Scalable VHDL Architectures for Non-uniform Sampling Driver Designs

F. Papenfuß, R. Hecht, D. Timmermann

Department of EE and Information Technology, Inst. of Applied Microelectronics and Computer Science, University of Rostock, Germany, E-mail: frank.papenfuss@etechnik.uni-rostock.de

ABSTRACT: Non-uniform sampling promises increased equivalent sampling rates with reduced overall hardware costs. Each device applying non-uniform sampling must use special circuits and architectures to achieve a correct and predictable sampling scheme with respect to time instant placement. The architecture of an ASIC implementation for non-uniform sampling varies with the application. Different boundary conditions for the ASIC design are for instance externally available system clock, required mean sampling rate, targeted equivalent bandwidth, desired sampling memory depth and last but not least the architecture of the used ADC. Taking all these factors into account it is desirable to have a flexible architecture adaptable to different non-uniform sampling applications. The paper presents an approach taken to address these issues using scalable VHDL architectures.

1 Introduction

When designing a device supposed to perform non-uniform sampling this device needs a unit generating the desired sampling pulse sequence enabling DASP[1] for a particular sampling run. The solution presented here will refer to this unit as the SD[2] core. According to DASP theory (cf. [1], [2] and [3]) and applications (cf. [4], [5] and [6]) the SD core produces periodic sampling with jitter. Though every digital circuit driving an ADC does this in principle due to inherent cycle-to-cycle jitter the SD core does it in a more 'controlled' way. First, the time axis can be thought of as being separated into time slots having duration T_s. In each time slot k a sampling instance t_k is produced. The sampling instance t_k (2) is a derived discrete random variable with a uniform distribution having value T_Q/T_s. It can be thought of as being derived from the integer random variable ε_k which has a uniform distribution between zero and M-1, where:

$$M = \frac{T_s}{T_Q}. \tag{1}$$

$$t_{k,m} = kT_s + \varepsilon_{k,m}T_Q \qquad k,m \in Z \tag{2}$$

[1] Digital Alias-free Signal Processing

[2] Sampling Driver

In (2) m is an index into the k-th time slot. This implies that T_s should always be chosen as a multiple of the time quantum T_Q when designing a sampling driver. For all considerations here the members of the set of instances $\{t_k\}$ are assumed to be independent from each other and of equal distribution. Since the SD core can only realize finite time steps the sampling device will generate samples on a fine time grid having resolution T_Q (time quantum).

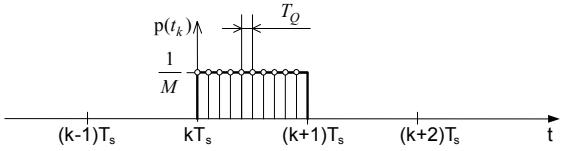

Fig. 1: Pdf of sampling instances at k-th time slot.

It is obvious from the given probability distribution in Fig. 1 that 50% of realized tupels $\{t_{k-1}, t_k\}$ will not satisfy the condition:

$$t_k - t_{k-1} \geq T_s. \tag{3}$$

This, however, violates the time specification of the attached ADC since the clock period of the SD is designed such that it matches the maximum possible conversion rate of the ADC used. In the following Section an architecture will be presented that solves this problem. It should be clear that with a device described in this paper alias suppression is not absolute. Merely the systems bandwidth is widened by the factor T_s/T_Q. From this it immediately follows that it is desirable to minimize T_Q. Minimization of T_Q is a broad topic but out of scope of this paper.

A vital property of the SD core is to produce sampling instances having a uniform pdf at every time slot k. It can be shown that if the distribution of a sampling instance is not chosen to stretch over the whole period T_s then spurious frequencies will occur in the spectrum of the analysed signal. The cause of this effect can be verified by estimating the expectation of the signals spectrum (4) obtained via DFT.

$$E[X_D(f)] = E\left[\sum_{k=-\infty}^{+\infty} x(t_k)\,\mathrm{e}^{-j2\pi f t_k}\right] = \sum_{k=-\infty}^{+\infty} E\left[x(t_k)\,\mathrm{e}^{-j2\pi f t_k}\right] = \sum_{k=-\infty}^{+\infty}\sum_{m=0}^{M-1} x(t_{k,m})\,\mathrm{e}^{-j2\pi f t_{k,m}}\,\mathrm{p}(t_{k,m}) \quad . \tag{4}$$

From (4) it is clear that the signal, sampled with time quantum resolution

$$x_D(t) = \sum_{k=-\infty}^{+\infty}\sum_{m=0}^{M-1} x(t_{k,m}) \qquad k,m \in \mathrm{Z} \tag{5}$$

is multiplied by a (here assumed) periodic signal formed from the sum of distributions of the sampling instances, also referred to as the sampling density function

$$\mathrm{p}_{per.}(t) = \sum_{k=-\infty}^{+\infty} \mathrm{p}(t_k) \qquad k,m \in \mathrm{Z}\,. \tag{6}$$

This effectively means that the expected spectrum is a convolution of the signals spectrum with the spectrum of the sampling density function (7).

$$E[X_D(f)] = \sum_{k=-\infty}^{+\infty}\sum_{m=0}^{M-1} x(t_{k,m})\,\mathrm{p}(t_{k,m})\,\mathrm{e}^{-j2\pi f t_{k,m}} = X_D(f) * P_{per.}(f) \tag{7}$$

From these considerations it is clear that it is desirable to realize a constant sampling density function. This is exactly the case for a set of distributions $\{p(t_k)\}$ depicted in Fig. 1.

2 Non-uniform/Uniform Sampling Unit Architecture

The architecture presented here was designed to be capable of doing both, uniform and non-uniform sampling. In non-uniform mode a PRNG generates random numbers that are used to defer a pulse for a particular amount of time determined by a vector entering the DCDL[3] data tap. Thus the time quantum steps T_Q are realized. In uniform mode a fixed vector is entering the DCDL data tap and SD core generated SAMP enable pulse select samples that are supposed to be taken. Therefore, it was desirable to have a system clock running at the maximum conversion rate of the ADC as this yields maximum performance in uniform mode. The uniform sampling mode can be used to perform interleaved sampling incorporating several SD cores and ADC chips.

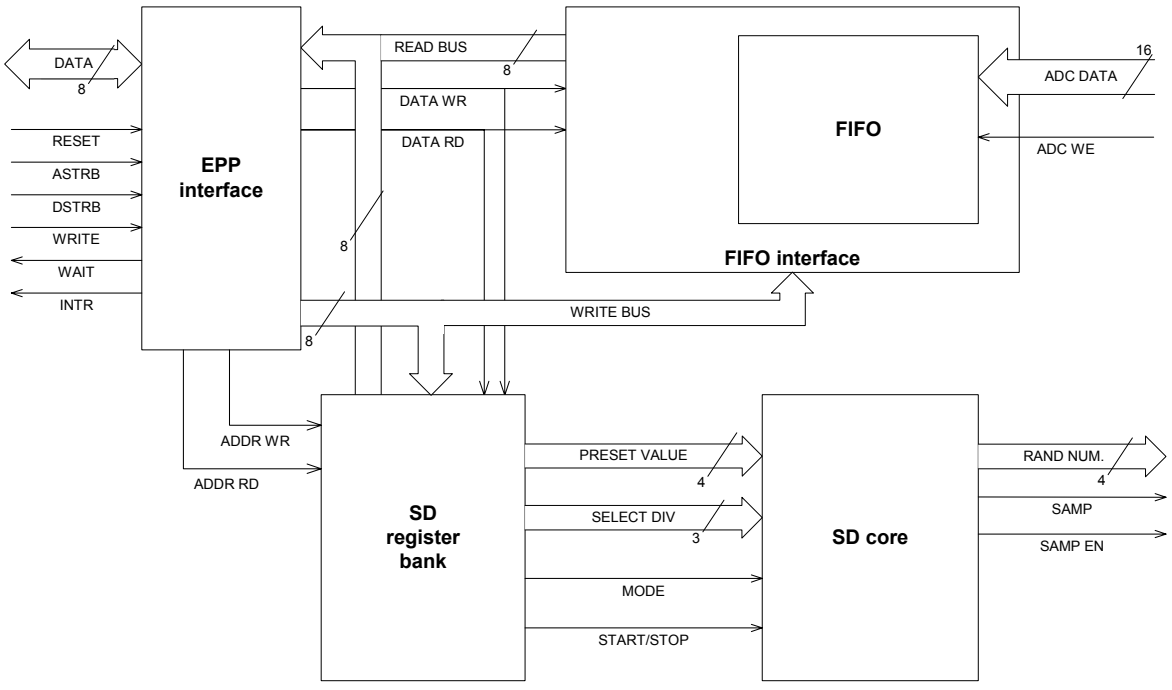

Fig. 2: Overall sampling driver architecture (main units).

[3] Digitally Controllable Delay Line

As mentioned already in Section 1, when realizing a pdf depicted in Fig. 1 there is a 50% chance that realizations of two consecutive sampling instants $\{t_{k-1}, t_k\}$ are too close to match timing restrictions of an attached ADC. However, there is a solution to this problem. By evaluating the PRNG[4] numbers of time slots (k-1) and k, respectively, the SD can determine if such condition will occur. It then skips a pulse for one time slot, which can also be interpreted as a phase shift of 360°. The process of phase shifting distributes the sample realizations belonging to a particular sampling instance differently along the time axis as described by (8). The resulting probability distribution of a sampling instance is depicted in Fig. 4. Note that the sampling instance index k is no longer directly related to the index i of the systems time slots.

$$t_k = t_{k-1} - \varepsilon_{k-1}T_Q + T_s + \varepsilon_k T_Q + \begin{cases} 0 & if \quad \varepsilon_{k-1} < \dfrac{M}{2} \\ T_s & if \quad \varepsilon_{k-1} \geq \dfrac{M}{2} \end{cases} \qquad (8)$$

However, the set of sampling instances $\{t_k\}$ still appear synchronous to the time slots. The sampling distribution algorithm described by (8) implements what is described in [3] as additive random sampling. The resulting sampling density function will be evenly distributed having a constant probability. This property does not depend on the distribution of ε_k (see [3] for proof) when additive random sampling is applied. This implies that the convolution in (7) will not change the shape of the expected spectrum, which is the most important property of our SD solution. The sampling instance probability distribution shown in Fig. 4, when analysed, may still seem to produce tupels $\{t_{k-1}, t_k\}$ not satisfying (3).

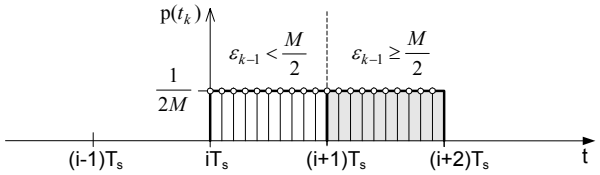

Fig. 4: Modified pdf of k-th sampling instance occurring at i-th time slot.

However, one must consider that the PRNG only generates pseudo random numbers. Two succeeding random numbers of the SD are not truly statistically independent. The random numbers are actually produced by a maximum length LFSR (see [7] and [8]). This means in particular that the following holds for the pseudo random numbers:

$$\varepsilon_k = (2\varepsilon_{k-1} + \tau_k) \bmod 2^n \qquad k \in Z \qquad (9)$$

Where τ_k is a binary random number[5] with even distribution and n is the dimension of the vector passed as random number to the DCDL chip. Since ε_k is an integer in the range of 0 to 2^n-1 it can be shown that pseudo random series of sample pulses will be generated which

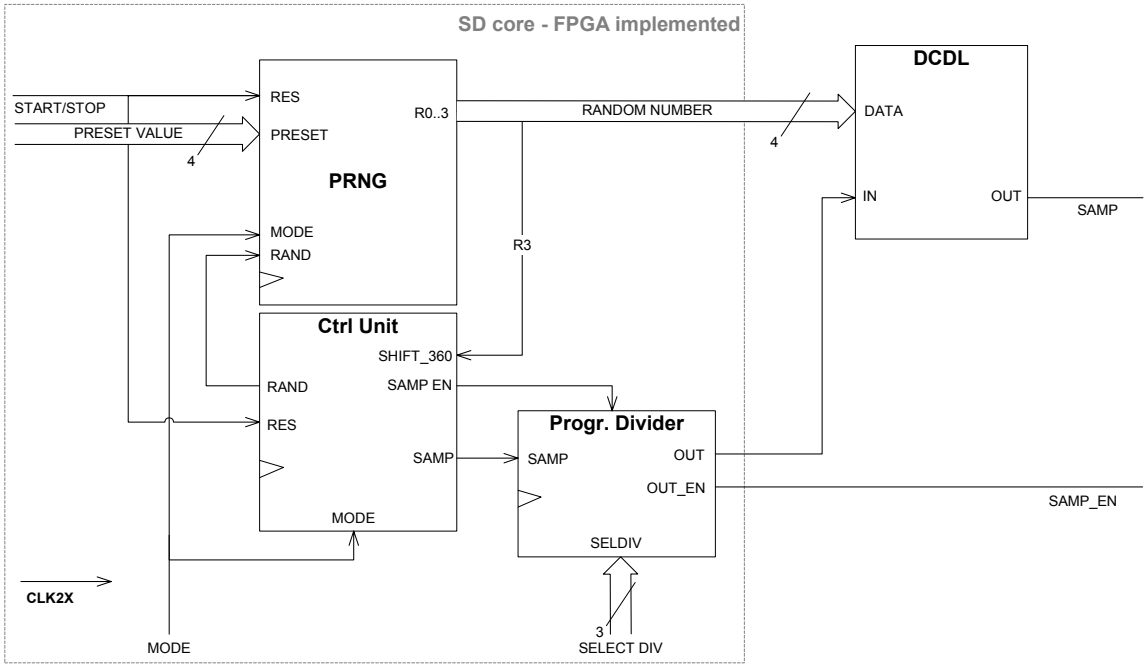

Fig. 3: Sampling driver core architecture.

[4] Pseudo Random Number Generator

never violate (3).

3 Results

A scalable VHDL design has been developed that performs additive random sampling as described in Section 2. The design is scalable because it allows being adapted to different system clocks and different time quantum values. It should be clear from (7) that a modification to the system clock also requires a change to T_Q or M to satisfy (1). The width and depth of the used FIFO can be varied within the bounds of the FPGA internally available RAM capability.

The SD core contained PRNG is easily modified in its length so that no repetitive pattern in the random numbers occurs as long as the FIFO is being filled. When modifying the length of the LFSR care must be taken to construct the correct feedback path for a maximum length LFSR. For details see [8]. The lower four bits of the PRNG are passed to the DCDL realizing M = 16 and thus T_Q = 625ps. The design is easily adapted to different time quantum resolutions M.

The presented solution uses the EPP interface, a well-established PC interface, to communicate with the attached host. The host initiates a sampling run and stops the sampling process after the FIFO has been filled. Steering of the SD core functionality is achieved via the registers contained in the SD register bank (cf. Fig. 2). The host for further processing then reads the data sequence.

The control unit inside the SD core uses a FSM[6] to produce the correct signalling required for DASP. This implies that the externally provided system clock is doubled inside the SD. Clock doubling is achieved using a DLL. For the time being, the XILINX Virtex E FPGA family is the target platform but the design is well suited to be ported to ASIC libraries. The DCDL (MC100EP195) in the current solution is a separate device because such functionality is simply not available in FPGAs today. When moving to an ASIC solution the DCDL could also be included into one chip.

Back-annotation confirms that the XILINX implementation works at an external clock rate of 100MHz (200MHz internal). ASIC libraries are expected to give higher achievable clock rates but such implementations are not required for currently planned applications.

4 Conclusions

The design of sampling driver circuits for non-uniform sampling is a sensitive process. The expected spectrum obtained from a non-uniformly sampled data set is the result of a convolution of the spectrum of the sampling point density function realized by the SD design and the spectrum of the input signal as presented in Section 1. Therefore, it is desirable to realize a constant sampling point density function. A straightforward solution described in Section 1 is only applicable if the time quantum is greater than the minimum conversion period, which can be handled by the ADC. This is almost never the case in DASP applications. Therefore, a different design was developed that actually realizes what is known in the non-uniform sampling theory as additive random sampling. Such sampling after a transient at the beginning of a sampling sequence converges to a constant sampling point density as shown in [3].

The design presented here was carried out in VHDL, making it portable to different FPGA and ASIC libraries. This is mainly interesting should the system clock of 100MHz (200MHz internally) of the current XILINX FPGA realization not suffice for future applications. The design is scalable in terms of included FIFO sampling buffer width, depth and in terms of system clock (i. e. mean sampling rate) and required time quantum (i. e. equivalent bandwidth).

[5] Actually τ_k is generated from the LFSR bits itself.

[6] Finite State Machine

References

[1] D. M. Bland and A. Tarczynski, "The effect of sampling jitter in a digitized signal". In *Proceedings of the 1997 IEEE International Symposium on Circuits and Systems,* Vol. 4, pp. 2685-2688, ISCAS '97.

[2] A. Tarczynski, "FIR filters for systems with input clock jitter". In *The 2001 IEEE International Symposium on Circuits and Systems*, Vol. 2, pp. 617-620, ISCAS 2001.

[3] I. Bilinskis and A. Mikelsons, *Randomized Signal Processing*, Prentice Hall International, 1992, ISBN 0137510748.

[4] Y. Artyukh, I. Bilinskis, M. Greitans, V. Vedin, "Signal Digitizing and Recording in the DASP-Lab System". In *Proceedings of the 1997 Workshop on Sampling Theory and Applications,* pp. 357-360, SampTA '97.

[5] Y. Artyukh, A. Ribakov, V. Vedin, "Evaluation of Pseudorandom Sampling Processes". In *Proceedings of the 1997 Workshop on Sampling Theory and Applications,* pp. 361-363, SampTA '97.

[6] Y. Artyukh, I. Medniks, V. Vedin, "Virtual Oscilloscope of the DASP-Lab System". In *Proceedings of the 1997 Workshop on Sampling Theory and Applications,* pp. 375-378, SampTA '97.

[7] M. Abamovici, M. A. Breuer and A. D. Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, 1990, ISBN 0780310624.

[8] P. Alfke, "Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators", *XILINX XAPP 052,* July 7, 1996 (Version 1.1)