Proceedings of the 10th WSEAS International Conference on SYSTEMS, Vouliagmeni, Athens, Greece, July 10-12, 2006 (pp20-25)

Reducing Jitter in Nonuniform Sampling Drivers

F. Papenfuß, D. Timmermann
Institute of Applied Microelectronics and Computer Engineering
College of Computer Science and Electrical Engineering, University of Rostock
Richard Wagner Str. 31, 18119 Rostock
GERMANY
frank.papenfuss@uni-rostock.de, http://www-md.e-technik.uni-rostock.de

Abstract: – It is know for quite some time now (cf. [1-3] and [4]) that practical *alias-free signal processing* systems utilizing a wider alias free band than set by the *sampling theorem* can be built successfully through the use of deliberate *nonuniform sampling*. Such systems are especially useful when processing *radio signals* digitally. However, such *sampling systems* usually require a high precision clock generator in order to maintain their full spectral dynamic range throughout their operational bandwidth. Clock generators are referred to as high precision if they expose 3 ps RMS cycle-to-cycle *jitter* or less. Such generators are expensive and not available over all frequency ranges (affordable support typically ranges from 10 to 170 MHz). Therefore, a robust *sampling driver* (SD) design written in VHDL or another portable hardware description language will usually utilise a delay locked loop (DLL) or phase locked loop (PLL) to obtain higher frequencies for internal logic producing the desired nonuniform sampling eventually presented to an attached ADC. Unfortunately every DLL or PLL will add jitter to the produced sampling instants thus diminishing spectral dynamic range. In this paper we propose a remedy to afore mentioned added jitter. The proposed jitter reducing methodology is robust and will work in a variety of nonuniform sampling driver designs. The obtained results are verified, using a sampling driver implementation featuring a FPGA, by *jitter measurements* based on an algorithm reported in [5].

Key-Words: – Nonuniform sampling, alias-free signal processing, sampling theorem, radio signals, sampling system, sampling driver, jitter, jitter measurement

1 Introduction

A sampling driver is a device connected directly to the encode input of an ADC. In contrast to traditional sampling systems applying a fixed frequency clock to the ADC encode input our approach applies a randomised train of sampling pulses to the ADC. Assuming that the SD system clock period T_{clk} is matched to the maximum ADC conversion time the alias-free bandwidth gain is given by

$$G_{BW} = \frac{T_{clk}}{T_q} \tag{1}$$

where T_q is the time quantum realised using a DCDL. It is the least significant delay step (see [6] for details). Earlier publications [4,6] showed efficient architectures to construct a suitable sampling scheme

$$\mathbf{s}\left(\omega^{(s)}, t\right) = \sum_{n=0}^{N-1} \delta\left(t - \mathbf{t}_n\left(t\right)\right) \tag{2}$$

when taking N nonuniform samples. The statistical properties (i. e. PDF of sampling instances and intersample intervals) of a randomised sampling scheme must be chosen carefully to avoid spurious frequencies in the spectrum of randomly sampled signal (see [6] and [7]). The sampling instances $\mathbf{t}_n$ are random variables and are placed on a grid with spacing T_q. The main goal is to generate a suitable sampling point density function (SPDF) $D_s\left(t\right)$. $D_s\left(t\right)$ possesses unit $\left[1/s\right]$ and its magnitude is determined by the probability that time point *t* will become a sampling instance. A suitably constructed random sampling scheme will possess a constant SPDF with a value given by T_q/T_m (where T_m is the mean sampling period). This implies that the sampling process (2) has a constant mean sampling rate, a valid assumption for a SD system with deliberate randomised sampling. For a more complete coverage of nonuniform sampling and the details on how to construct a suitable nonuniform sampling scheme for alias suppression see [3,8-12]. One possible sampling scheme producing a constant SPDF is additive random sampling (ARS) explained in [9,13]. It is implemented by the architecture referred to in this paper.

In an actual implementation of a nonuniform sampling system an ARS scheme will be applied to a RF signal $x_{RF}\left(t\right)$. The only condition that the signal

has to fulfil is that the sum of all the bands in which it is present must not exceed $[0, f_m/2]$. These sub-bands may be distributed arbitrarily within the alias free interval $f \in [0, f_q/2]$ as depicted in Fig. 1b. This is different from the uniform sampling case were the signal's allowed total bandwidth is given by $[0, f_{clk}/2]$. Here, f_{clk} represents the clock frequency used in a uniform sampling system. If $X_{RF}(f)$ is chosen to be present above $f_{clk}/2$ then this is referred to as under-sampling. Bands possessing the same alias inside the base-band have to be avoided, as they cannot be distinguished. Fig. 1a gives an example where band B_3 marked with X cannot be chosen because it would possess the same base-band alias as B_2. Thus, for a fixed frequency f_{clk} there are always a lot of invalid sub-band combinations for $X_{RF}(f)$ which in a world of an ever growing

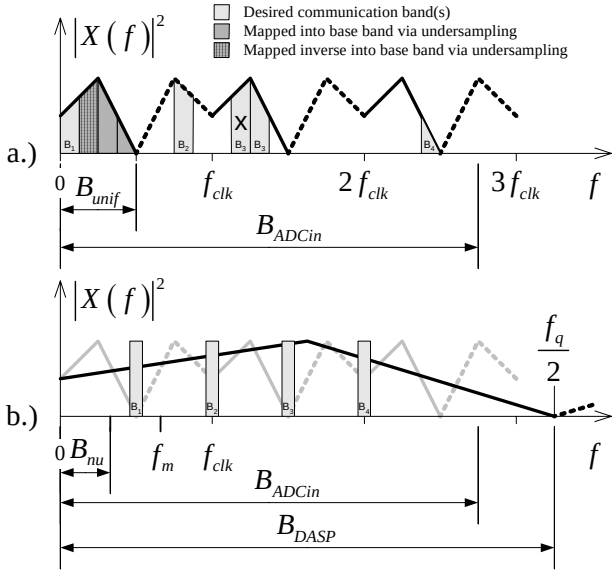

Fig. 1: Placement of sub-bands in a system utilising a.) uniform (under-)sampling and b.) nonuniform additive random sampling (B_{ADCin} - analogue bandwidth of used ADC, B_{unif} - total bandwidth provided by uniform sampling, B_{nu} - total bandwidth provided by nonuniform sampling, B_{DASP} - digital alias-free signal processing (DASP) bandwidth usable by sub-bands of B_{nu}).

demand for bandwidth and demand to dynamically configure the radio front-end of a communication system represents a serious limitation in terms of configurability. Thus, the advantage of using nonuniform sampling as opposed to uniform under-sampling is obvious. In a radio front-end of software defined radio (SDR) systems nonuniform sampling gives more flexibility to select multiple bands in a reconfigurable RF system.

It should be noted that in the case of nonuniform sampling the sub-bands, as indicated in Fig. 1b, when spaced uniformly along the frequency axis can be used to analyse a periodic signal. The harmonic components of which in the case of uniform sampling would alias into base-band B_{unif}. This technique is utilised in sampling scopes to analyse wide band periodic signals (described e. g. in [14])

2 Sampling Instance Creation

The sampling point creation process is implemented using a SD comprising from a control unit, a pseudo random number generator (PRNG) and a digitally controllable delay line (DCDL). The architecture we use is sketched in Fig. 2. The DCDL is the most vital part of the SD since it realises the time quantum step T_q. PRNG and control unit reside in a FPGA but for the DCDL an external specialised chip is used.

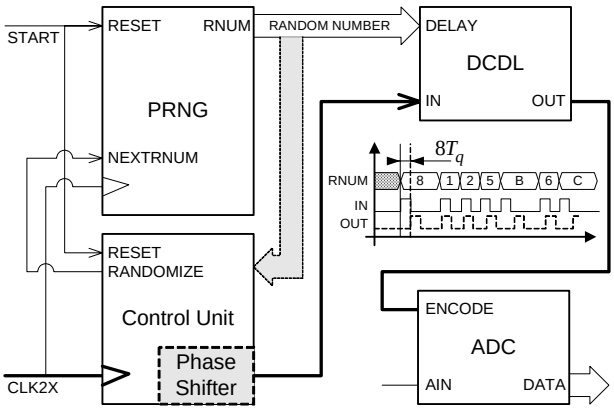

Fig. 2: Fundamental structure of an ARS sampler based on a random number generator coupled with a phase shifter.

It would be beneficial to have the DCDL integrated onto the SD chip but such functionality is currently not available in FPGA chips.

The sampling instant creation algorithm is implemented in the control unit as a finite state machine (FSM). It can be summarised as follows:

1.	If *start* signal is or becomes asserted issue a pulse to the DCDL, instruct PRNG to generate next random number (will occur on next rising clock edge).
2.	If MSB of random number is asserted insert a 180° phase shift (effectively leaving out one clock cycle for the sampling pulse to follow).
3.	Start over with first step.

Table 1: Sampling instance creation algorithm.

Actually, the algorithm in Table 1 is a little more complicated because the attached ADC is a pipelined design and needs maintenance pulses when the SD is stopped. However, this is left out here to keep the focus on nonuniform sampling instance creation.

2.1 Sampling Pulse Jitter

It has been pointed out already that nonuniform sampling is most reasonably applied to direct digital signal processing of radio signals. Our reference design therefore uses a time quantum T_q of 625 ps delivering an alias-free bandwidth of 800 MHz suitable to directly scan a larger portion of the RF spectrum by digital means. The bandwidth of 800 MHz is well tuned to the analogue bandwidth of the used ADC (AD9433) of 750 MHz. The time quantum of 625 ps represents a challenge in terms of chip and PCB design. Hence, the sampling system must be carefully constructed with regard to signal integrity issues. The jitter sensitive path connecting the components involved (i. e. FPGA, DCDL and ADC) is drawn bold in Fig. 2. Actually, the path from the clock generator to the FPGA is involved too but not covered by Fig. 2. We use a low jitter (≤ 1 ps RMS), 100 MHz oscillator to clock the FPGA. But as indicated in Fig. 2 (CLK2X) this clock has to be doubled inside the FPGA to drive the FSM of the sampling driver. Two standard approaches exist to achieve such clock doubling; either through use of a DLL or PLL, respectively. We use a digital clock management (DCM) unit featuring a DLL.

The random numbers along with the sampling pulses coming from the control unit are passed to the DCDL. The DCDL delays the pulses according to the applied random number and eventually drives the ADC encode input. The ideal delay step width of 625 ps and integer multiples of it can practically not be realised due to fluctuations in the DCDL delay steps caused by the chip production process. Therefore, a calibration procedure is vital to find out the true delay step values of a particular DCDL. Unfortunately the delay step values are also temperature dependent. The errors of the sampling instants produced by the SD can thus be partitioned into two categories: First, systematic errors, delay fluctuations due to manufacturing deviations on a per DCDL chip basis. Errors due to temperature drift also fall into this category. Second, random errors, these comprise from random shifts on a per cycle basis stemming from different error sources. This error is usually referred to as cycle-to-cycle jitter. Jitter coming from the clock source and jitter added by the FPGA fall into this category.

In order to quantify both of the above-mentioned errors we developed a calibration algorithm in [5]. This algorithm uses the setup shown in Fig. 3 requiring an additional calibration daughter module. The advantage of such a structure is that the clock which the ADC uses for encoding as well as the test signal are derived from the same clock source via the two paths labelled ❶ and ❷ in Fig. 3, respectively.

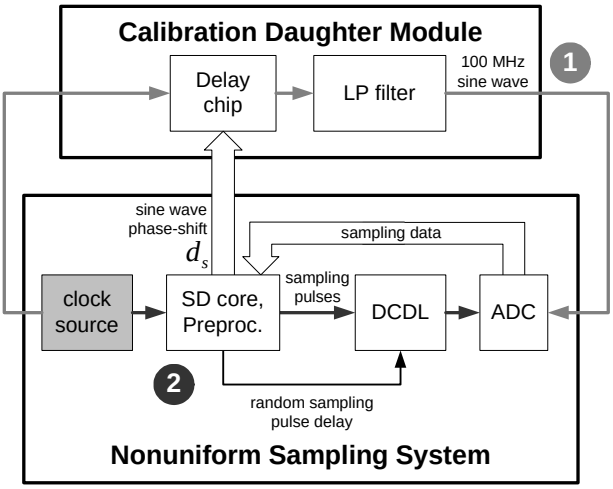

Fig. 3: Sampling driver and calibration module.

Therefore, both signals possess the same medium and long-term phase drift (often referred to as wander) seen over several hundreds or thousands of clock cycles. This is important because it eliminates errors due to phase drift when measuring in picosecond range.

The sampled calibration sine wave is depicted in Fig. 4. The expected graph would be a flat line with some added noise since the SD was working in

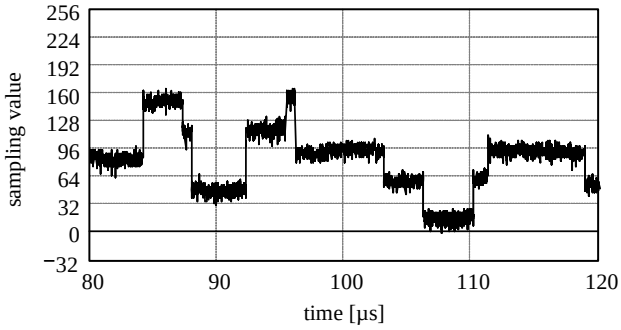

Fig. 4: Excerpt from DCM "corrupted" data, 100 MHz sine wave test signal sampled uniformly at 100 MSPS.

uniform mode when the data was gathered. As is clearly observed, besides the expected noise, there are odd transitions. It looks as if the signal would make sudden jumps. More precisely, it is like it would not be exactly the sampled sine wave but the sine wave plus some added signal. As it turns out the deviations are stemming from the digital clock management unit inside the FPGA. This unit is responsible for creating the FPGA internally phase aligned 200 MHz clock. Using the model of the calibration signal x_c

$$x_c(t) = \hat{a}_c \cos\left(2\pi f_c t + \varphi_c(d_s)\right) + \hat{o}_c \qquad (3)$$

were $\hat{a}_c$ is the amplitude and $\hat{o}_c$ the offset estimate, respectively. The phase of the calibration signal is a

function of the delay value d_c passed to the delay chip of the calibration module. A sampling pulse phase shift between two consecutive sampling pulses is, assuming that d_s remains unchanged, given by

$$\Delta\hat{t} = \frac{1}{2\pi f_c}\left(\arccos\left(\frac{x_2 - \hat{o}_c}{\hat{a}_c}\right) - \arccos\left(\frac{x_1 - \hat{o}_c}{\hat{a}_c}\right)\right). \quad (4)$$

With estimates $\hat{a}_c = 1807$ and $\hat{o}_c = -5.2$ obtained from the delay determination (described in [5]) as well as $x_{\max} = 174$ and $x_{\min} = -4$ we obtain a maximum phase shift of $\Delta\hat{t} = 157$ ps. This is well within the specified value of ± 150 ps in [15, M3, p.32], yet, represents a serious source of error, about 25% of the time quantum we use. Though most of the time the shift will be less than this maximum the whole measurement will be contaminated by this random error source. Equation (4) is used also to gather statistics for the DCDL delay steps using the algorithm described in [5]. In Fig. 5 we present statistics of the DCDL delay when

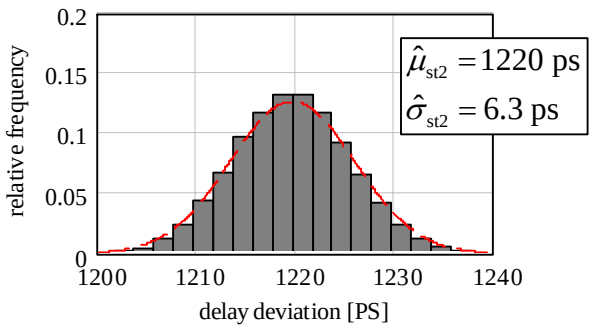

Fig. 5: Statistics of DCDL step delay for tap delay two when DCM jitter is present at the ADC *ENCODE* input. Statistic estimates were obtained from 65536 samples.

digital code two is applied to the *DELAY* tap. The standard deviation of the other delay steps is similar to the one presented here.

3 Sampling Instance Jitter Reduction

Clearly, for any serious measurement the above error source should be eliminated as completely as possible. We therefore propose here a simple but effective circuitry as a remedy to the jitter problem as depicted in Fig. 6.

After the SD core has generated the sampling pulses they should be passed through a “clock gate”. The original high precision clock that entered the FPGA is clocking this „clock gate“. This will only work if the clock signals CLK and CLK2X (see Fig. 6) are aligned. This is true for the used DCM. But there is another problem. The original system clock is only 100 MHz yet the SD clock is 200 MHz and edges may appear or disappear on a 5 ns grid.

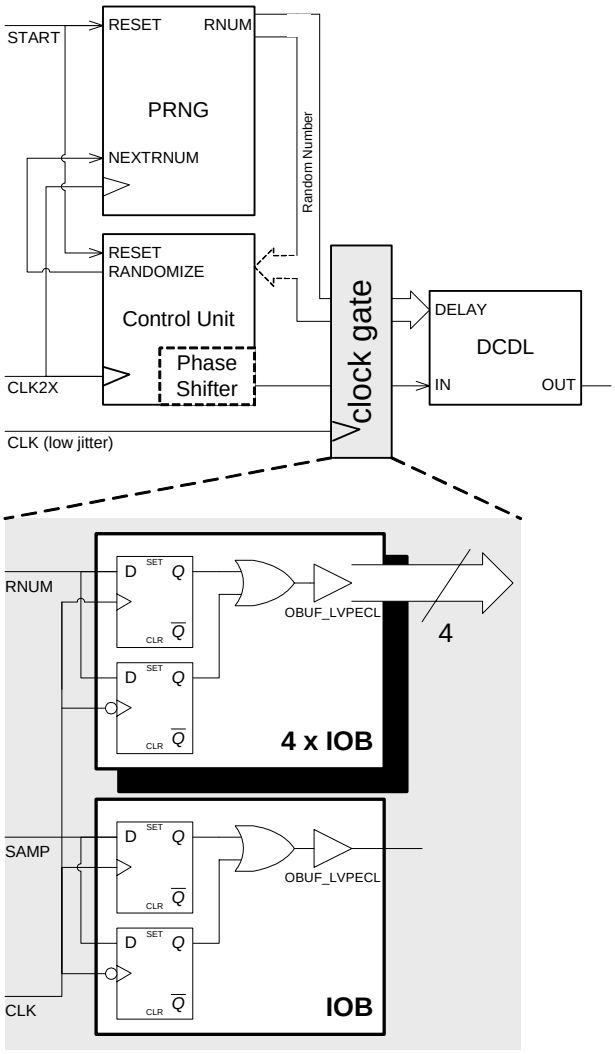

Fig. 6: DDR solution to DCM sampling jitter corruption.

Thus, when clocking the “clock gate” with only 100 MHz sampling pulses will eventually get lost invalidating the whole sampling scheme.

The perturbation present in Fig. 4 can be overcome using double data rate (DDR) flip flops (see e. g. [15, M2, p.3]). These can be instantiated as a black box component and will serve as a clock gate virtually operating at the required rate of 200 MHz. The double data rate components are inferred at the input output blocks (IOB) of the FPGA to ensure optimal alignment of edges between signals leaving the FPGA chip (single bits of the random number as well as the sampling pulses). Fig. 7 shows the 100 MSPS uniform sampled test signal after the described modification was made to the SD design. As is easily observed the corruption by the added DCM jitter has been removed completely. The impact on the statistical estimation is seen in Fig. 8. Given the fact that the precision of the delay estimation is already high with respect to the used