Crate**

The firmware includes a full power converter and circuit simulator but using it does not exercise the analog components. For this, the analogue converter and magnet circuit simulator card has a variable time constant first-order filter on the DAC A channel, which is usually the Voltage Reference. The output of the first-order filter is returned on the ADC A and ADC B inputs, to simulate the DCCT current measurement signals. The DAC
A signal is returned on ADC C as the voltage measurement. In this way, the firmware can simulate the digital I/O signals while using the real analog signals.

Figure 21 - Back panel of the Portable FGC3 crate

The portable crate can be used to prototype the integration of the FGC3 with a power converter because of the connectors on the back panel. These are shown in Figure 21. Every signal from the FGC3 is available on a connector, which are easy to wire up. The electrical specification for all the signals is available in (5).

When the FGC3 is used to control a real power converter, the simulator card must be unplugged to allow the real measurements to be acquired by the ADCs, without interference from the simulated signals.

The FGC3 can be controlled using the USB interface (Type A) or through the Ethernet interface if a gateway is set up at the site – this is recommended because it allows PowerSpy to be used.

To arrange to have a portable FGC3 crate and for all other questions concerning the FGC3.1, please contact Nick Ziogas.
11. References


Note: The “(X)” suffix after the “CERN” is a workaround for a bug in Microsoft Word and does not mean anything. Because of the bug, citations need different authors, otherwise they get replaced by “-“.