FAST ORBIT FEEDBACK IMPLEMENTATION AT ALBA SYNCHROTRON

X. Serra-Gallifa, S. Blanch-Torné, D. Fernandez-Carreiras, A. Gutiérrez, Z. Martí, O. Matilla, J. Moldes, A. Olmos, R. Petrocelli, CELLS-ALBA Synchrotron, Cerdanyola del Vallès, Spain

Abstract

After the successful accelerator commissioning and with the facility already in operation one of the top short term objectives pointed out by accelerator division was the Fast Orbit Feedback implementation (FOFB). The target of the FOFB system is to hold the electron beam position at submicron range both in vertical and horizontal planes correcting the instabilities up to 120 Hz. This increased beam stability performance is considered a major asset for the beamlines user operation. To achieve this target, the orbit position is acquired from the 88 Libera BPMs at a 10 kHz sampling rate, distributed through an independent network and the corrections are calculated and sent to the 176 power supplies (PSU) that drive the corrector coils. This correction loop is executed at 10 kHz and the total latency of the system is characterized and minimized optimizing the bandwidth response.

INTRODUCTION

ALBA [1] is a 3 GeV third generation synchrotron light source of 268m of perimeter located in Cerdanyola del Vallès (Barcelona). The synchrotron gave its first light to the beamlines in 2011 and the first beamlines commenced operation one year ago.

From almost the beginning of construction of ALBA, FOFB system was used in other light sources as a technique for beam stabilization. For this reason preliminary studies [2] about noise sources were done and the equipment needed for the correction was foreseen and installed. However, FOFB is not an essential need for a synchrotron use and the final development was delayed to a second phase. This time offers a good opportunity to analyse the noise sources before applying FOFB corrections, showing that Alba have an amazing low noise in electron orbit without FOFB, see figures 1 and 2.

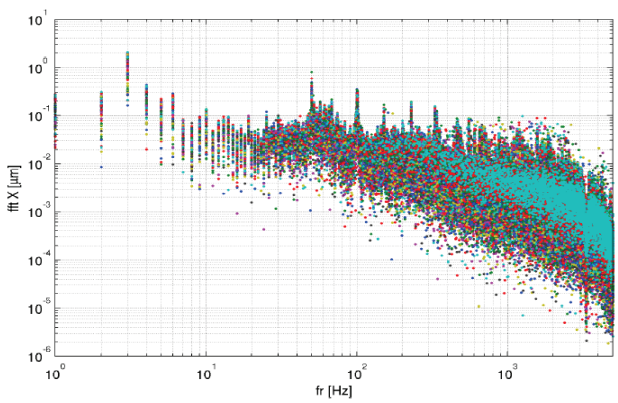

Figure 1: Horizontal FFT of 88 BPM during 1 second.

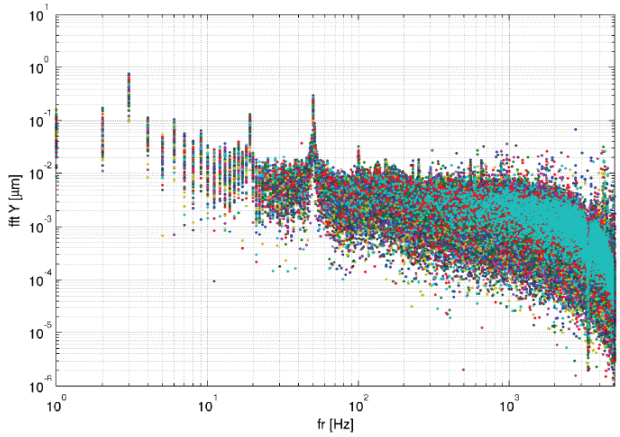

Figure 2: Vertical FFT of 88 BPM readings during 1 second.

At Alba with booster off, the beam stability is in a 10% beam size window, and this is the target value FOFB, but without FOFB. However, when booster is on, there is a significant increase of noise near 3Hz that kick us out of the target. This year the top-up system will start its commissioning and booster will be permanently on, adding the aim of the facility of a continuous improvement its performance FOFB have to be implemented.

THE SYSTEM IN ALBA

Alba installed from the beginning 104 Libera Brillance electron beam monitors (eBPM), which were used for diagnostics and for the slow orbit feedback. Also there are 88 sextupoles with 2 dipoles for correction of each axis, which are powered by OCEM PSU. These elements are distributed in 16 sectors, each of them has cPCI CPU based system with a 3U card containing FPGA responsible to read the eBPM data and write PSU values. The eBPMs are connected in a nested ring configuration, see figure 3.

For the first phase of FOFB, Alba takes advantage of spare FPGA boards of timing system compatible with the eBPM optical communications to speed up the project. With these boards and the help of Diamond Light Source (DLS), we have been able to read the position of the 104 eBPM at 10 kHz using the DLS Communication Controller [3].

ISBN 978-3-95450-139-7

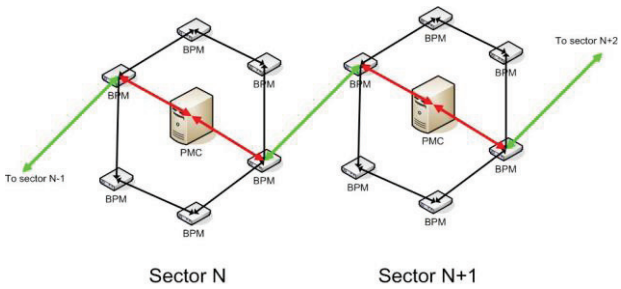

Figure 3: Schema of the FOFB network configuration. The last sector is bounded with the first one.

In a second phase a tender has been published to buy new FPGA boards capable to fulfil all the project requirements: communicate with eBPM to acquire the beam position implementing network redundancy, being capable of processing the correction to be applied, and write the data directly to the PSU which is planned as mid-term objective.

Software Approach: Linux Soft Real Time

From the beginning it was in mind the idea of using the already installed cPCI chassis and CPUs, installing some kind of real time operating system and developing our own software.

In order to evaluate the different alternatives in a realistic environment a complete test bench was set up in the laboratory (see figure 4), with all the necessary components:

- cPCI chassis.
- cPCI CPU (three models were evaluated).
- FOFB sniffer cPCI card, reused from timing.
- PSU communication controller (PSC-IP2 on cPCI).
- Optical Link interface (PSITBO)
- Corrector power supply units (PSU).
- Corrector coil connected to one of the PSU.

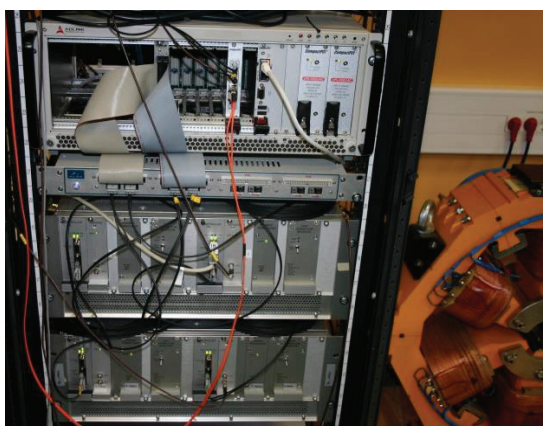

Figure 4: Test bench of FOFB. From top to bottom the cPCI chassis (with the CPU, FOFB sniffer and PSC-IP2), PSITBO and 4 PSU. At right the corrector magnets.

The first decision was to select a suitable operating system. We discarded commercial options for several reasons, mainly due to their high cost, the fact that it would tie us to a given proprietary development platform and that it may require a change in the hardware platform. Furthermore, this required rewriting the drivers for our cPCI cards, which were already developed and running in the linux kernel. We decided to try a standard 2.6.27 linux kernel compiled with the CONFIG_PREEMPT flag. This provides soft real time performance, not hard real time. We also tried a kernel compiled with the CONFIG_PREEMPT_RT flag, which theoretically can provide hard real time, depending on the underlying hardware. After evaluating it we checked that in our case this didn't gave us a significant advantage and discarded its usage, trying to keep things as simple and standard as possible. Furthermore, in our case soft real time is enough: the fact of sporadically loosing one of the correction loops is not a critical issue for us. Our standard CPU was single core, but we soon found out that we were going to need a multicore CPU, and hence we also need the kernel to be compiled with the CONFIG_SMP flag.

Closely related with the selected OS decision was the hardware to be used. First tests with our standard cPCI CPU revealed that it was clearly obsolete in terms of performance. It is an Adlink cPCI-3840, a single core machine. It was obvious that we needed a more powerful machine. Two different CPUs were evaluated, the Intel Core Duo Adlink cPCI-3965 and the quad core version of the Intel i7 Adlink cPCI-3970.

The job to be done by our software process consists on three steps:

- Reading the orbit data of the liberas from the FOFB sniffer memory and moving this data through the PCI bus into the CPU.
- Using this data, computing the correction to be applied to each coil.
- Writing this correction set point to the PSI PSC-IP2 interface card through the PCI bus. This card will pass this set point to the PSITBO, which will take care of writing it to the PSU.

For getting real time performance, some steps have to be performed by our process before entering the main endless correction loop. These are:

- Set real time scheduler and the highest real time priority for our process. We also check that no other process has real time priority.
- Reserve a CPU core for our process, by moving all possible processes into the other cores. Note that some kernel processes cannot be moved.
- Move all the IRQs except the one raised by the sniffer card to the cores not reserved for our process.

There's a CPU per each one of the 16 storage ring sectors, each one controlling up to 12 PSU. This loop will repeat continuously and it has to be completed within 100 µs. The start of each loop is synchronized (always within software limitations) because the reading of the sniffer card blocks until new data is available. The computation of corrections and writing into the PSI PSC-IP2 cards is not a constant time (again due to software limitations), but it seems to be good enough.

ISBN 978-3-95450-139-7

The software tests consisted on measuring the partial latency of each one of the steps of the loop and the total latency, and then report this to a log file. The study of this log file allowed us to check the mean and standard deviation of each of the steps and the total loop latency. We were especially interested in the number of loops that exceeded the 100 µs constraint: that would indicate that a continuous data rate of 10kHz could not be achieved.
The software tests revealed that with the single core CPU the system is useless. With the dual core CPU some very little number of loops randomly exceed the 100 µs limit. This would very rarely produce beam instabilities once considering the bandwidth of each component of the system, but in a high CPU usage scenario it may produce some difficult to predict instabilities. With the quad core CPU the system seems to be fully stable: we detected no loops exceeding the time limit, even when loading it.

Hardware testing consisted on checking that our process was fast enough to write data at 10 kHz in the PS controller. To do this we first manipulated the FPGA in the sniffer board so it sets a TTL output every time new data is available (this gives a perfect 10 kHz square signal, with very little jitter). We then configured a diagnostics TTL output of the PSU to output the set point that this controller receives from our software process. We then ran our main software loop slightly modified to alternatively write 2 different values as corrector set points, discarding the computed values. We also logged the loops that exceeded the 100 µs limit. Then we connected both TTL signals to an oscilloscope and we could see two in phase square signals, with some jitter in the PS controller signal. We saw that sporadic glitches can be detected, in which the signals got desynchronized. We first thought that the cause for this was that the loop was exceeding the time limit, but by checking the logs we could see that this was not happening. We have not yet determined the cause of the problem, but since it happens sporadically (less than 1%) and they will be filtered by the bandwidth of the system, so we think that the system is still usable.

Real world tests in the storage ring are still pending to be performed, and this will be the real test to check if the system is feasible with this software approach.

There are many advantages of this solution: correction algorithm can be easily modified and loaded, settings of the system can be easily modified, the current hardware can be reused (only 16 new CPUs have to be acquired) and by using a linux kernel the entire software platform can be reused as it is: only minor modifications on the drivers for both the FOFB sniffer and the PSI PSC-IP2 cards were required.

The obvious cons are that software is less predictable (compared with hardware) and has bigger jitter.

Hardware: New FPGA Board

Nowadays, the sniffer FPGA boards used to read the data from eBPMs are a spares of cPCI boards before used in timing system application. These boards don't fit exactly with the features required for FOFB, in the fact that breaks the redundancy of eBPM network. In addition, the PSI PSC-IP2 FPGA board, which is in charge of optical link to the PSU, are discontinued and more stock spares is desirable in a the mid future. For these reasons, a tender to buy new FPGA boards was needed to implement the reading of eBPM and the writing to the PSU.

The process is closed and the final boards that will be used are expected to be received during the spring of 2014. In next step, the DLS communication controller will be integrated to replace the FOFB sniffers. Then the functionality of the PSI PSC-IP2 optical link will be implemented and the PSI board will be replaced too. Finally, if the software control of FOFB cannot achieve the targets using only the CPU, the FPGA processing capabilities will assist in the process while keeping algorithm implementation flexibility as much as possible.

CONCLUSION

ALBA FOFB system is based in a fully flexible software based approach which was selected for its excellent flexibility for algorithm correction implementation. The different tests performed in the laboratory are conclusive about its feasibility. From third quarter of 2013 tests with real beam are planned to confirm it. In parallel a tender for FPGA based cPCI boards have been done issued and new boards are expected for in spring 2014. This new boards will be used to implement eBPM network redundancy, set the current correction to the PSU and, if needed, help in the correction algorithm processing. The facility target is having a fully operational FOFB system during 2014.

ACKNOWLEDGMENT

I would like to start the acknowledges with the accelerations division of ALBA that offer their time explaining all they know of FOFB in special Angel Olmos[4], Ubaldo Iriso and Marc Muñoz. Other important acknowledge is for all the people that have implemented a FOFB in their facilities easing the implementation at ALBA, here we thank the help of Guenther Rehm, Michael Abbott and Isa Uzun from DLS; Jean-Marc Koch and Francis Epaud from ESRF; and Marco Lonza from ELETTRA, for the support they gave us. Finally, we would like to thank to Alejandro Homs from ESRF for the collaboration in the software approach.

REFERENCES

[1] D. Fernández-Carreiras *et al.*, "The design of the Alba Control System. A Cost-Effective Distributed Hardware and Software Architecture", ICALEPCS 2011, Grenoble, FRBHMUST01.

[2] D. Beltran and M.Muñoz, "Initial Design of a Global Fast Orbit FeedBack System for the Alba Synchrotron", ICALEPS 2007, RPPA23.

[3] I. S. Uzun *et al.*, "Initial Design of the Fast Orbit Feedback system for Diamond Light Source", ICALEPS 2005, P3_030.

[4] A. Olmos *et al.*, "First Steps Towards a Fast Orbit FeedBack at Alba", IBIC 2013.

ISBN 978-3-95450-139-7