Internship report of Summer Student project

David Martin

July 25, 2016

Abstract

For my project at CERN, I worked in the TOTEM team with Michele Quinto and Francesco Cafagna as supervisors. Their team is currently working on an update on TOTEM that includes a module able to measure precisely the time of flight of particles emitted from the collision at CMS. With this additional data, TOTEM will be able to reconstruct precisely the point of the collision in CMS. The main problem posed for this new module is to provide a precise synchronized clock signal to both the TOTEM detectors situated 200 meters after and before CMS. In fact, due to some external parameters, as temperature, the length of the optical fiber guiding the clock signal can vary yielding thus a unwanted phase difference of the clock between the two detectors. The idea is to get rid of the noisy phase difference to make very precise time of flight measurement of the order of the picosecond. This is achieved by continuously measuring the phase difference and correcting the time measurements according to the current phase difference.

0.1 Introduction

My goal was more precisely to devise a C++ program controlling remotely the three most important devices of the module, in order to be able to do the first remote phase difference correction measurements. The three devices I had to control were the EDFA (laser amplifier), the optical switch and the network analyzer. The EDFA was used to amplify the optical signal after the loss produced by a multiplexer. The optical switch allows the user to choose on which optical fiber (for the roman pot after or before CMS) he will do the phase difference measurement. Finally, the network analyzer makes the measure of the phase difference between the input and reflected signal. For more detail one can have a look at [1]

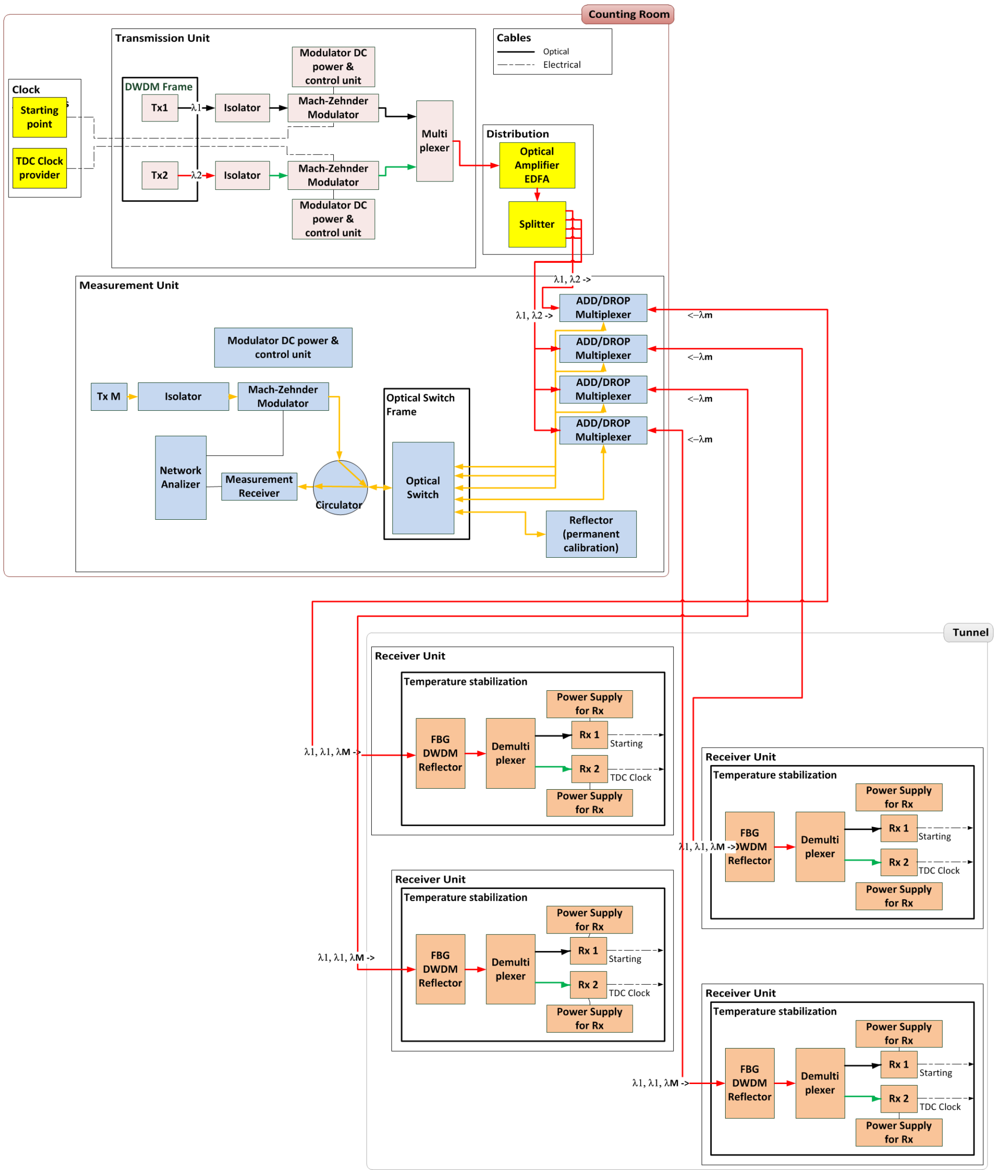


Figure 1: The time of flight measurement module

0.2 libraries

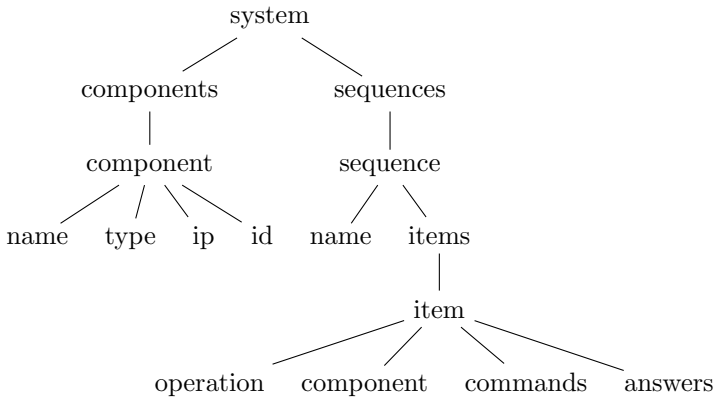
We wanted to use only open source libraries compatible with the programming environment of the TOTEM team for the remote control C++ script. We needed to communicate to the three devices using Ethernet protocol, but in the same time this communication via Ethernet should be able to exchange

commands understandable by the devices : SCPI commands. SCPI stands for Standards Commands for Programmable Instruments, it is a standard norm of language adopted by most of the electronic constructors in 1990. It consists roughly into keywords plus parameters that allow to configure the device, and query that allow user to ask for a measurement result for example. We turned toward an open source library named liblxi, which use the LAN eXtensions for Instrumentation (LXI) to communicate to data acquisition systems using Ethernet. It contains built-in functions to easily transmit SCPI commands to devices.
We also needed a way to store the information about the time measurement module, such as long SCPI command sequences doing one precise operation (readying the EDFA for measurement for example), or the IP adresses of the devices. We choose to write an xml file to store this information, the remote program should then parse this xml file, retrieve the information and be able to launch the operations remotely on the module. The open source library BOOST contains an xml parser and because the team still works on an the old 1.56 BOOST version, we choose the same version for our code for compatibility reason. Finally, we used :

- liblxi, available here https://github.com/lxi/liblxi , that we slightly modified to get it work in C++.
- BOOST 1.56, available here http://www.boost.org/

0.3 xml description

We divided the xml file into two major parts : one is the child 'components' and contains the names, the ip adresses of all the devices in the module. The other is the child 'sequences', which contains different SCPI 'sequence' to be launched remotely. Each 'sequence' has an 'operation' name describing its action, plus some 'command' which are the SCPI commands (found in [3], [2]) to be executed and finally some 'answer' which are the queries asked and that should return some errors report or measurement value. Right now, there are 5 different sequences. There is 'config edfa' which parametrize the laser amplification and set it on, 'set optical connection' which connect the optical input to the output channel number 3, 'unlock map-200' which unlock the chassis for further operations, 'phase delay measurement' which parametrize the network analyzer for measurements, and 'phase loop' which makes amplitude and phase measurements in loop and store them in a .txt file. Note that the user must run 'phase delay measurement' before 'phase loop' to have the network analyzer properly parametrized.



0.4 remote program

The script in itself starts with a loop on the xml childs 'components' described above and open a lxi connection to each of the devices, then ask their identity and print it on the screen. It then goes to a loop where the user must specify the sequence he wants to launch. The user can either type the name of the sequence to launch it, type 1 to see the list of all available sequences or type 0 to end the script and exit. The C++ program can run multiple options.

- modify the path to the xml file to parse using - -file path-to-parse

- modify the .txt file that receives the measurements using - -result file.txt
- modify the number of measurements using - -measure integer-number
- modify the waiting time in ms between two measurements using - -time integer-number
- see all the options available using - -help

0.5 First measurements

As the amplitude and phase delay measurements are comma separated in the .txt file, they can be easily extracted to the Libre Office Calc utility and we were thus able to plot the first remote measurements versus time.

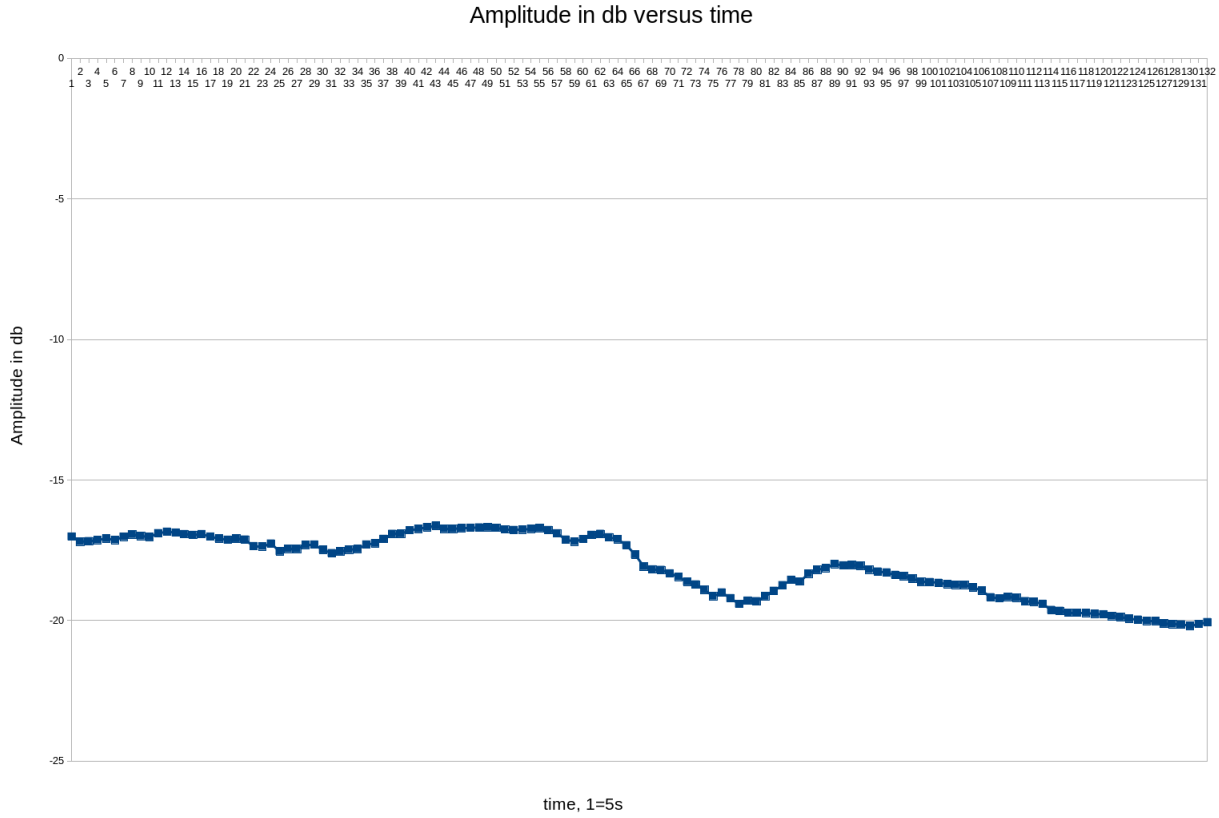


Figure 2: Amplitude versus time

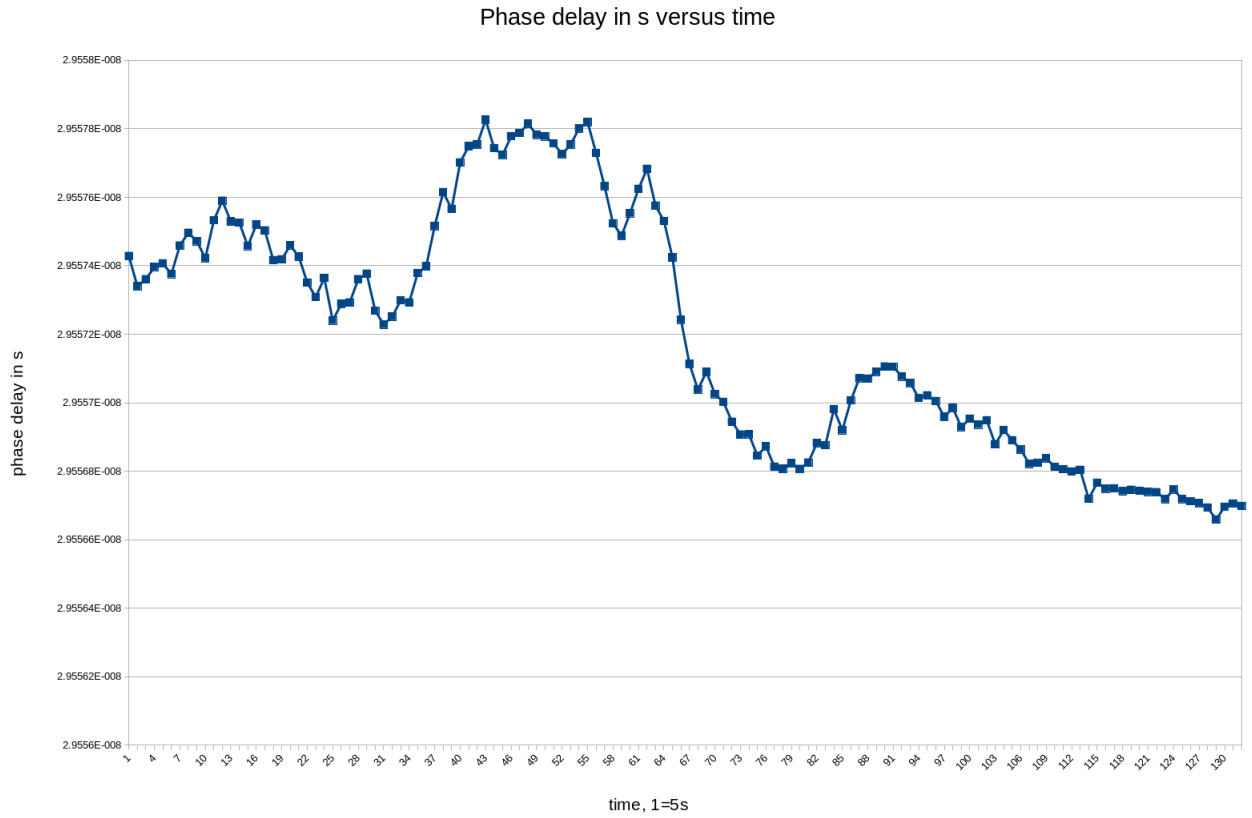


Figure 3: Phase delay versus time

Bibliography

[1] Francesco F Cafagna. The totem precision clock system. *CERN geneve*, 2016.

[2] JDSU Company. Map-200 programming manual.

[3] Rhode and Schwarz. Zvl vector network analyzer operating manual. *Rhode and Schwarz Company*, 2015.