POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Design and implementation of a new Controller for CERN's sub-ppm stable Current source

Laurea Magistrale in Electronics Engineering - Ingegneria Elettronica

Author: Valerio Nappi

Advisor: Prof. Federica Villa

Co-advisors: Miguel Cerqueira Bastos, Slawosz Uznanksi, Adrian Byszuk

Academic year: 2022-2023

---

1. Introduction

This work is about the design, development and implementation of the controller for the so-called CERN DCCT Calibrator (CDC), a programmable current source used to calibrate DC Current Transformers (DCCTs). It can output currents up to ±10 A and it is configurable for various applications, including checking and calibrating DCCTs and testing in-situ current references. At the core of the instrument, there is a DCCT.
DCCTs are used to measure the output current of power converters that power magnets for beam manipulation in CERN's facilities, and as such they have strict requirements on their precision. The work also covers the metrological assessment and evaluation of the most critical parameter of the final system, the 12-hour stability.

2. Controller overview

The original CDC, designed in the early 2000s [2], is based on a modular architecture. It injects a stable reference current into a winding with variable turns, wound on a DCCT core. The process creates magnetization in the core, which is detected and used in a feedback loop to generate a secondary current mirroring the reference such that the magnetization is nulled, creating a variable reference current that is output.
The CDC is now undergoing an upgrade to meet new requirements for the HiLumi upgrade, involving hardware and control system improvements. The control module uses the FGC (Function Generator Controller) Network Command-Response Protocol (NCRP) [4] to receive commands via ethernet. The control module uses three SPI links, a 1-Wire bus and an ADC in order to control the system.
The controller is based on a Xilinx Zynq-7000 SoC, which is a System-on-Chip that combines an ARM Cortex-A9 processor with a Xilinx 7-series FPGA. The ARM processor runs a Linux distribution, which is used to run the control software, while the FPGA is used to implement the SPI and 1-Wire interfaces.
The development process is based on the concept of Continuous Integration/Continuous Delivery (CI/CD) that allows for fast development cycles and high maintainability. The CI/CD pipeline is based on the GitLab platform, which is used for version control, issue tracking, and CI/CD.

2.1. The FGC Network Command-Response Protocol

The FGC Network Command-Response Protocol (NCRP) defines the communication between CERN's power converters and the accelerator complex. Since the CDC is part of the power converters infrastructure, it will adopt the FGC NCRP.
The fundamental unit of the language is the *property*, which is a named array of values that can be read or written. The protocol defines the *set* operation which serves to write the values in the property, as well as the *get* which performs a read operation on the property.

```
 !  s device:property.name[1, 3] values
!  g device:property.name[1, 3] options
```

Figure 1: Examples of NCRP commands

Referring to Figure 1, the command shall begin with the exclamation mark, followed by the *set* or *get* keyword. The *set* keyword is represented by the `s` character, while the *get* keyword is represented by the `g` character. The operation requires the device name, which is a string or an index identifying the device, followed by the property name, which is a string identifying the property. The property name can be followed by an array of indices, which are used to identify the elements of the property. The indices are enclosed in square brackets and can point to a single element or a range. The command is then followed by the values, represented as strings, which are separated by a comma. The *get* operation is similar, but it does not require the values, as it is a read operation. Instead, it can accept options.

3. Hardware

The hardware for the controller is based on a custom carrier board on which a commercial Zynq-7000-based module is connected. The carrier board implements the readout circuitry for voltage monitoring, which scales the four voltages to a range suitable for the ADC. It also implements the 1-Wire bus master, which is used to identify the modules connected to the system, together with the LVDS transceivers for the SPI busses, The carrier board is designed to be mounted on an IEEE Eurocard 3U slot, and it is powered by the 5V DC rail from the backplane.

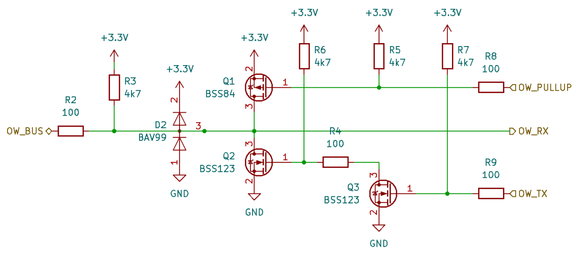

Figure 2: The 1-Wire bus driver circuit.

The onboard circuitry is powered by an onboard 3.3 V regulator.
The 1-Wire bus master is implemented using a discrete transistor circuit (Figure 2), which is used to generate the 1-Wire signals. The driver circuit features the support for the strong pullup, that is required to power the devices in the so-called "parasitic power" mode. The bus is also used to read the temperature of the modules.
The LVDS SPI busses are implemented using the SN65LVDS1DBV integrated circuit for the transmitting signals and the SN65LVDT2DBV IC for the receiving signals. These transduce the 3.3 V single-ended signals from the FPGA to the differential signals used by the modules.
The Zynq-7000 module is a Cora Z7 by Digilent, a commercial off-the-shelf product, which is used to implement the control system. It is connected to the carrier board via pin headers connectors, which provide power to the module, the ADC signals and the SPI and 1-Wire interface signals. The module also has an Ethernet connector, which is used to receive commands from the network.
The ADC is implemented using the on-chip XADC, which is a 12-bit ADC with 17 input channels, 6 of which are available on the Cora module as single-ended with an input range of 3.3V. The ADC is controlled via the AXI interface, which is used to read the ADC values and configure the ADC. A precision commercial opamp, the AD8658, is used in the voltage scaling of the power supply rails and anti-aliasing filter circuitry.

4. Gateware

The gateware is the part of the design that is implemented in the FPGA. It is developed with the Xilinx Vivado Design Suite. Some readily available logic modules (IP blocks) are integrated in the design. The core of the system is the Zynq PS (Programmable System) IP, which represents the configuration for the ARM processor and the system in general, such as the clock generation. The IPs on the FPGA interface with the PS are based on the AXI protocol, which is used as a communication bus throughout the system. The SPI interfaces are created instantiating three copies of the Xilinx QuadSPI IP, the 1-Wire interface is an instance of an IP from CERN and the ADC controller is again an IP from Xilinx, as the on-chip XADC is used. A few AXI to GPIO controllers complete the gateware design and are used to control status LEDs on the front panel and on the module.

5. Software

The software is composed of a Linux system on which a Python application is run. The Linux distribution is based on PetaLinux, which is used to generate the system image that is loaded into the Cora module. The application is based on the Python 3 programming language, and it is used to implement the control system, as well as the userspace drivers.

5.1. Linux distribution

The Linux distribution is based on PetaLinux, which is based on the Yocto Project, a framework for creating custom Linux distributions for embedded systems. The tool is used to generate and compile the Linux kernel, the root filesystem and the bootloader, which are then loaded into the Zynq module.
PetaLinux is a series of commands that wrap the Yocto Project and automatically generate some parts from the hardware files exported from Vivado. Further configuration is done via Menuconfig tools or configuration files. Yocto development is based on the concept of layers, which are collections of recipes. A recipe can introduce a functionality or modify a recipe from a layer below. The recipes in this project, for example, expand the SPI driver (`spidev.c`) device compatibility list to support the CDC devices, as well as modify the SSH daemon to store the RSA key in the persistent memory, so that the key is not regenerated at every boot. A custom recipe also creates an init script that is used to start the application at boot.

5.2. Application

The application is made of several layers. The innermost layer is the NCRP language definition, which contains classes for defining the most basic entities of the NCRP protocol, such as the `Command`, `Response`, `Property` and `Error` classes. The next layer is the NCRP language parser, which parses strings and returns NCRP `Command` objects if there are any. It is implemented using regular expressions, as the protocol is a regular language. The parser is used by the NCRP server, which is the core of the application. The server is a multithreaded application, that listens for incoming connections and spawns a new thread for each connection. Each thread is then used to handle the connection, interacting with the `CDC` object, that contains all internal states and methods. The `CDC` object contains a dictionary of `Property` objects, each of which contains the logic for the *set* and *get* operations, when allowed. The application is completed by the `SystemMonitor` module, which is responsible for checking for external systems' alarms and monitoring the power supply and the 1-Wire bus. In addition, the `cdc-http` module is used to implement a web server that is used to monitor the system via a web interface.

5.3. Userspace drivers

To control the SPI devices with the *spidev* driver, userspace drivers are written in Python. Since two of the modules share the same protocol, the drivers are implemented as a base class, the `SPIController`, which is then inherited by the two drivers. The drivers are implemented as a class that contains the logic to operate on the registers of the device and provides a higher-level control interface. For the third SPI bus, the driver is implemented as a class that contains the logic to control an AD7689 ADC connected via SPI. A second driver, the `Nullmeter` driver, is used to create a higher-level interface for the ADC, which is then used by the `CDC` object to monitor the voltages. The 1-Wire bus uses a Python driver that was already available from CERN, while the XADC does not need a userspace driver as a kernel driver is available.

6. CI/CD pipeline

Two CI/CD pipelines are implemented for the project. The first pipeline is used to build the

Linux distribution and to package it in an easy-to-deploy format, while the second pipeline is used to test the application and to build and deploy the documentation.
Petalinux is not particularly suited for git, as many paths in the project are hard-coded as absolute paths, so it is not even possible to move the project in the same machine, let alone in a different one. The authors suggest that the project shall be stored as “packaged”, but that approach defeats the purpose of having a version control system. Moreover, Petalinux requires a specific version of Ubuntu to run. To solve this issue, a Docker container is used to run the Petalinux toolchain. The container is then used in the CI/CD pipeline to build the Linux distribution, by creating each time a new project and copying over the modified files. The pipeline then runs the commands to package the distribution in an SD image file, that can then be burned on an SD card. A second partition is created on the SD card image, which contains the application and the persistent files. The packaged distribution is then uploaded as an artifact, which is a file that is stored on the GitLab server and can be downloaded from the web interface.
The second pipeline is used to test the application and to build and deploy the documentation. The pipeline is run on a virtual machine hosted on the CERN GitLab instance. The pipeline is divided into three stages: the first stage is used to run the tests with the *Pytest* framework, the second stage is used to build the documentation and the third stage is used to deploy the documentation. The documentation is built using *Sphinx*, generating part of the documentation automatically, from comments in the source code. The documentation is then deployed to a web server hosted on the CERN GitLab instance.

7. Metrology

Once the controller was operational and working, the final system was assembled and tested. The system was tested for the most critical parameter, the 12-hour stability. The 12-hour stability is defined as the maximum peak-to-peak fluctuation in the output current over twelve hours.
The first test of the system was done using an Alpha Electronics FNP resistor as a load. The

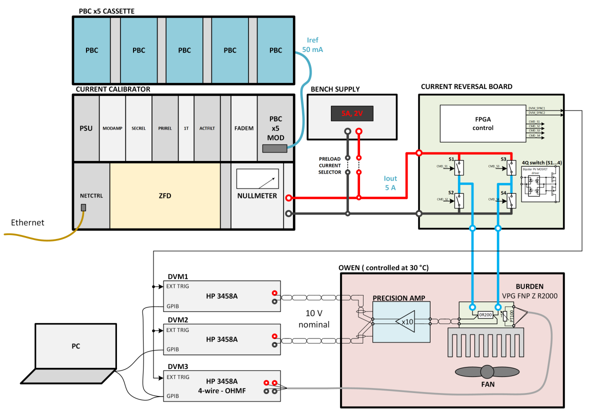

Figure 3: The 12-hour stability measurement setup. Images adapted from [5].

FNP resistor is a 0.2 Ω resistor with a 1 ppm/°C nominal temperature coefficient. The resistor was placed in a temperature-controlled chamber, and the temperature was set to 30 °C. This methodology is derived from the tests of the CDC after the upgrades [5].

7.0.1 Ovenized resistor test

Since the precision required by this measurement is in the order of fractions of ppm, the thermal EMF effects are significant. For this reason, the current in the resistor is reversed every second, so that the quantity to be measured changes sign, but since the connections are not altered, the thermal EMF effects are canceled out by subtracting two consequent measurements. The current is set to 5 A, with the 1 V on the resistor amplified by a x10 low noise amplifier. The output of the amplifier is then measured by two HP 3458A multimeters. This device is known to be one of the best voltage digitizers available. This first measurement showed

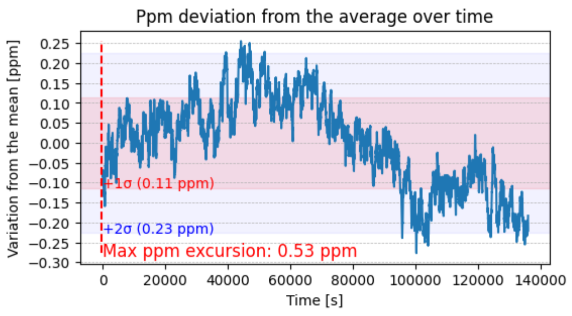

Figure 4: The 12-hour stability measurement with the HP 3458A multimeters

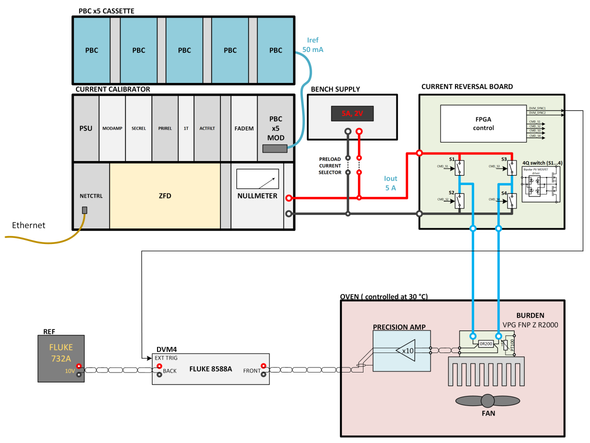

Figure 5: The 12-hour stability setup with the Fluke 8588A multimeter

how the 0.55 ppm 24-hour stability specified in the datasheet of the voltage digitizers was the limiting factor in the measurement. The setup is shown in Figure 3, while the measurement is shown in Figure 4.

7.0.2 Reference scanner test

Since the previous measurement technique did not yield the expected results, a different approach was used. A Fluke 8588A reference multimeter was used to measure the output voltage. The Fluke 8588A has a better 12-hour stability to the previous instrument, and it features an input scanner. This allows us to exploit the 20-min stability of 0.1 ppm, by switching every second to measure a Fluke 732B voltage reference with the back terminals, and the voltage with the front terminals. The Fluke 732B 10V voltage reference, which has been calibrated yearly and traced, has been proven to be stable and to drift by 0.2 ppm per year. These references, by design, have a very stable and predictable drift [3]. The measuring instrument is then calibrated at every read. The setup is shown in Figure 5, while the measurement is shown in Figure 6.
The CDC is expected to perform better than this, in particular regarding the initial drift. The resistor is kept at temperature before starting the test by a lab power supply that outputs 5 A, and the current is then switched to the CDC. If the current is not perfectly matched, or if the switch is too slow, the temperature of the resistor can change enough for the temperature coefficient Tc of the resistor to become relevant. A measurement of the Tc is performed by exploiting the same setup, keeping the current constant in the resistor and changing the temperature of the oven by 5 °C. With this technique, the Tc is measured to be ±2.8 ppm/K, which is vastly out of specification. With this Tc, the results of the variations in resistor temperature are in the order of 0.12 ppm. This, due to the unavailability of better samples of the resistor, is the final limiting factor of the measurement. A better measurement of the CDC stability can be performed by using better digitizers, but the results will still be at least affected by the Tc of the resistor.

7.0.3 HPM7177 test

The HPM7177 is a high-performance voltage digitizer, with a sample rate of 10 kSPS, proven to be stable to 0.2 ppm over 12 hours [1].
To increase the accuracy of the measurement, 5 units in parallel measure the output voltage, while a sixth unit keeps track of the trigger. The setup is shown in Figure 7, while the measurement is shown in Figure 8.
This test provides a slightly better measurement than the previous one, which finally confirms the stability to be better than 0.15 ppm, confirming that the final instrument is stable enough for the application. In this measurement, the contribution from the FNP resistor is still present and relevant and may account for a significant part of the measurement.

8. Conclusions

This thesis presented the design, development, and implementation of a new controller for CERN's sub-ppm stable Current source. The upgraded controller, based on a Xilinx Zynq-7000 SoC, integrates advanced hardware and

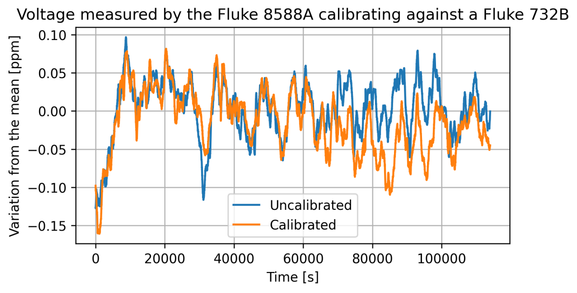

Figure 6: The 12-hour stability measurement with the Fluke 8588A multimeter

software elements to enhance reliability, maintainability and operational flexibility. The controller's design and implementation are characterized by its maintainability and testability, making the design valid for a long time, and making it adaptable for future upgrades or modifications. The CI/CD approach adopted in this project not only streamlined the development process, but also ensured that the system could evolve with changing requirements or constraints.

The metrological assessment of the controller's 12-hour stability highlights its capability to meet the stringent accuracy requirements essential for CERN's applications. The measurement techniques proposed during this project, including the use of high-precision instruments and parallel high-performance digitizer systems, contribute significantly to the assessment of the instrument's performance. Further improvements in the measurement techniques are possible, and they are expected to yield better results, in particular by selecting a better sample of the FNP resistor.

In conclusion, this thesis has successfully delivered a state-of-the-art controller for CERN's DCCT calibrator, demonstrating the proposed streamlined and modular design approach's effectiveness. The controller's lifetime is expected to be at least 10 years, and it is expected to be used in the calibration of DCCTs in CERN's accelerator complex for the next 20 years.

9. Acknowledgements

I want to thank my supervisor, Prof. Federica Villa, for her guidance and trust, and my co-supervisor Miguel Cerqueira Bastos for his insightful support. My gratitude also goes to Slawosz Uznanksi for his directional clarity, to Adrian Byszuk for his expertise in FPGA and embedded systems and to Nikolai Beev for his support in the metrological assessment. Special thanks to my colleagues in the CERN HPM and CCE teams for their collaborative spirit and technical support. I am grateful to my family, friends, and my girlfriend Silvia, for their support and encouragement throughout this journey.

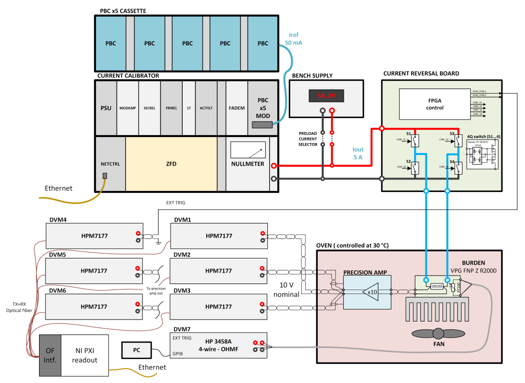

Figure 7: The 12-hour stability setup with the HPM7177 voltage digitizers

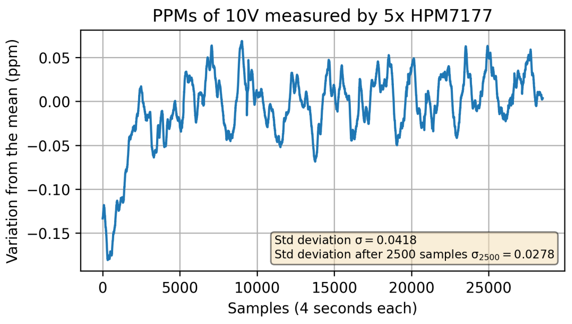

Figure 8: The 12-hour stability measurement with the HPM7177 voltage digitizers

References

[1] N. Beev, M. Cerqueira Bastos, M. Martino, D. Valuch, L. Palafox, and R. Behr. Design and metrological characterization of a digitizer for the highest precision magnet powering in the high luminosity large hadron collider, Due 2023.

[2] G. Fernqvist, B. Halvarsson, and J. Pett. The cern current calibrator-a new type of instrument. In *Conference Digest Conference on Precision Electromagnetic Measurements*, 2002.

[3] Fluke. Fractional ppm traceability using your fluke 734a/732b, 2011.

[4] Q. King, N. Laurentino Mendes, and M. Cejp. Fgc command/response protocol, 2018.

[5] M. Kovačić. The upgraded cern current calibrator techical seminar, 2022.