Spread Spectrum Channel Sounder Implementation with USRP Platforms

Adrien LE NAOUR, Olivier GOUBET, Christophe MOY, Pierre LERAY

SUPELEC/IETR
Avenue de la Boulais, CS 47601, 35576, Cesson-Sévigné Cedex, France
christophe.moy@supelec.fr

***Abstract* — This paper shows results of a student project on the implementation of a spread spectrum channel sounder using USRP platforms and GNU Radio environment. Such an implementation would have required months of development (ASIC, board design and manufacturing, RF design, etc.) 15 years ago. A few weeks of part time work for 2 students were only necessary here thanks to a software defined radio approach. It is planned that this channel sounder could be used in a future cognitive radio context for environment awareness.**

***Index terms*— USRP, GNU radio, spread spectrum, education**

I. INTRODUCTION

Implementing radios with students is very challenging as many aspects of radio engineering areas have to be jointly addressed and mastered:

- Digital communications basics,
- Analog RF,
- RF to BB conversion,
- Digital BB processing implementation.

This makes such an activity only possible at the latest stages of students curiculum. This paper illustrates that thanks to a software radio approach 0, this can be made with undegraduate students with a reduced background in only a few of these fields.

Only a decade and a half ago, the implementation of a radio communication link, transmitter and receiver, starting from scratch was a task requiring months of development for a specialist in both signal processing and electronics. In the context of a spread spectrum channel sounder of a few MHz of bandwidth (less than 1 µs of delay resolution) for example, this implied to dimension, specify and implement the digital processing operations in a FPGA for instance, and to design a board supporting the necessary electronics around [2]. Then frequency translation from baseband to RF required the specification and the design of another board, with a lot of analog imperfections to solve (IQ imbalance, carrier leakage, etc.). This could be even more challenging for higher data bandwidth, implying the use of ASICs instead of FPGAs [3]. Depending on the carrier frequency also, the range of difficulty varies a lot.

In a software defined radio (SDR) perspective, many of these drawbacks fall down as one can re-use sub-parts, if not all parts, of an existing radio system, and just adapt its operation through code reprogramming. SDR philosophy is also a catalizer for open source code and design tools sharing in the radio field.

Ettus Research (now a part of National Instrument) provides a hardware platform supporting any carrier frequency between DC and 4.4 GHz, for a bandwidth up to a few MHz depending on the board [4]. Then RF to baseband conversion sub-system is ready for prototyping (except specific requirements in terms of transmitting power, depending on each application).

For digital processing, software radio paradigm priviledges a pure software implementation of radio processing when possible. In conjunction with USRPs platforms, this can be done for baseband processing at least. USRP platforms are connected to a host computer through USB or ethernet connexions. The GNU Radio Companion [5] is a friendly environment for the development of baseband processing thanks to a set of pre-defined block sets, which enables to build a complete radio chain without developping specific code (most of the time).

Next part of the paper describes the project we propose as a lab to students, before they graduate for engineering diploma. Part III exposes how students first studied their application using Ptolemy II simulator, another open source tooling for embedded systems design. Part IV details how the channel sounder has been implemented

in GNU radio environment for USRP platforms. Part V gives some perspective on how using the channel sounder in a communication system mitigating channel effects. Finally, channel sounding implementation results using last N210 platform version and a home made C++ environment are given in part VI, as well as concluding remarks in a last section.

II. PROJECT DESCRIPTION

Channel sounding is a very pedagogic and a didactic topic for teaching telecommunications to students. Implementing a channel sounder enables to point out in real conditions all the problems due to propagation, channel effects, and front-end imperfections, such as transmitter and receiver local oscillators synchronization issues. In other words, all that is often hidden when learning digital communications at the beginning.

The channel sounding scheme proposed here is based on direct sequence spread spectrum technique [6], as shown in Figure 1.

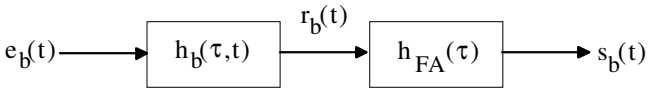

Figure 1 – Channel sounding principles

The signal $e_b(t)$ sent by the transmitter (Tx) is a pseudo-random code c(t). After a matched filter $h_{FA}(t)$ to the code c(t) at the receiver (Rx), the obtained result $s_b(t)$ is an approximation of the channel impulse response $h_b(\tau,t)$, including all analog imperfections if we consider the channel in a wide sense (i.e. including analog RF of Tx and Rx). As stated by the following equations:

$$e_b(t) = c(t)$$
$$h_{FA}(t) = c(\xi - t)$$
$$r_b(t) = (c \otimes h_b)(t)$$

with $\otimes$ the correlation operator, and $\xi = L_c.T_c$

L_c is the number of chips of a code sequence

T_c is the chip length

As a consequence:

$$s_b(t) = (c \otimes h_b \otimes h_{FA})(t) = (c \otimes h_{FA} \otimes h_b)(t) = (Rc \otimes h_b)(t)$$

where $R_c(t)$ is the autocorrelation function of code c(t):

$$R_c(t) = \int_{-\infty}^{+\infty} c(\tau) \cdot c(t-\tau) \cdot d\tau$$

and: $$h_b(\tau,t) = \sum_{k=0}^{K-1} \alpha_k(t) \cdot e^{-j\theta_k(t)} \cdot \delta(\tau - \tau_k(t))$$

for a channel of K paths, each with a delay τ_k and a phase θ_k. Consequently:

$$s_b(t) = \sum_{k=0}^{K-1} \alpha_k(t) \cdot e^{-j\theta_k(t)} \cdot R_c(t - \xi - \tau_k) \cdot d\tau$$

For a m-sequence code, the autocorrelation function $R_c(t)$ is made of a unique peak of width $2.T_c$ at value L_c. Then the smaller T_c (i.e. the larger the bandwidth $W=1/T_c$) and the longer L_c, the closer is $s_b(t)$ to the real channel impulse response $h_b(\tau,t)$. Moreover, the resolution of the sounder is $T_c=1/W$.

The same structure maybe used also in a communication system as a way to compensate channel effects, such as in a RAKE receiver for instance [3][6][7].

As shown in the next section, students have made simulations using Ptolemy II software as a first approach to better understand channel sounding principles. Then they converted the simulations into reality using a software radio approach through GNU Radio Companion environment and hardware USRP boards.

III. PTOLEMY SIMULATION

A. Presentation

Ptolemy II is an open source framework developped by the UC Berkerley at EECS department of Edward Lee. Our simulations use Synchronous Dataflow (SDF) model. Ptolemy simulations made the students understand the basic principles involved in the implementation of the channel sounder, such as m-sequences (maximum-length sequences) correlation, Root Raised Cosine (RRC) filtering and multiple access (CDMA), before they had a first teaching in digital communications.

We chose m-sequences generated with 7 shift registers ($L_c = 2^7-1$) for their good autocorrelations properties (at zero delay, but also at non zero delay, in opposition to Hadamard codes), and rather good cross-correlation properties with other sequences (for CDMA purposes). They reach a peak of 127 at the origin and remain equal to -1 everywhere else. Sequences are 127 chips long. Using high values for the m parameter improves the signal to noise ratio (SNR) after despreading at Rx.

B. Simulation of a spread spectrum communication chain

Such an approach is the basis of direct sequence spread spectrum communications, when signal is spread with a specific pseudo-noise sequence. Each data set to be transmitted, is combined with an m-sequence. Thus, sending one bit actually requires sending 127 chips. The longer the spreading sequence, the better the processing gain, and thus the recovered SNR at Rx. However, this also decreases data transfer rate. When received, the signal is synchronously multiplied by the same m-sequence [6]. Because of their autocorrelation properties, the data can be reconstructed. Another way (used in this paper) consists in having a filter matched to the same m-sequence at Rx and select the peak sign at autocorrelation point. This avoids synchronization issues and enables channel compensation [8].

Transmitting a signal on a shared spectrum requires filtering. Root raised cosine filters maximize the signal to noise ratio and minimize intersymbol interferences following Nyquist criteria. Excess bandwidth is set to 0.8 and interpolation in the transmitter filter is set to 4 here.

Cross-correlation between two different m-sequences is negligible compared to the correlation peak's height (neglecting near-far effect). It is then possible to multiplex several transmissions on the same radio spectrum if they are coded with distinctive m-sequences. This principle is used in CDMA (Code Division Mutliple Access) technology with multiple communications between transmitters and receivers sharing the same channel. Each transmitter uses its own pseudo-noise sequence to spread the data signal. Then all the signals are added and transmitted through the channel. Each receiver despreads the resultant signal with the pseudo-noise code used by the transmitter it communicates with. Because of the cross-correlation properties mentionned earlier, only one data signal is despread and recovered, the other other being scrambled and considered as noise. Figure 2 illustrates this principle using Ptolemy II simulator.

In the context of a channel with imperfections, data can be transmitted inphase and a code for channel sounding on quadrature, so that channel estimation enables to correct channel effects, as in UMTS uplink for instance.

C. Simulation of a channel sounder

Channel imperfections such as multiple paths and Gaussian noise can be considered. The channel sounder allows to get the impulse response of the channel.

A first version of the sounding chain is shown in Figure 3 (no phase rotation) in the Ptolemy environment. Two echoes (multipaths) are added here to the direct path, as well as additive white gaussian noise (AWGN) to simulate the channel.

Figure 4 shows the obtained channel impulse response in Ptolemy simulations in a noisy environment.

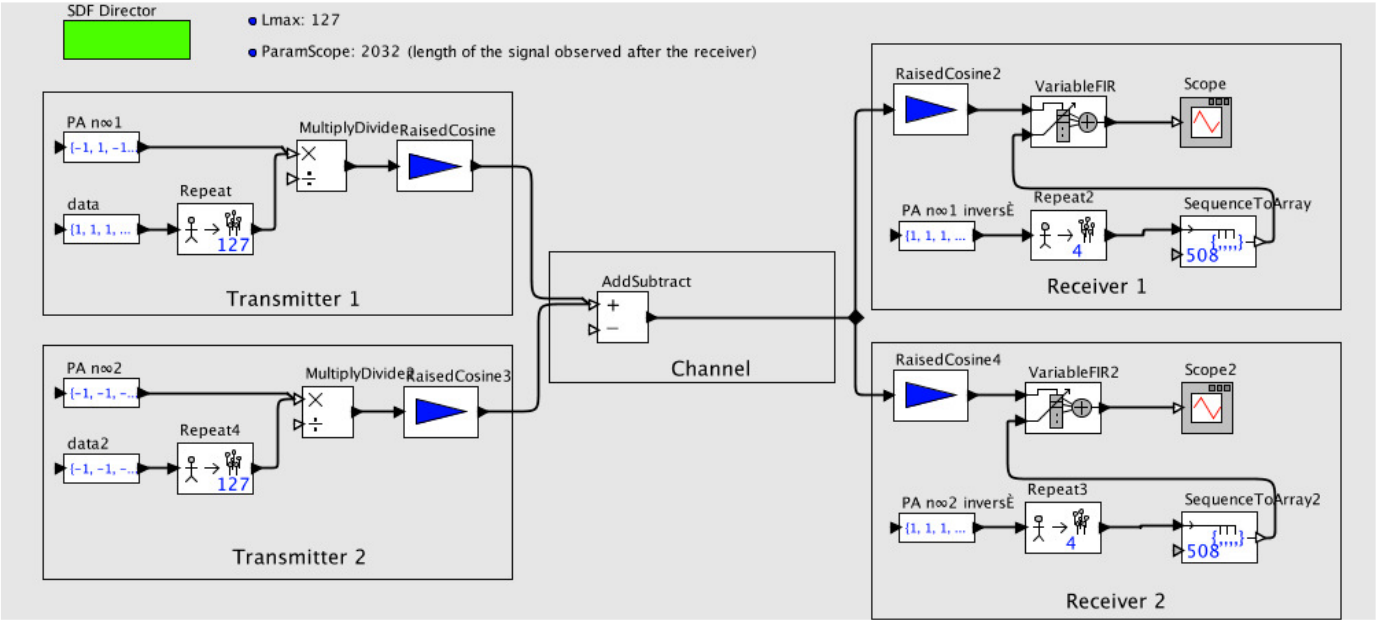

Figure 2 – Simuling CDMA with Ptolemy

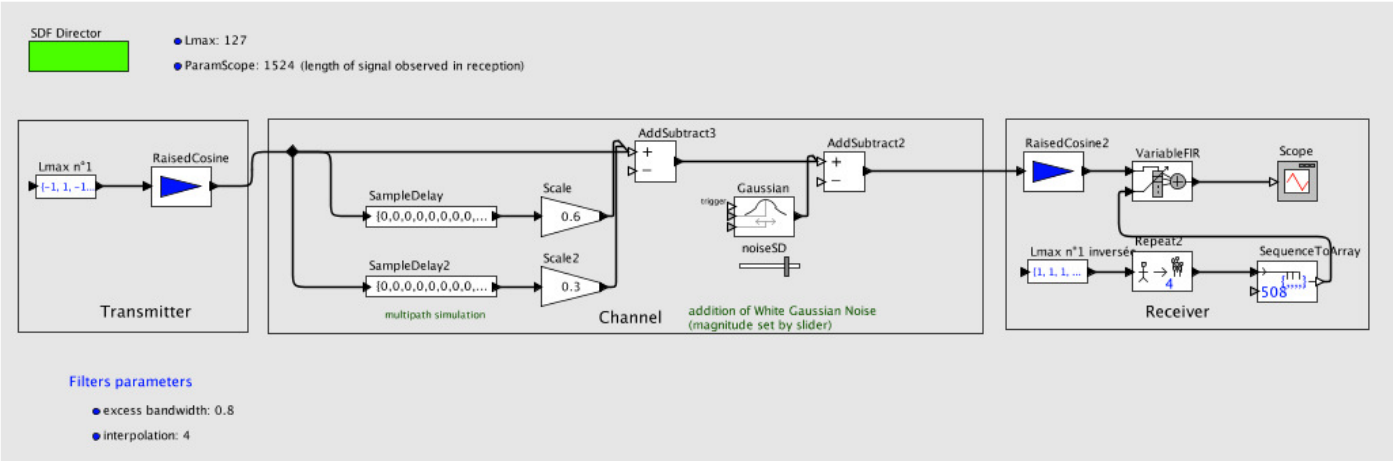

Figure 3 – Transmission chain with multipath and AWGN

Three peaks corresponding to the direct path and the two echos can be observed. We also can see the effect of the white gaussian noise. The noise can be mitigated while averaging the impulse response on several autocorrelation sequences when the channel is stationnary [2][7].

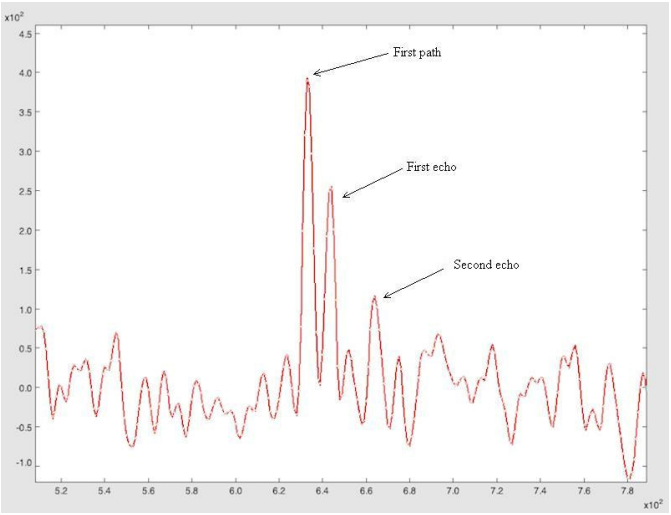

Figure 4 – Channel impulse response

We can see that beyond first objective of Ptolemy which targets the modeling and design of real-time embedded systems, Ptolemy II may also be used as a valuable tool for educational purpose, in particular in communications.

IV. Channel Sounder Implementation

GNU Radio Companion (GRC) is a free development environment for software radio. Familiarizing with GNU Radio project [5] is easy due to similarities between Ptolemy II and GRC processing blocks. It has been first checked that transposed in GRC environment, without connecting to radio platforms, the simulations give similar results to Ptolemy.

A. Real implementation on USRP: transmitter

Two stations are required in order to implement the channel sounder on USRP platforms: a transmitter capable of generating m-sequences and a receiver capable of correlating the received signal with the m-sequence used at Tx. Each station requires a host computer equipped with one USRP1 platform (via USB connection). GNU Radio is used through GRC to conceive the designs.

In GRC, in-phase component (I) and quadrature component (Q) components are sent to the Tx USRP platform through the imaginary and real parts of the *USRP sink block* input of Figure 5. In-phase component is dedicated to the channel sounder, e.g. it is made of successive m-sequences. Nothing is transmitted through the quadrature component for the channel sounder (it could be used to transmit data as exposed in III.B).

B. Real implementation on USRP: receiver

The receiver of Figure 6 has to include a correlator block. It is made using a FIR filter whose coefficients are made of the m-sequence code reversed in time, so that the convolution becomes a correlation [2][7]. This is the principle of a matched filter (to the transmitted code). The working frequency is 2.4 GHz ; however the actual frequency of Tx and Rx local oscillators in USRP devices can deviate from that frequency (the difference is up to a few kHz).

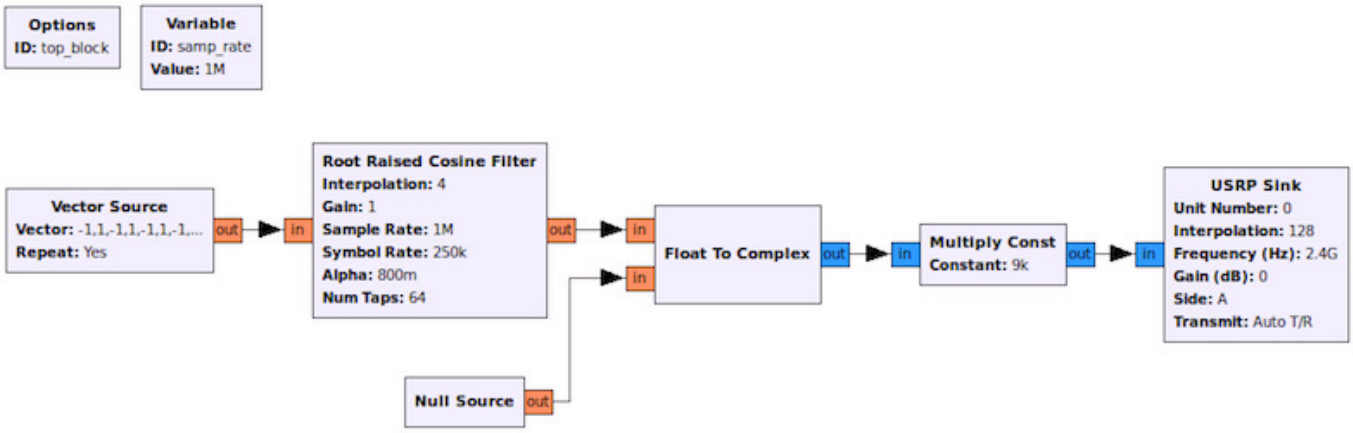

Figure 5 – Channel sounder transmitter

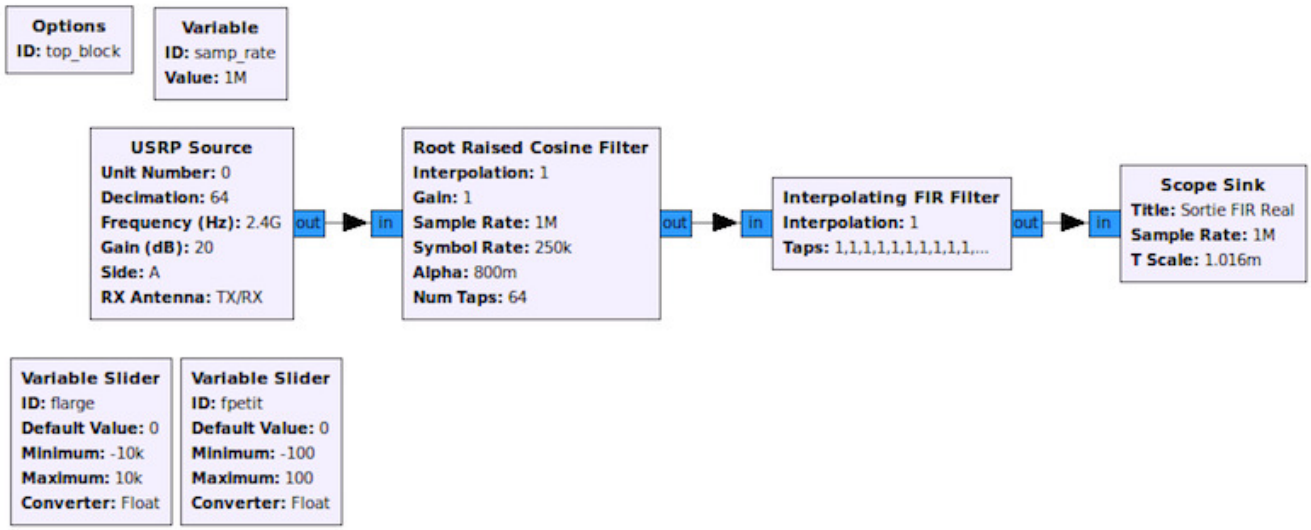

Figure 6 – Channel sounder receiver

In order to precisely tune the frequency of local oscillators and reduce the frequency gap between the receiver and the transmitter, two sliders were added. Students noticed that the correlator can not support a too important frequency shift. At least the phase shift between the two oscillators should not be significant (let's say less than 20°) during the correlation on L_c chips.

The measured channel impulse response at the output of the correlator is shown in Figure 7 on several successive correlation periods with I component on top and Q component below. The observation of the impulse response provides useful information about the imperfections introduced by the channel. We can see that in these conditions of experimentation, the signal to noise ratio is high ; indeed, USRP platforms are side by side and signal amplification is within its maximal range. Multiple paths are absent, because of short-range transmission. But the carriers frequency desynchronisation is clearly visible as correlation peaks rotate over the course of time, whereas we should only see positive peaks in I component and nothing except AWGN in Q. We see on Figure 7 the desynchronisation phenomenon while observing the signal over 10 sequences.

Phase can be extracted from this sounding information and could be used to compensate in real-time the frequency shift between Tx and Rx oscillators in a communication link [2][7]. But in a communication context, we may get rid off synchronization issue by other means, as exposed in the next section.

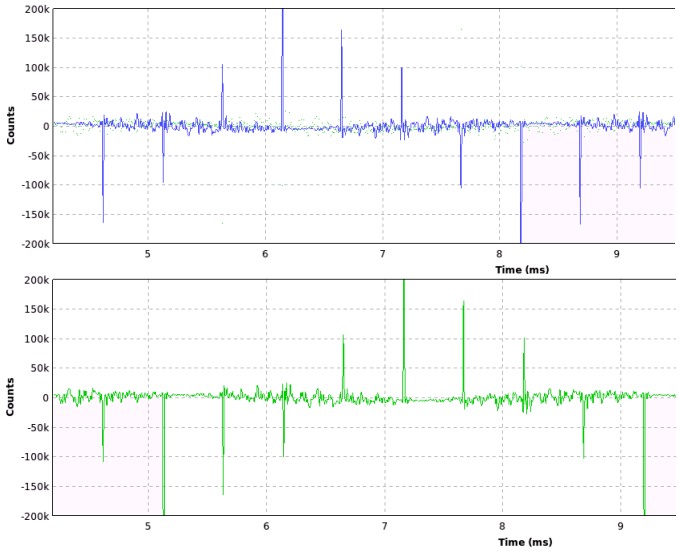

Figure 7 – Channel impulse response : top (I), low (Q)

V. SPREAD SPECTRUM COMMUNICATION

Thanks to the easy implementation of radio systems with a software radio approach, students also investigated spread spectrum communications, beyond channel sounding. The specific issue of carrier synchronization is illustrated here.

A. Synchronization with a Costas Loop

First, the Costas Loop was considered to deal with the desynchronization issue. Once the signal is properly synchronized, a peak detection function can be added at the output of the correlator. A positive peak means that a '1' was transmitted, and a negative peak means that a '0' was transmitted.

However, students faced that the Costas Loop was not enough to completely synchronize the carriers, as the phase regularly shifts from 0° to 180°. Costas Loop based designs did not appear viable, since the loop could not lock when the desynchronization was too high. The reason could be that students were not skilled enough in digital communications to find the good parameters values, during their project, or because the desynchronization was out of range of the Costas loop. A complete study of the Costas loop provided in the GNU environnement would have been neccessary for that.

B. RAKE-like channel compensation

A second solution that was envisioned consists in using the channel impulse response obtained with the channel sounder to compensate the phase-shift in a direct sequence spread spectrum communication link. In this context, communication data are spread on Q component with another code of the same length L_c than the sounding code of I component. For each '1' data (respectively '0') the code sequence (respectively the opposite sequence) is sent.

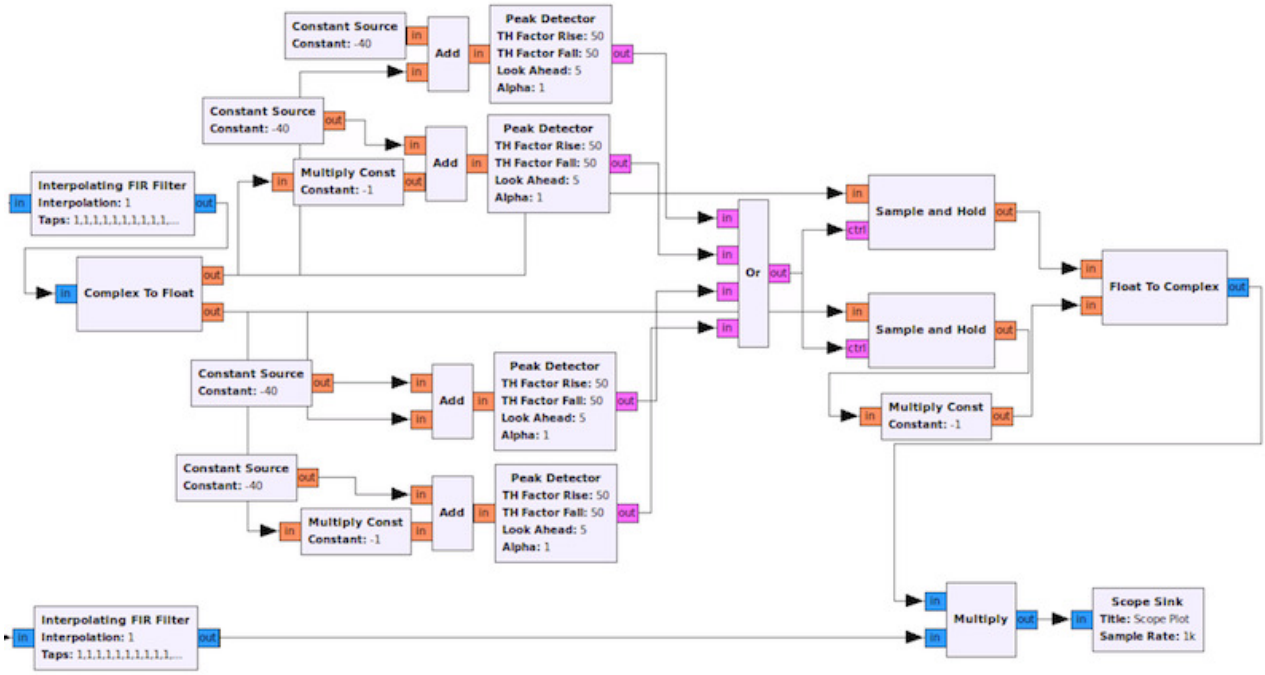

Figure 8 – phase-shift compensation algorithm using a RAKE-like approach

The phase-shift can be measured on the impulse response given by the correlation on I component code, when a correlation peak appears. The receiver design is shown in Figure 8. However, an efficient correction requires that the correlation function is performed while carriers shift stays negilgible. Simulations showed that a desynchronisation of 400 Hz can be compensated at a chip rate of 1 MHz (when correlation is 127 µs long). But students observed a desynchronization of several khz between the two platforms local oscillators, thus requiring a ten times higher chip rates (10 MHz). But students could not reach that transfer rate with USRP 1 platforms connected via USB link to the host platform.

As the communication chain was developped and simulated with the addition of the imperfections due to desynchronization, this allowed to validate the design for lower values of desynchronization only (than in reality). Results are shown on Figure 9. Sequence {1,1,-1} was transmitted with a simulated desynchronization of 150 Hz at a chip rate of 1 MHz. Successive sequences of two positive peaks, followed by one negative peak are obtained, as expected.

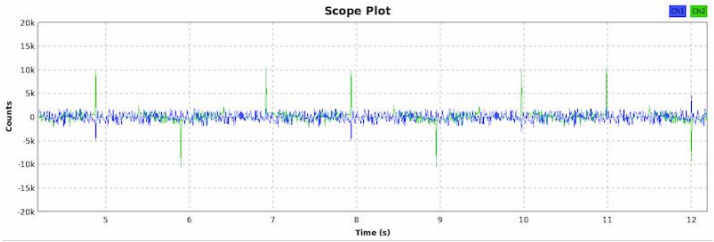

Figure 9 – Data transmission with phase-shift

We can observe that the channel estimation peaks no longer rotate. It is also possible to monitor the evolution of the phase-shif as we can see in Figure 10. Then we can demodulate data while selecting the Q component peaks values and recover the {1,1,-1} sequence.

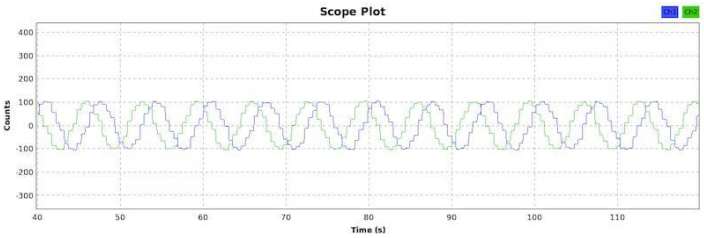

Figure 10 – Phase shift evolution

This system finally acts as the arms of a RAKE receiver [2][3][7].

VI. N210 PLATFORM EXPERIMENTS

A developpement environment has been done in Supélec SCEE lab in order to run communications chains with the latest N210 platform of Ettus Research (as GRC could not support N210 platform at the time this student project was done) in the Microsoft Windows environment. This developpement environment is based on the one hand on processing blocks which have been encapsulated in our C++ framework, and on the other hand on the management architecture for cognitive radio [9] developped in Supélec research [10], called HDCRAM (Hierarchical and Distributed Cognitive Radio Architecture Management) [11].

In this environment, students implemented the sounding chain at a chip rate of 1 Mchip/s. This gives a resolution in time of 1 µs. Above all, this permits to obtain a correlation in 127 µs. Moreover carriers desynchronization has been reduced while using a lower carrier frequency (1.4 GHz) thanks to new RF daughter boards. All that made the phase shift negligible on a correlation period, so that the channel sounding system has been validated in real conditions. Figure 11 shows the I component measured channel impulse response with a resolution of 1 µs @ 1.4 GHz. This measure has been done at a distance of 20 m in two rooms separated by a 12 m corridor.

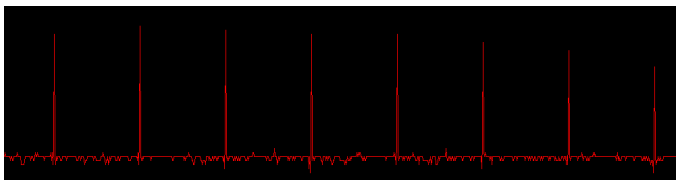

Figure 11 – Measured channel impulse response with a resolution of 1 µs (1 MHz chip rate) @ 1.4 GHz

Thanks to the flexibility of software radio, students could also easily perform many other measurements with different conditions in terms of carrier frequency and chip rate. Next Figures give a few examples.

This system could now be used to make channels measurement campaigns in many different environments, taking into account the office topology, the level of transmission, the carrier frequency, etc. This will be the topic of next year students.

Figure 13 shows a new phenomenon that occurs when carrier frequency goes down (here 300 MHz). The correlation is not only rotating, but also suffers from shape deformation.

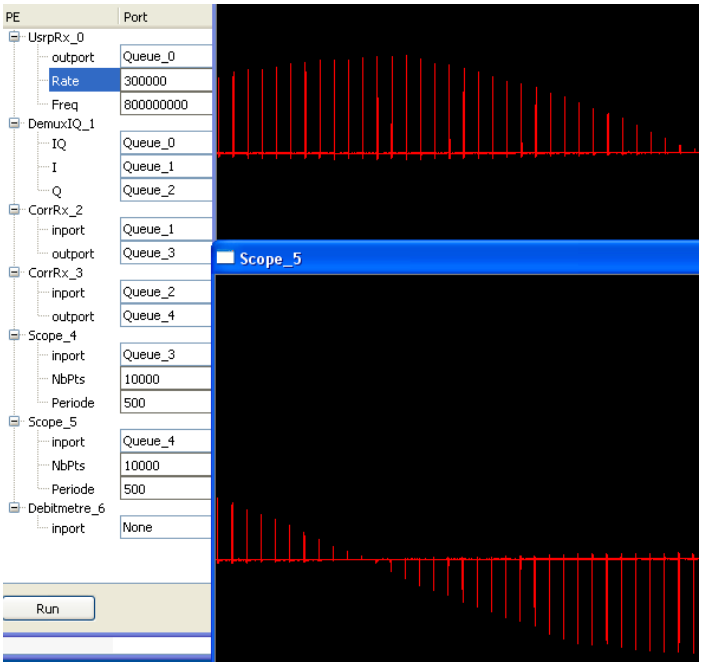

Figure 12 – Measured channel impulse response with a chip rate of 300 kHz @ 800 MHz

Students have not clearly identified the cause of this phenomenon. However, a spectrum analyzer measurement of the radio spectrum also reveals the existence of many frequency harmonics at multiples of the carrier frequency so that the hypothesis of non linearities caused by the amplification may be done. Further investigations are planned to really explain it.

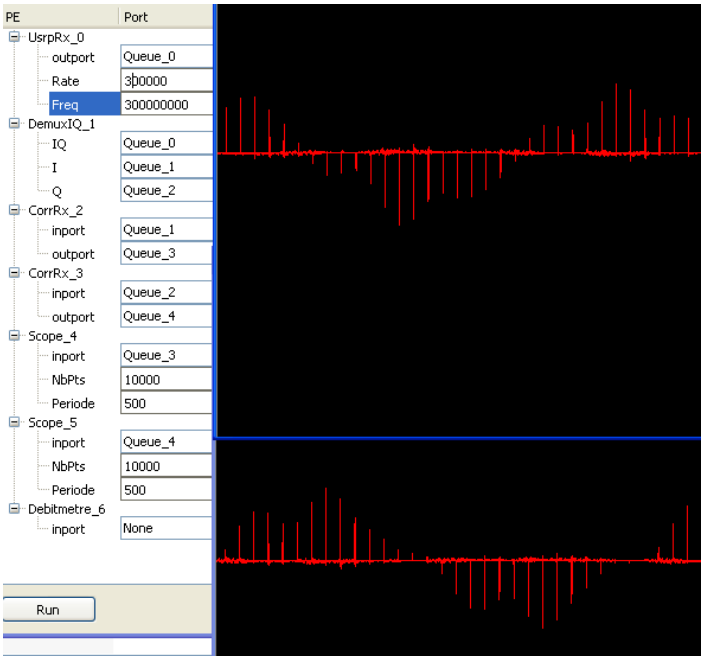

Figure 13 – Measured channel impulse response with a chip rate of 300 kHz @ 300 MHz

VII. Conclusions and Future Work

This paper shows how software radio can be used as a pedagogic tooling for teaching digital communications, beyond theoretical basics. It enables indeed to investigate early in the curriculum of undergraduate students such hard topics as synchronization, multipath, even non linearities, etc. After 2 decades, we can say that software radio promoted the use of open source software solutions for radio design and development. But if this starts to be effective at the education or lab level, this has not yet come true at the industry level. We can wonder when the maturity of such an approach will make it a reality. This paper is also a very good exercise for the writing of a communication paper by students.

This lab and others in the future can be found at:
http://www.rennes.supelec.fr/ren/perso/cmoy/SCEE-SERI/

VIII. Acknowledgment

Authors would like to thank all the contributors to the GNU Radio and Ptolemy projects.

IX. References

[1] J. Mitola, “The Software Radio Architecture,” *IEEE Communications Magazine,* vol. 33, n°5, May 95

[2] C. Moy, "Conception d'un Système de Transmission Numérique à Etalement de Spectre Hybride DS/FH de Type RAKE Adapté au Canal de Diffusion Troposphérique - Mesures de Propagation sur une Liaison Expérimentale à 4.5 GHz", Ph.D. dis., INSA Rennes, June 99

[3] C. Moullec, G. El Zein, J. Citerne, "An integrated all digital diversity receiver for spread spectrum communications over multipath fading channels", ISSSTA’94, 4-6 July 1994, Oulu, Finland.

[4] http://www.ettus.com/products

[5] http://gnuradio.org/redmine/wiki/gnuradio

[6] J.G. Proakis, "Digital Communications", Mc Graw Hill 1995

[7] C. Moy, G. El Zein, J. Citerne, "Performance of Hybrid Spread Spectrum DS/FH RAKE Receivers for Troposcatter Links at 5 GHz", ISSSTA'98, Sun City, South Africa, 2-4 September 1998

[8] G. L. Turin, "Intoduction to Spread Spectrum Antimultipath Techniques and their Application to Urban Digital Radio", *Proc. IEEE*, vol. 68, n°3, March 1980

[9] J. Mitola, “Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio”, Ph.D. dis. Royal Inst. of Tech., Sweden, 2000

[10] J. Palicot, "Radio Engineering: from Software Radio to Cognitive Radio", Wiley ISTE, July 2011

[11] C. Moy, "High-Level Design Approach for the Specification of Cognitive Radio Equipments Management APIs", *Journal of Network and System Management,* vol. 18, number 1, pp. 64-96, Mar. 2010