An FPGA-Based Linear All-Digital Phase-Locked Loop

Martin Kumm, Harald Klingbeil, and Peter Zipf

***Abstract*—In this paper, an all-digital phase-locked loop (ADPLL) is presented, implemented on a field programmable gate array (FPGA). All components like the phase detector, oscillator and loop filter are realized as digital, discrete-time components fed from analog-to-digital converters. The phase detection is realized by firstly generating an analytic signal using a compact implementation of the Hilbert transform and secondly computing the instantaneous phase with the CORDIC algorithm. A phase-unwrap component was realized that extends the linear range of the phase detector, so that the linear model is valid in the full frequency range. This property leads to a constant lock-in time for arbitrary frequency changes. An analytic solution for the lock-in frequency range and the stability range including processing delays are given. All relations to design an ADPLL of the presented structure are derived. A detailed example application of an ADPLL designed as offset local-oscillator is given.**

***Index Terms*—Phase-locked loop (PLL), all-digital PLL (ADPLL), field programmable gate arrays (FPGA), direct digital synthesis (DDS).**

EDICS Category: ACS280, DCS190A0, SIPRO110

I. Introduction

As components for digital signal processing are steadily improving, more and more applications of processing signals in the radio frequency (RF) range are moving from the analog to the digital domain. The manifold advantages like the high accuracy, a perfect predictability and the possibility to use more complexity while no calibration is needed are obvious.

The motivation of this work arose from the need of a flexible offset local-oscillator (offset LO) in a fast phase detection system for closed-loop RF controls in the heavy-ion particle accelerator SIS18 at GSI [1], [2]. Since the signal frequencies of that application are in the range of tens of MHz, the processing power of a microcontroller with integrated ADCs and DACs is insufficient while the design of a custom ASIC for quantities of a few hundred is too expensive. This area of application is covered by FPGAs, which play an important role in high energy physics experiments [3] as they offer the speed, density and computational power that are otherwise only achievable with ASICS and are flexible like microcontrollers (in terms or configurability). Therefore, an FPGA with fast ADC and DAC converters was chosen as platform for the offset LO which is based on a discrete-time all-digital PLL (ADPLL).

Manuscript received April 19, 200x; revised January 11, 200y.

M. Kumm and P. Zipf are with the University of Kassel, 34121 Kassel, Germany (e-mail: kumm@uni-kassel.de, zipf@uni-kassel.de).

H. Klingbeil is with with GSI Darmstadt, 64291 Darmstadt, Germany (e-mail: H.Klingbeil@gsi.de)

Copyright (c) 2010 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending an email to pubs-permissions@ieee.org.

ADPLLs have been described and analyzed in several publications, e. g., [4]–[8], but there are only a few reports about FPGA based or discrete-time PLL designs.

One key component in a PLL is the phase detector. Most of the VLSI implementations use one of the following two phase detectors:

1) Phase frequency detectors (PFD) have been widely used, e. g., in [4], [5], [8]–[10]. They produce one or more pulse width modulated (PWM) signals representing the phase difference. The proportional analog voltage can be produced by simply low-pass filtering the PWM signal (which is usually done by the loop filter).
2) Time-to-digital converters (TDC) are common for fully digital PLLs. They are built by a chain of tapped constant delays [4], [10].

These two methods can be combined to profit from the large frequency range of a PFD and the fine resolution of the TDC [10]. Both methods are well understood, but the drawback is, that they are hard to transfer to FPGA logic. The PFD requires a filter to get the value of the phase difference. Due to the unlimited bandwidth of the rectangular shape of a PWM signal, digital filtering is only possible accepting large phase errors. The TDC requires constant delays. The phase resolution highly depends on the accuracy and temperature stability of these delays. Realizing constant delays on FPGAs is only possible at a very low level by first manually defining the placement and routing and measuring the delay of the routed circuit afterwards. Besides this extra effort, the predictability, and the easy portability of the design is lost.

One way to circumvent these difficulties is to combine the advantages from both domains using hybrid PLLs, where the critical parts are realized as analog circuit. In [11], an FPGA based PLL implementation was presented where the loop filter was realized as analog circuit. Likewise, a PLL was presented in [12], where all components were digital and prototyped in an FPGA except the voltage controlled oscillator (VCO). Of course, this is paid by a reduced flexibility as parts of the circuit are fixed.

An ADPLL completely realized on an FPGA was presented in [6] for a digital carrier synchronization. A clocked phase detector was used whose time resolution is in principle limited to the minimum period of the reference clock. While such a concept may be sufficient for digital communications, it does not allow to reach a phase accuracy of a few degrees as a clock frequency of several GHz would be necessary. In [13], a VLSI implementation of a discrete-time ADPLL was described where the CORDIC algorithm [14] was used to perform a combined approximated phase detection and output

signal generation. The drawback of such a combined phase detection and signal generation is the limited range of the phase detector, similar to a sinusoidal phase detector. The range of the phase detector affects the lock-in and pull-out frequencies, as shown later in Section III-C, which limits the usable frequency range in wideband applications. A common way to enhance the frequency range or the lock-in time is to use a dedicated frequency estimation circuit, e. g., [7], [15], which has a separate loop or feed-forward path for the frequency acquisition.

This paper describes a new ADPLL, where all parts are working as discrete-time components running at a fixed sampling rate. All input and output signals are sampled waveforms coming from ADCs or going to DACs. To avoid the discussed phase detection problems, the sampled input signals are first converted into analytic signals using the Hilbert transform which is followed by a Cartesian-to-polar conversion. It is shown that by spending more resources, it is possible to realize a highly linear phase detector with extended detection range. This makes it possible to analyze and simulate the whole PLL using the z-transform, which enormously ease the design process. Apart from the simplified design, the lock-in time becomes independent from the magnitude of a frequency change, and additional circuits for frequency estimation becomes unnecessary.

In the following, two linear PLL models, one for an approximated continuous PLL with reduced complexity and one exact discrete model are derived. Next, the resulting properties like coefficient relations, working range, stability, and the lock-in range are given in Section III. In Section IV, the implementation details are explained. As mentioned before, the ADPLL is used as offset LO from which the main requirements to the presented structure were derived. It is described as example in Section V including the frequency offset generation. Finally, the results from simulations and measurements are presented in Section VI followed by a conclusion and outlook in Section VII.

II. Linear PLL Models

A. The Continuous PLL Model

The presented PLL can be beneficially approximated with a model of a continuous PLL. This simplifies the calculation of some generic PLL parameters. The fundamental equations needed for this work are derived in the following. Further details about PLLs can be found, e. g., in the textbooks [16] or [17]. Fig. 1(a) shows the basic block diagram of a general time continuous (or analog) PLL (APLL). The phase detector block (PD) determines the phase difference of the input signal $v_i(t)$ in relation to the output signal $v_o(t)$ of the VCO. The task of the following controller (which is also called loop filter) $F(s)$ is to minimize the absolute phase difference $\Delta\varphi(t)$ by tuning the frequency of the VCO (f_{VCO}), which is proportional to $v_{VCO}(t)$.

A transfer function can be given for phase signals in the linear range of the phase detector. The term *phase signal* refers to the argument of a sinusoidal signal, i. e., a signal of the form $v(t) = \hat{v}\cos(\omega t + \phi)$ has a phase signal representation

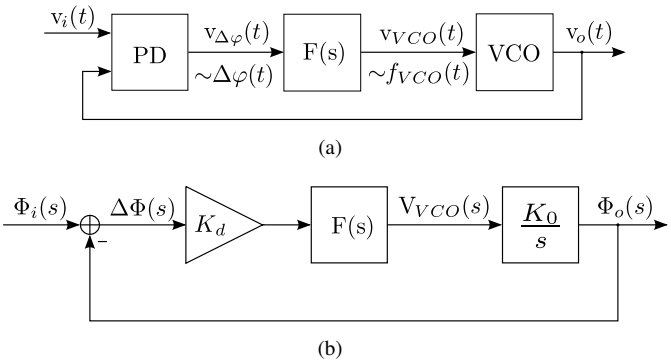


Fig. 1. (a) Block diagram and (b) phase model of the time continuous PLL.

of $\varphi(t) = \omega t + \phi$. It is easy to see that the phase detector can be modeled as subtracter for phase signals. The VCO acts like an integrator as the angular frequency is the differentiated phase with respect to time. Fig. 1(b) shows the block diagram of the resulting phase model, the gain of the phase detector and the VCO are named K_d and K_0 respectively. For this phase model the transfer function of the phase error is

$$E(s) = \frac{\Delta\Phi(s)}{\Phi_i(s)} = \frac{s}{s + K_d K_0 F(s)} \quad . \tag{1}$$

One important PLL property is the type, which is referred to the number of perfect integrators in the loop. It has an effect on the behavior of the steady state phase error [16]. A type-1 PLL, e. g., is able to correct phase differences but has a steady state phase error for frequency differences (which corresponds to phase ramps). The implemented ADPLL has mainly the characteristic of a second order type-2 APLL. It uses a discretized version of the very common proportional integral (PI) controller as loop filter

$$F_{PI}(s) = K_P + \frac{K_I}{s} \tag{2}$$

which is able to correct frequency differences without error. Substituting (2) in (1), the resulting phase error transfer function is

$$\begin{aligned} E_{PI}(s) &= \frac{s^2}{s^2 + K_0 K_d K_P \; s + K_0 K_d K_I} \\ &= \frac{s^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \end{aligned} \tag{3}$$

with the natural radian frequency

$$\omega_n = \sqrt{K_0 K_d K_I} \tag{4}$$

and the damping factor

$$\zeta = \frac{K_P}{2}\sqrt{\frac{K_0 K_d}{K_I}} \; . \tag{5}$$

The trade-off between maximal overshoot and settling time to steady state can be influenced by the damping factor ζ. It is usually set to a value where less overshot is present while a frequency error is quickly corrected leading to a practical range of $0.5 < \zeta < 2$, often a value of $\zeta = 1/\sqrt{2} \approx 0.707$ is used. The natural frequency affects only the settling time and should be high enough to have a fast lock, but must be low enough to benefit from the phase filtering property [16].

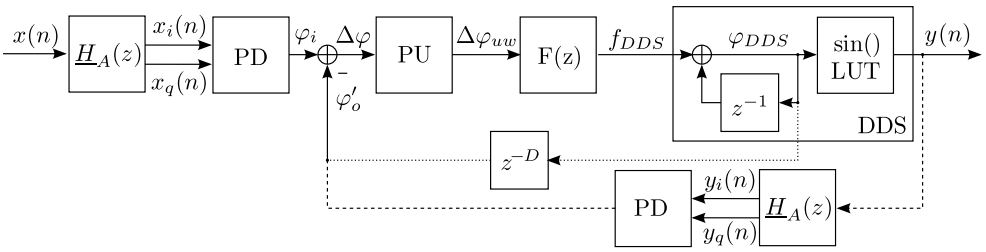


Fig. 2. Block diagram of an ADPLL with analytic Filter $\underline{H}_A(z)$, arc tangent phase detector (PD), phase unwrap (PU), loop filter $F(z)$ and DDS. The components in the dashed path are alternative realizations to the components in the dotted paths as described in the text.

B. The Discrete PLL Model

The block diagram of the proposed ADPLL is shown in Fig. 2. Each component is described in detail in Section IV. In comparison to the analog PLL the loop filter has been replaced by a discrete-time one ($F(z)$), and the VCO has been replaced by a digitally controlled oscillator (DCO), realized as direct digital synthesizer (DDS) [18]. An analytic signal ($\underline{x}(n) = x_i(n) + jx_q(n)$) is generated from the input signal using the discrete Hilbert transform [19], [20], realized as digital filter[1] $\underline{H}_A(z)$. The phase is obtained using a Cartesian-to-polar conversion (PD block). The phase detection range is extended with the phase unwrap block (PU), which will be described in detail in Section IV-D. As the phase of the output signal is available inside the DDS after phase accumulation (φ_{DDS}), no additional phase detector is necessary in principle (dotted line in Fig. 2).

If an absolute phase compensation is required, i. e., the phase difference must be zero and can not be any constant value, then two methods are possible. If the filter $\underline{H}_A(z)$ has a constant group delay, the delays of $\underline{H}_A(z)$ and the DDS can be compensated using a constant delay in the feedback path (z^{-D}). Only if $\underline{H}_A(z)$ has a varying group delay, a phase detection of the output signal using the same filter is necessary (dashed line in Fig. 2).

The corresponding phase model is shown in Fig. 3. The DDS is described by the discrete transfer function of the phase accumulator with gain k_0. Furthermore, the processing delays (e. g., for pipelining) are included as D_{FF} for the feed forward path and D_{FB} for the feedback path. If $\underline{H}_A(z)$ has a non-constant group delay, it can be approximated with D_{FB} for small delay variations. For worst-case stability examinations, the maximal group delay has to be used. The gain of the discrete phase detector is named k_d. The phase error transfer function of this model is

$$E(z) = \frac{1 - z^{-1}}{1 - z^{-1} + k_d k_0 \; F(z) \; z^{-(D_{FB}+D_{FF})}} \quad . \tag{6}$$

C. Choosing the Loop Filter

As we have a fully digital system, the error transfer function can be set to any desired transfer function $E_d(z)$ that is realizable (i. e., it must be causal). The corresponding loop

[1]The term $\underline{H}_A(z)$ is used to distinguish the analytic signal generation filter from the pure Hilbert transform $H_H(z)$. They are related with $\underline{H}_A(z) = 1 + jH_H(z)$.

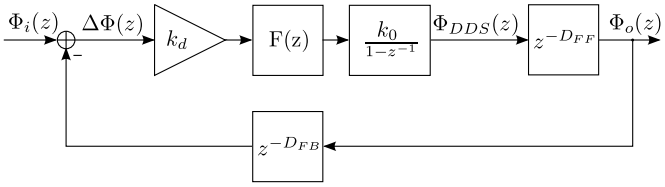


Fig. 3. Phase model of an ADPLL.

filter transfer function can then be obtained by solving (6) for $F(z)$ and setting $E(z)$ to $E_d(z)$. The properties of $E_d(z)$ have to be chosen carefully. For most applications, the phase error should tend to zero with increasing time leading to the condition

$$\lim_{t\to\infty} \Delta\varphi(t) = \lim_{z\to 1}(z-1)E(z) = 0 \quad . \tag{7}$$

For applications where a minimum settling time is desired [12], a finite impulse response, i. e., dead beat behavior can be obtained by setting $E_d(z)$ to a pure delay transfer function, e. g., $z^{-(D_{FB}+D_{FF})}$. All poles in $E_d(z)$ are then canceled by zeros. Other applications make use of the phase filtering property [16] of a PLL. In this case $E_d(z)$ has to be chosen such that the input to output transfer function $H(z) = \Phi_o(z)/\Phi_i(z)$ has a low-pass behavior.

We conclude, that these partially contradicting properties are highly application dependent. The ADPLL described in the following uses a PI controller for $F(z)$ resulting in a good phase error behavior of $E(z)$ and a low-pass behavior for $H(z)$.

The transfer function of the DDS corresponds to the discrete approximation of an analog integrator after applying the backward difference formula [21]. The same method is used for discretizing the continuous PI controller[2]. The discretization is done by substituting

$$s \leftarrow \frac{1}{T_s}(1 - z^{-1}) \tag{8}$$

in (2) resulting in the IIR filter

$$F_{PI}(z) = \frac{b_0 + b_1 z^{-1}}{1 - z^{-1}} \tag{9}$$

[2]Note that the bilinear transform is commonly more accurate in discretizing a continuous transfer function but is also more complex, often leading to more expensive hardware structures. In the current case the arising inaccuracy is negligible as the natural frequency is several orders of magnitudes smaller then the sampling rate.

with the coefficients

$$b_0 = K_P + K_I T_s \tag{10}$$

$$b_1 = -K_P \tag{11}$$

and $T_s = 1/f_s$ as sample period.

For the special case when $D_{FF} = D_{FB} = 0$ and $k_0 k_d = K_0 K_d T_s$, the ADPLL transfer function (6) with PI controller as loop filter is equal to the result of applying the rectangular method (8) to the analog PLL transfer function (3). In this case, the behavior of the ADPLL is approximately equal to a second order analog PLL. For large sampling rates with respect to the phase signal frequencies, this approximation holds also for delays greater than zero as explained in Section III-A.

D. Calculation of the Controller Coefficients

Using the analog reference model we have a very descriptive method for determining the coefficients of the discrete controller by specifying the damping factor ζ and the natural radian frequency $\omega_n = 2\pi f_n$. The relation between continuous and discrete parameters can be obtained by substituting (4) and (5) in (10) and (11) resulting in

$$b_0 = (2\zeta + \omega_n T_s)\omega_n / (K_0 K_d) \tag{12}$$

$$b_1 = -2\zeta\omega_n / (K_0 K_d) \quad . \tag{13}$$

III. Properties of the ADPLL

A. Working Range

The approximation quality of the discrete transfer function (6) with PI controller (9) to the second order system (3) is a function of the overall delay $D = D_{FB} + D_{FF}$, the sampling rate f_s and the controller parameters. It can be seen from Fig. 3 that moving the delay from D_{FF} to D_{FB} and vice versa while keeping D constant has no effect to the system dynamic – only the output is shifted in time. In principle, it is hard to derive an analytic solution for the approximation quality or the stability range for a varying order of the transfer function.

A detailed analysis of type-2 ADPLLs with feedback delays can be found in a recent publication of Wilson et al. [22]. In this work, the two dominant poles of the discrete transfer function are considered, whereas a criteria is given for which parameters this assumption is valid. They use a slightly different loop filter but the results can be easily transferred to the proposed one.

Unfortunately, no values of the approximation error were given. Therefore, additional time domain simulations were performed for a practical order range of $D = 1...100$. The approximation error was defined to the maximum of the absolute value of the difference between the step responses of the discrete and continuous models $E(z)$ and $E(s)$, normalized to the phase step $\Delta\phi$:

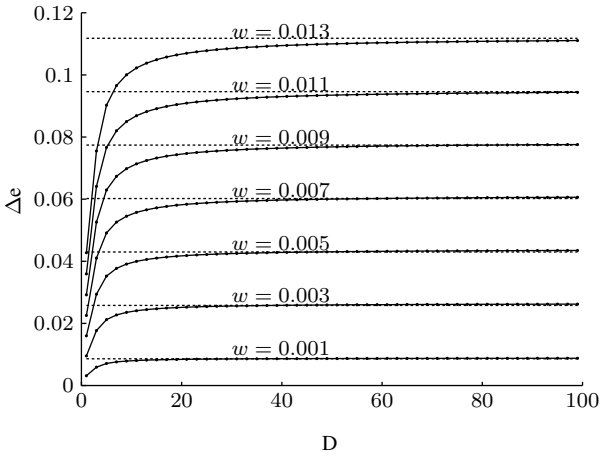
$$\Delta e = max\left|\left(e_{step,disc}(n) - e_{step,cont}(t = nT_s)\right)/\Delta\phi\right| \tag{14}$$

The simulation was performed until $t = 4\frac{1}{f_n}$, where the system can be assumed to be in steady-state.

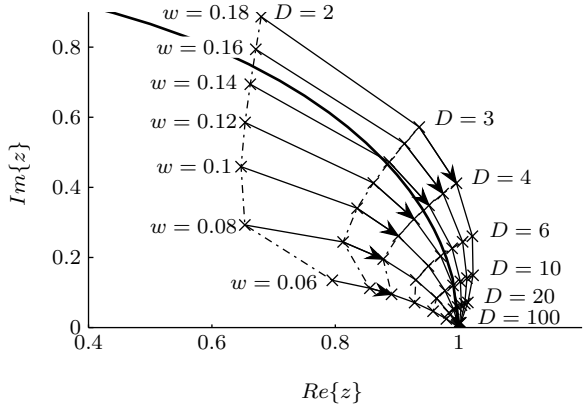
It was figured out that the approximation quality is approximately inverse proportional to the normalized natural frequency f_n/f_s for large D. Hence, the natural frequency is linearly adjusted to a working point w according to $f_n = wf_s/D$. Fig. 4(a) shows the result of simulations for different working points with $\zeta = 0.707$. It can be seen that the error is bounded for the given parameter range to

$$\Delta e_{max} \leq 8.6w = 8.6 f_n / f_s D \tag{15}$$

which is shown a dotted line in Fig. 4(a).



(a)



(b)

Fig. 4. (a) Approximation error vs. PLL order D for different working points (b) trajectories of the dominant pole on the z-plane for different working points and increasing order

B. Stability

A discrete system is called to be stable if all poles are located inside the unit circle. Therefore, the pole locations on the z-plane were numerically evaluated for different working points and orders. The trajectories of the pole with the largest magnitude are plotted in Fig. 4(b), which shows the upper right part of the unit circle (thick curve). The pole locations are marked with a "×", all pole locations that correspond to the same working point w are connected with straight lines, poles corresponding to the same order D are connected with dashed lines. The arrows indicate the direction to higher orders. It can be observed than this pole approaches 1 for large D, whereas the trajectory may cross the unit circle. All trajectories for

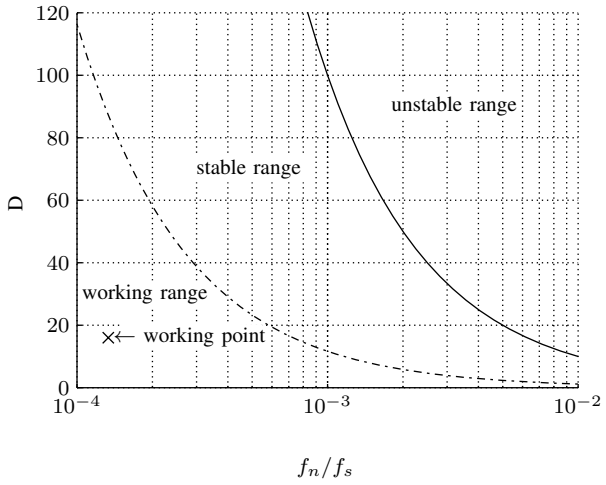


Fig. 5. Stability and working range of the ADPLL with $\zeta = 0.707$ and $D \leq 1000$. The working point was used in the design explained in Section V.

$w \leq 0.1$ do not cross the unit circle up to at least $D = 1000$, which seems to be a criteria that includes nearly all practical ADPLL applications.

The stability range, as well as the practical working range for which the approximation error is less than 10% are summarized in Fig. III-B. As long as we stay in the working range the analog model can be used to describe the discrete ADPLL. In the following sections this assumption is made to derive further properties of the ADPLL.

C. Analytical Determination of the Lock-In Range

One important property of a PLL is the frequency working range. There are several frequency limits that are defined by different behaviors, the most important are briefly explained in the following according to [16]. The initial output frequency f_o of the VCO/DDS is usually set to the center of the frequency working range f_0. The lock-in range Δf_L is defined as the maximum input frequency deviation from the initial output frequency where the PLL locks without any cycle slipping after closing the loop. Beyond this range there is the so called pull-in range Δf_{PI} over which the PLL is able to lock, but cycle slipping may occur. This happens when the linear range of the PLL is left. As an illustration, Fig. 11(a) and Fig. 11(b) show a simulation of the frequency step responses of the phase error and the DDS frequency, respectively, for different frequency steps. As soon as the frequency jump is larger than the lock-in range ($\Delta f_L = 110.2$ kHz in this example) phase slips occur which delays the lock-in time.

Mathematically, the lock-in range Δf_L can be derived from its definition that the phase difference must always be less than the maximal detectable phase $\varphi_{det,max}$ of the phase detector

$$|\Delta\varphi(t)| \leq \varphi_{det,max} \ \forall \ \ t \tag{16}$$

for an input phase signal of $\varphi_i(t) = \omega t$ with

$$\omega_0 - \Delta\omega_L \leq \omega \leq \omega_0 + \Delta\omega_L \quad . \tag{17}$$

At frequency $\omega = \omega_0 + \omega_L$, this leads to the formal condition

$$max|\Delta\varphi(t)| := \varphi_{det,max} \tag{18}$$

with

$$\Delta\varphi(t) = \Delta\omega_L \ t \circledast e(t) + \Delta\varphi_{0,max} \tag{19}$$

where $e(t)$ is the impulse response of the error transfer function, $\circledast$ is the convolution operator, $\Delta\varphi_{0,max}$ is the maximal initial phase offset and $\Delta\omega_L = 2\pi\Delta f_L$. For the PI controller, the phase difference results in

$$\begin{aligned}\Delta\varphi(t) &= \mathcal{L}^{-1}\{\Delta\omega_L \frac{1}{s^2} E_{PI}(s)\} + \Delta\varphi_{0,max} \\ &= \frac{\Delta\omega_L}{\omega_r} \ e^{-\zeta\omega_n t} \ \sin(\omega_r t) + \Delta\varphi_{0,max}\end{aligned} \tag{20}$$

with the resonant frequency $\omega_r = \omega_n\sqrt{1-\zeta^2}$ [23]. It describes the well-known damped oscillation for $\zeta > 0$. The extreme values can be found for $t = (k\pi + \alpha)/\omega_r$, $k \in \mathbb{Z}_0$ [24] with

$$\alpha = \arctan(\sqrt{1-\zeta^2}/\zeta) \quad . \tag{21}$$

For the damped oscillation the first extreme value ($k = 0$) corresponds to the maximum value for positive $\Delta\omega_L$. Setting this result in (18) and (20) results in

$$\Delta\omega_L = (\varphi_{det,max} - \Delta\varphi_{0,max}) \ \omega_n \ f(\zeta) \tag{22}$$

with

$$f(\zeta) = \frac{\sqrt{1-\zeta^2}}{\sin\alpha} \ e^{\alpha\zeta/\sqrt{1-\zeta^2}} \ . \tag{23}$$

The maximal initial phase offset $\Delta\varphi_{0,max}$ that can occur is obviously π. Note that for an ordinary phase detector with a maximal detectable phase of $\varphi_{det,max} \leq \pi$, the lock-in range is zero—one initial phase slip can always occur. The dependency of ζ is nearly linear for $0.5 < \zeta < 2$ and can be approximated to

$$f(\zeta) \approx 1.8\,(\zeta + 0.5) \tag{24}$$

with an error of less than 1.7%, leading to

$$\omega_L \approx (\varphi_{det,max} - \Delta\varphi_{0,max}) 1.8\omega_n\,(\zeta + 0.5) \tag{25}$$

for $0.5 < \zeta < 2$.

For calculating the pull-out frequency $\Delta\varphi_{0,max}$ can be set to zero, as the PLL is already in locked state when the frequency change occurs. Note that this result is different from Gardners's [16] empirical relation for the pull-out frequency of a PLL with a sinusoidal phase detector (which was detailed in [25]):

$$\omega_{PO} \approx 1.8\omega_n\,(\zeta + 1) \tag{26}$$

It differs by a factor of $\varphi_{det,max}$ and an offset.

IV. Implementation

A. Phase Detection

As already mentioned in Section II-B, the phase detection consists of a Hilbert transform followed by a Cartesian-to-polar conversion, shown as blocks $\underline{H}_A(z)$ and PD in Fig. 2.

The filter $\underline{H}_A(z)$ splits the real valued input signal into the so-called in-phase $x_i(n)$ and quadrature component $x_q(n)$. The in-phase $x_i(n)$ is a delayed version of the input signal and $x_q(n)$ corresponds to the Hilbert transform of $x_i(n)$, which has a phase difference of 90° in respect to $x_i(n)$. These

two signals can be interpreted as Cartesian complex (analytic) signal $\underline{x}(n) = x_i(n) + jx_q(n)$. The corresponding amplitude and phase can be obtained by converting this complex signal into its polar form $\underline{x}(n) = |\underline{x}|\, e^{\varphi(n)}$. The phase $\varphi(n)$ is equal to the delayed phase of the input signal. Note that the input signal must be perfectly sinusoidal in theory for this type of phase detector. It has been shown in practice that a small bandwidth is sufficient and that the whole PLL is robust even if the signal has a significant part of harmonics like a rectangular waveform. Nevertheless, harmonics can disturb the phase accuracy, depending on the high frequency behavior of the loop filter.

The realization of the Hilbert Transform is described in the next section. The Cartesian-to-polar conversion is implemented using the CORDIC algorithm, which is described in Section IV-C.

B. Hilbert Transform

Different architectures for approximating the Hilbert transform were already proposed in [20]. These filters directly produce an approximation of the analytic representation of the input signal $x(n)$ in the form

$$\underline{y}(n) \approx x(n) + j\mathcal{H}\{x(n)\} \tag{27}$$

where $\mathcal{H}\{\cdot\}$ denotes the Hilbert transform.

One important property of an analytic signal is that no signal components for negative frequencies exist. The idea in [20] was to extend a class of multiplier-less frequency sampling filters (FSF) described in [26] to filters with complex coefficients. These FSFs can be seen as generalization of the popular cascaded integrator comb (CIC) filter.

One of the transfer functions presented in [20] was further optimized and is composed of very simple subfilters that can be realized without any multiplier:

$$\underline{H}_A(z) = \left(\frac{1}{4} Z_{-90}(z) Z_{-30/-150}(z) P(z)\right)^O \tag{28}$$

with

$$Z_{-90}(z) = 1 - jz^{-1} \quad , \tag{29}$$

$$Z_{-30/-150}(z) = 1 + jz^{-1} - z^{-2} \tag{30}$$

and

$$P(z) = \frac{z^{-2}}{1 - 0.5z^{-2}} \quad . \tag{31}$$

The corresponding frequency response is shown in Fig. 6 for different O. The operation of the filter can be best explained by looking at the poles and zeros of $\underline{H}_A(z)$ in the z-plane, which is included in Fig. 6. Subfilters $Z_{-90}(z)$ and $Z_{-30/-150}(z)$ have zeros at $-90°$ and $-30°/-150°$, respectively, which are responsible for the damping of negative frequency components. The IIR subfilter $P(z)$ produces poles on the real axis at $\pm 1/\sqrt{2}$ which sharpens the transition band. Each subfilter is used multiple times (parameter O) to enhance the stopband attenuation.

For one stage ($O = 1$), the passband ripple ranges from $-0.0257...0$ dB and the stopband attenuation is better than 22.2 dB, both in the frequency range $|0.065 f_s ... 0.435 f_s|$. The resulting passband ripple and the stopband attenuation of the total filter can be calculated by multiplying these logarithmic values with the number of stages O. The frequency range is slightly getting narrower with each stage.

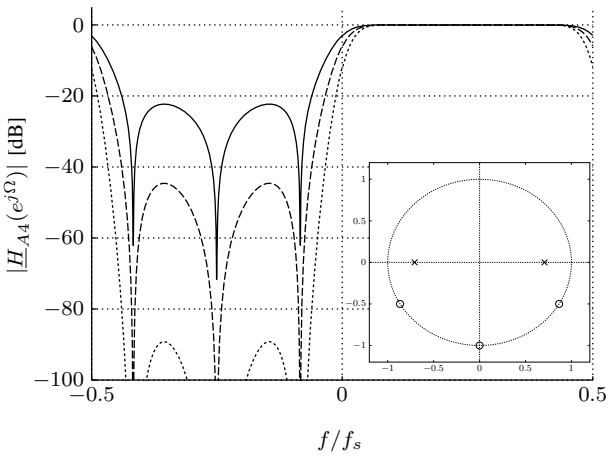


Fig. 6. Frequency response of the complex frequency sampling filter $\underline{H}_A(z)$ for $O = 1$ (solid line), $O = 2$ (dashed line) and $O = 4$ (dotted line).

The implementation of the complex subfilters is quite simple. A multiplication of a complex number by j can be effectively implemented in hardware by negating the imaginary part and swapping real and imaginary part afterwards:

$$(a + jb) \cdot j = -b + ja \tag{32}$$

To give an impression about the low hardware complexity Fig. 7 shows a block schematic of one stage of $\underline{H}_A(z)$. Note that the gain blocks of $1/2$ can be realized as binary shift operation.

C. CORDIC Algorithm

The phase of the complex signal $\underline{x} = x_i + jx_q$ can easily be detected by the trigonometric relation $\varphi = \arctan(x_q/x_i) \pm \pi$. This relation can be effectively implemented in FPGA logic using the CORDIC algorithm, which was originally presented by Volder in 1959 [14]. The implementation of Guntoro [27] has been used, which is a pipelined 16 stage CORDIC implementation with preprocessing to cover all quadrants of the input vector and a post processing which scales the output to an M-bit binary coding of

$$\varphi = \left(-a_{M-1} + \sum_{i=0}^{M-2} a_i 2^{i-M+1}\right)\pi \tag{33}$$

with $a_i \in \{0, 1\}$, leading to the range $-\pi ... \pi - \pi/2^{M-1}$, which has several advantages as we will see later. The total phase accuracy is better than $\pm 0.05°$ with a total latency of 18 clock cycles.

D. Extending the Lock-In Range

It can be seen from (22) that the lock-in range can be increased by either increasing the natural frequency, the damping or the maximal detectable phase of the phase detector. As the natural frequency is limited by the PLL stability and a

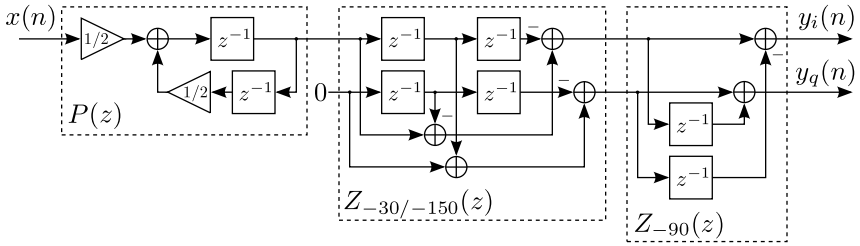


Fig. 7. Implementation of $\underline{H}_A(z)$, each subfilter is implemented O times. The real filter $P(z)$ is implemented first as the imaginary part of the input signal is zero. The lowest adder of the first $Z_{-30/-150}(z)$ can be omitted due to the missing imaginary input but is shown for convenience.

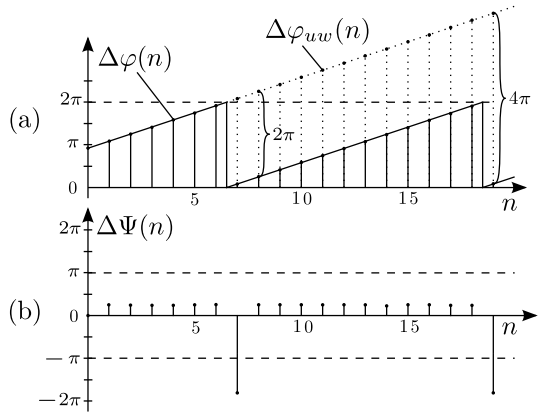


Fig. 8. Example timing plot of (a) the phase error sequence $\Delta\varphi(n)$ (solid line) and the unwrapped phase error $\Delta\varphi_{uw}(n)$ (dotted line), and (b) the difference of two samples $\Delta\Psi(n)$ for a positive frequency difference.

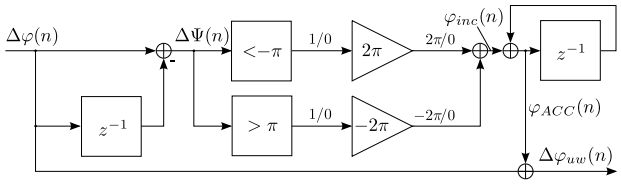


Fig. 9. Block diagram of the phase unwrap method.

larger damping leads to a longer lock-in time, a simple phase unwrap method for extending the range of the phase detector is explained in the following.

The output of a discrete four quadrant arc tangent phase detector with two input signals having a frequency difference of Δf and an initial phase difference of $\Delta\varphi_0$ is

$$\Delta\varphi(n) = (2\pi\Delta f T_s n + \Delta\varphi_0) \mod 2\pi \quad . \tag{34}$$

A phase wrap from 2π to zero for positive Δf appears in each period as shown exemplary as sequence corresponding to the solid line in Fig. 8(a). For negative Δf, phase wrap from zero to 2π appears. Let us now look at the difference of two sequent samples of the phase difference

$$\Delta\Psi(n) = \Delta\varphi(n) - \Delta\varphi(n-1) \tag{35}$$

which is shown in Fig. 8(b). Due to the modulus operation in each phase difference the result of (35) can be classified in different cases:

$$\Delta\Psi(n) = \begin{cases} 2\pi\Delta f/f_s & \text{for case I or II} \\ 2\pi\Delta f/f_s - 2\pi & \text{for case III} \\ 2\pi\Delta f/f_s + 2\pi & \text{for case IV} \end{cases} \tag{36}$$

which are:

case I: $\Delta f \geq 0 \wedge \Delta\varphi(n) > \Delta\varphi(n-1)$
case II: $\Delta f \leq 0 \wedge \Delta\varphi(n) < \Delta\varphi(n-1)$
case III: $\Delta f > 0 \wedge \Delta\varphi(n) < \Delta\varphi(n-1)$
case IV: $\Delta f < 0 \wedge \Delta\varphi(n) > \Delta\varphi(n-1)$.

In cases I and II no phase wrap occurs as the slope of $\Delta\Psi(n)$ has the same sign as Δf. In cases III and IV, an undesired phase wrap occurs that has to be eliminated. For the example signal in Fig. 8(b), most of the time the samples of $\Delta\Psi(n)$ correspond to case I (regular positive frequency difference) while the samples at $n = 7$ and $n = 19$ correspond to case III (phase wrap occurred).

By now, we don't have an exact criterion for detecting the phase wrap as we don't know the frequency difference Δf. Therefore, let us look on the possible frequency ranges. The frequencies of both real input signals are limited by the Nyquist frequency $f_s/2$ leading to a frequency difference lying in the interval $\Delta f \in (-f_s/2, f_s/2)$. Setting this interval in (36) results in

$$\Delta\Psi(n) \in \begin{cases} (-\pi, \pi) & \text{for case I or II} \\ (-3\pi, -\pi] & \text{for case III} \\ [\pi, 3\pi) & \text{for case IV} \end{cases} \quad . \tag{37}$$

Now, we have a unique characteristic that can be used to detect the phase wrap by checking if $\Delta\Psi(n)$ is less than $-\pi$ (case III) or greater than π (case IV). The compensation can be realized by adding 2π (case III) or subtracting 2π (case IV) according to (36). This compensation value is accumulated ($\Delta\varphi_{ACC}$) to be able to compensate several periods and then added to the input phase difference resulting in the unwrapped output signal $\Delta\varphi_{uw}$ as shown in Fig. 8(a). A block schematic of the principle is shown in Fig. 9.

The coding of the phase detector output (33) allows, that only the two most significant bit (MSB) of $\Delta\Psi(n)$ must be evaluated to detect values greater than π or less than $-\pi$. This can be realized as 2 bit look-up table (LUT) whose content is listed in TABLE I. The LUT result is accumulated and added to the input phase. Therefore, the word size of the output phase must be extended to allow values greater than $\pm\pi$. With the used coding, adding or subtracting 2π influences only the word extension with the consequence that the output of the accumulator can be concatenated to the MSB of the input phase, which saves the lower adder in Fig. 9.

TABLE I
TRUTH TABLE OF PHASE-UNWRAP LUT.

$\Delta\Psi$		φ_{inc}		Case	Range of $\Delta\Psi$
MSB	MSB-1	1	0		
0	0	0	0	I	$0 \leq \Delta\Psi \leq \pi$
0	1	1	1	IV	$\pi \leq \Delta\Psi < 2\pi$
1	0	0	1	III	$-2\pi < \Delta\Psi \leq -\pi$
1	1	0	0	II	$-\pi \leq \Delta\Psi \leq 0$

A word size extension of P bit results in an extended maximal detectable phase of $\varphi_{det,max} = 2^P\pi$. The word size extension parameter P can be determined by setting this result in (22) using $\Delta\varphi_{0,max} = \pi$ as maximal initial phase, leading to

$$P = \left\lceil log_2 \left(\frac{\Delta\omega_L}{\pi\omega_n f(\gamma)} + 1 \right) \right\rceil \quad . \tag{38}$$

Due to the low hardware costs it is appropriate to set the lock-in frequency to the Nyquist rate ($\Delta f_L = f_s/2$) to ensure that the linear behavior is valid for any possible frequency jump.

The unwrap method does not influence the transfer function of the ADPLL, it enhances the linear range in which the model is valid. To illustrate the usefulness of the unwrap method, frequency step responses were simulated which are shown in Fig. 11. The parameters of the PLL designed in Section V were used. First, the simulation was done without phase unwrap ($P = 0$) starting from an initial phase difference $\Delta\varphi_0$ of zero. Using (22) with the parameters $\zeta = 0.707$ and $f_n = 16$ kHz leads to a lock-in range of $\Delta f_L = 110.2$ kHz. As can be seen in Fig. 11(a) and Fig. 11(b), applying a frequency step with a little more than Δf_L leads to a phase overflow at $180°$ and, therefore, to a longer lock-in time. Taking larger frequency steps leads to rapidly increasing lock-in times and might lead to instability (depending on the loop-filter). In contrast to that, Fig. 11(c) and Fig. 11(d) show the same signals using the phase unwrap component. Much larger frequency steps were applied resulting in the same shape (scaling is proportional to the frequency step) and the same lock-in time which is independent of the frequency step.

E. Additional Low-Pass Filter

The phase error due to imperfections of the Hilbert transform was analyzed in [20] with the conclusion that periodic phase errors of twice the input signal frequency may occur. Furthermore, harmonics of the input signal may produce high frequency phase errors. These errors result in a phase modulation of the DDS frequency causing unwanted spectral components. Fortunately, the DC value of these errors is zero for amplitude imperfections and can be filtered out. Due to the fact that $E_{PI}(z)$ has only a limited attenuation at high frequencies an additional low-pass filter was added to suppress high frequency phase errors. The cut-off frequency of that filter can be much higher than the natural frequency of the PLL, so that the models of the PLL dynamics are still valid.

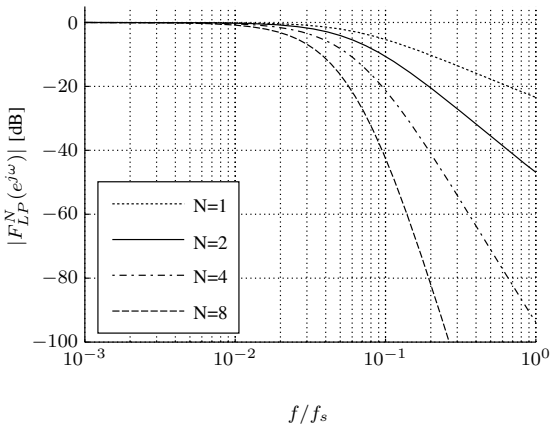


Fig. 10. Frequency response of the multiplier-less IIR low-pass filter $F_{LP}(z)$.

The used low-pass filter is based on the discretization of a simple first order analog filter

$$F_{LP}(s) = \frac{1}{1 + sT_c} \tag{39}$$

with a 3-dB cut-off frequency of $f_c = 1/T_c$. A discrete approximation with low hardware complexity can be obtained by applying the backward difference (8), leading to the discrete filter

$$F_{LP}(z) = \frac{b_0}{1 + a_1 z^{-1}} \tag{40}$$

with $b_0 = 1/(1 + T_c/T_s)$ and $a_1 = -1/(1 + T_s/T_c)$.

The specification of the low-pass filter is relaxed as we need a cut-off frequency that lies in the band between the loop bandwidth (which is typically a few kHz up to some 100 kHz) and twice the minimum signal frequency. Hence, there is a degree of freedom in choosing the coefficients.

The approach $b_0 := 2^{-L}$ leads to $a_1 = 2^{-L} - 1$, which means that only one subtracter and one delay is needed to realize the filter. With this restriction, the 3-dB cut-off frequency is

$$f_c = \frac{fs}{2^L - 1} \quad . \tag{41}$$

A cascade of filters can be used to increase the stop-band attenuation. The different frequency responses for different filter orders N (which is equal to the number of cascaded first order filters) are shown in Fig. 10 for $L = 4$ or $f_c = f_s/15$.

F. Direct Digital Synthesizer

The DDS was implemented in a common LUT based fashion [18] whereas the first quarter of the sine period is stored in a LUT. The remaining quadrants are calculated in real-time by either inverting the result, or inverting the argument of the LUT and adding $\pi/2$ or doing both, depending on the input value from the phase accumulator.

G. Noise Considerations

The sources of noise can be divided in two sources: Externally injected noise from the input signals and internal noise due to quantization effects. The rejection of injected noise

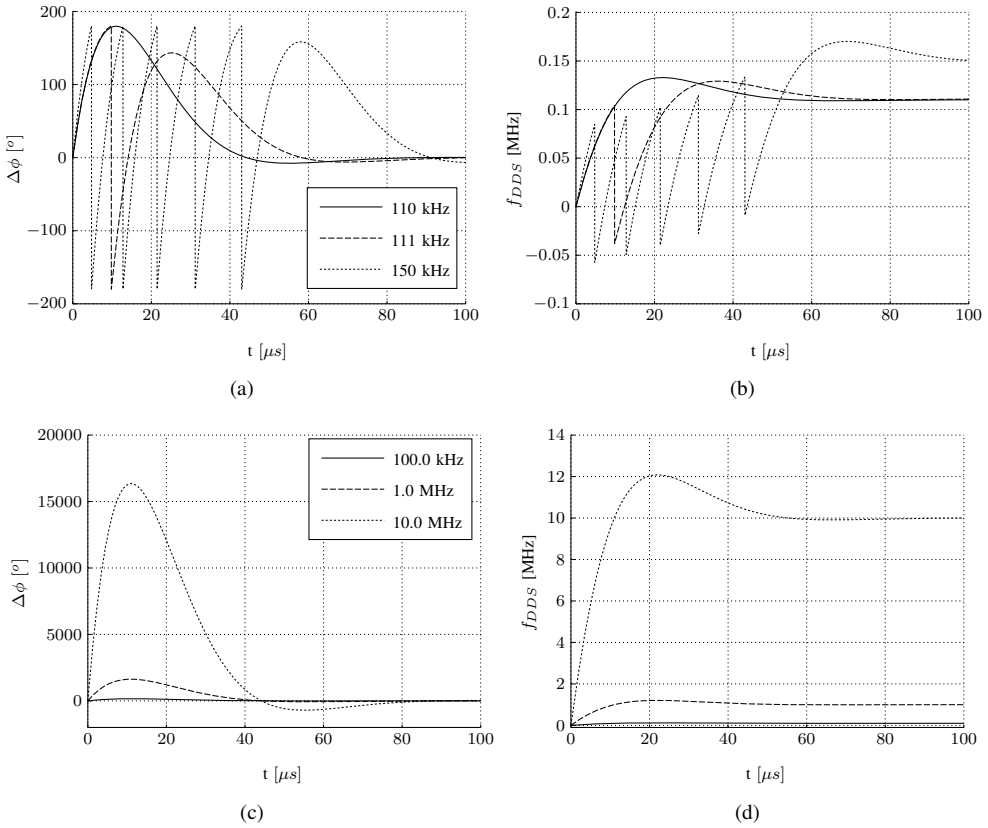


Fig. 11. Frequency step responses of the PLL designed in Section V with disabled phase unwrap (a),(b) and enabled phase unwrap (c),(d) showing the phase error $\Delta\varphi(n)$ (a),(c) and the DDS frequency $f_{DDS}(n)$ (b),(d) for different frequency steps.

TABLE II
SFDR OF A DDS IN DEPENDENCY OF THE PHASE RESOLUTION FOR AN AMPLITUDE WORD SIZE OF 12 BIT [28].

Phase word size [bit]	8	10	12	14	16
SFDR [dBc]	53	58	68	74	77

highly depends on the loop parameters, a detailed description including type-2 PLLs can be found in the textbooks [16] or [17].

The internal noise is dominated by the quantization noise. The quantization noise from the ADCs and DACs limits the signal to noise ratio (SNR) to approximately 6 dB per bit. Hence, care must be taken in choosing the right converters. A further noise source is internally generated quantization noise due to truncation. This may necessary after operations like, e. g., multiplications where the word size is nearly doubled at the output. This kind of quantization noise can be reduced to any lower bound by increasing the internal word size. The most critical word size of the design in relation to the resulting complexity is the phase resolution of the DDS, as each additional bit doubles the memory requirements of the LUT. The influence of a truncated phase accumulator to the spurious free dynamic range (SFDR) of the output signal was analyzed in [28] for phase word sizes up to 16 bit and amplitude word sizes of 12 bit. The results from [28] are summarized in Table II for an amplitude word size of 12 bit, which is close to practical applications.

V. DESIGN OF THE OFFSET LO

The following section gives an example of designing an AD-PLL using the presented formulas. The described design acts as offset LO that is used at the heavy-ion particle accelerator SIS18 at GSI for closed-loop RF controls [1], [2]. Within this system, the RF signal is converted to an intermediate frequency (IF). Therefore, a local oscillator is required whose frequency is exactly the sum of RF ($f_{RF} = 0.8...5.4$ MHz) and IF ($f_{IF} = 21.4$ MHz) in a phase synchronous way, resulting in a frequency range of $f_{LO} = f_{RF} + f_{IF} = 22.2...26.8$ MHz. The main quality requirement to the offset LO is the maximal phase deviation (related to the RF signal) which must be less than $\Delta\varphi_{max} = \pm 5°$ to avoid dipole oscillations [1]. This must be guaranteed up to a maximal frequency ramp rate during acceleration of $\dot{f}_{RF,max} = 68$ MHz/s (for protons in SIS100). In addition, a method for realizing the frequency offset is described which extends Fig. 2 to an offset LO. However, this is an optional feature which can be omitted without restrictions.

A. Offset Frequency Generation

The generation of a frequency offset in analog PLLs is often realized by a frequency translation in the feedback path of the PLL. The output signal is mixed with the IF signal and the frequency difference is filtered with a low-pass filter. The effect is a positive frequency shift by the IF at the PLL output. This method is preferred for analog PLLs as there is a lack of

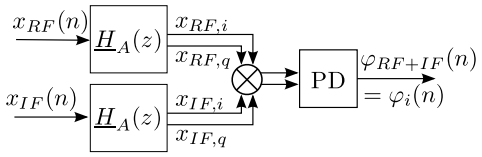


Fig. 12. Implementation of the frequency offset with complex multiplier as single sideband mixer.

ideal mixers, which makes the filtering difficult as harmonics have to be filtered, too. As an analytic representation of the frequency translated signal is necessary for phase detection, it is advantageous to first generate analytic signals from both inputs and to use a complex multiplier as single sideband mixer as shown in Fig. 12. A low-pass filter is not necessary anymore. Furthermore, both input channels can be used in the full frequency range from zero to nearly the Nyquist frequency.

B. Determination of the PLL Parameters

For the given requirements, a damping factor of $\zeta = 0.707$ was chosen as good compromise between overshoot and speed. The steady state error for a frequency ramp was used to determine the minimum necessary natural frequency. The steady state phase error for a linear frequency ramp at the input is

$$\lim_{s\to 0} \frac{2\dot{\Delta\omega}}{s^2} E(s) = \frac{2\dot{\Delta\omega}}{K_I K_d K_0} = \frac{2\dot{\Delta\omega}}{\omega_n^2} \quad . \tag{42}$$

Hence, it can be reduced by increasing the natural frequency. The minimum necessary natural frequency for the mentioned requirements using $\dot{\Delta\omega} = 2\pi \dot{f}_{RF,max}$ for a steady state phase error of $\Delta\varphi_{max}$ is given by

$$f_{n,min} = \frac{1}{2\pi}\sqrt{\frac{2\dot{\Delta\omega}}{\Delta\varphi_{max}}} = 15.75 \text{ kHz} \tag{43}$$

so, f_n was set to 16 kHz. The total delay of the components in the loop is $D = 16$ (including pipeline delays) and a sampling rate of $f_s = 120$ MHz was used. Hence, the working point is $w = 2.13 \cdot 10^{-3}$ (marked as "×" in Fig. III-B) resulting in a maximum error of less than 1.8% according to (15).

Next, the unwrap word size P is determined using (38) by setting the lock-in frequency range to $f_s/2$ leading to $P = \lceil 9.25 \rceil = 10$ bit. The controller coefficients are calculated using (12) and (13) with $K_d = 1$ and $K_0 = 2\pi f_s$ resulting in $b_0 = 188.65 \cdot 10^{-6}$ and $b_1 = -188.53 \cdot 10^{-6}$. Note that these values have to be converted to the binary coding of (33).

VI. Results

The implementation of the ADPLL witch was tested on an FPGA board equipped with an Altera Cyclone III FPGA (EP3C25), two ADC channels (LTC2255, max. $f_s = 125$ MSPS, 14 bit) and two DAC channels (AD9744, max. $f_s = 210$ MSPS, 14 bit), running at $f_s = 120$ MHz. The components were modeled in VHDL and the synthesis was done with the Altera Quartus II 9.0 software. The total design uses 4341 (18%) logic elements (LE), 53728 (9%) memory bits (mainly for the DDS) and 11 (17%) embedded 18x18 bit multipliers (for the PI controller and the complex multiplier).

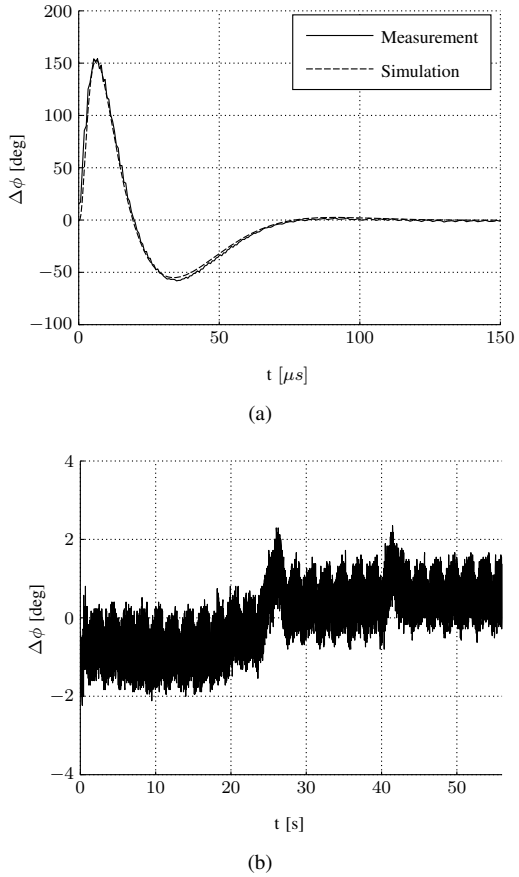


Fig. 13. (a) Measured and simulated output phase error for an input phase step of 180°, (b) measured output phase error for an input signal which is frequency modulated with a 17.5 Hz ramp ranging from 0.8 MHz to 5.4 MHz.

The simulations of the ADPLL were done using Matlab/Simulink. Starting with basic transfer function models in phase domain, the simulation was next detailed with time domain RF signals and then, VHDL components were included using Altera DSP Builder to verify the implementation. Finally, the integration was tested by measuring the time-resolved phase error. For this purpose, the PLL output signal was down-converted to the RF range by mixing with the IF signal and filtering the lower side band. Afterwards, the phase difference between input and output signal was measured with the DSP-based phase detector described in [1] which has a resolution of better than ±1°. The measured step response of the phase error from an input phase step of 180° is shown in Fig. 13(a), as well as the simulated step response. It can be seen that the model fits very well to the measurement. Next, a fast triangular frequency ramp was applied with a modulation frequency of 17.5 Hz in a range from 0.8 to 5.4 MHz corresponding to $\dot{f}_{RF} = 80.5$ MHz/s. The total phase error is shown in Fig. 13(b) and stays within ±2.2°.

VII. Conclusion and Outlook

A generic design of a linear discrete-time ADPLL using the Hilbert transform and the CORDIC algorithm as phase detector was presented. A discrete loop filter was derived whose parameters can be calculated by the very descriptive parameters *natural frequency* and *damping*, which are well-

known from second order systems, although the system is of much higher order. An analytic expression for the lock-in frequency range of this type of PLL was derived. With the use of a presented phase-unwrap component, the linear range of the PLL can be extended to the full frequency bandwidth. We have demonstrated the application as offset LO working in the lower RF range up to 60 MHz using commonly available ADC, DAC and FPGA components.

This design uses only 15% of the FPGA resources. Therefore, the proposed ADPLL can be embedded in more complex systems where an accurate synchronous signal detection is needed, e. g., in software defined radios (SDR). Otherwise, the resources can be used to realize a more sophisticated loop filter as, e. g., the Kalman observer presented in [8]. For low power applications, it could be beneficial to reduce the data rate by decimation in the loop filter as proposed in [29].

Looking at the trends in microelectronic evolution, the digital processing of signals in the ultra short wave band and above seems possible in the near future.

REFERENCES

[1] H. Klingbeil, "A Fast DSP-Based Phase-Detector for Closed-Loop RF Control in Synchrotrons," *IEEE Trans. Instrum. Meas.*, vol. 54, no. 3, pp. 1209–1213, 2005.

[2] H. Klingbeil, B. Zipfel, M. Kumm, and P. Moritz, "A Digital Beam-Phase Control System for Heavy-Ion Synchrotrons," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2604–2610, 2007.

[3] L. Musa, "FPGAS in High Energy Physics Experiments at CERN," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, Sept. 2008, pp. 2–2.

[4] V. Kratyuk, P. K. Hanumolu, U.-K. Moon, and K. Mayaram, "A Design Procedure for All-Digital Phase-Locked Loops Based on a Charge-Pump Phase-Locked-Loop Analogy," *IEEE Trans. Circuits Syst. II*, vol. 54, pp. 247–251, 2007.

[5] E. Mokhtari and M. Sawan, "CMOS High-Resolution All-Digital Phase-Locked Loop," in *IEEE Int. Symp. Micro-NanoMechatronics and Human Science*, 2004, pp. 221–224.

[6] A. I. Ahmed, S. H. Rahman, and O. A. Mohamed, "FPGA Implementation and Performance Evaluation of a Digital Carrier Synchronizer Using Different Numerically Controlled Oscillators," in *Canadian Conf. Elect. and Comp. Eng.*, 2007, pp. 1243–1246.

[7] N.-G. Kim and I.-J. Ha, "Design of ADPLL for Both Large Lock-In Range and Good Tracking Performance," *IEEE Trans. Circuits Syst. II*, vol. 46, pp. 1192–1204, 1999.

[8] W. Namgoong, "Observer-Controller Based Digital PLL," *IEEE Trans. Circuits Syst. I*, to be published.

[9] C.-P. Chen, M.-J. Yang, H.-H. Huang, T.-Y. Chiang, J.-L. Chen, C. M.-C., and K.-A. Wen, "A Low-Power 2.4-GHz CMOS GFSK Transceiver With a Digital Demodulator Using Time-to-Digital Conversion," *IEEE Trans. Circuits Syst. I*, vol. 56, no. 12, pp. 2738–2748, Dec. 2009.

[10] K.-H. Choi, J.-B. Shin, J.-Y. Sim, and H.-J. Park, "An Interpolating Digitally Controlled Oscillator for a Wide-Range All-Digital PLL," *IEEE Trans. Circuits Syst. I*, vol. 56, no. 9, pp. 2055–2063, Sept. 2009.

[11] G.-R. Tsai, M.-C. Lin, J.-W. Hsieh, and Y.-C. Lin, "A Novel Ultra-High Speed Signal Capture Based on a Single FPGA Chip," in *IEEE Asia-Pacific Conf. Circuits Syst.*, 2004, pp. 261–263.

[12] F. Kobayashi and M. Haratsu, "A Digital PLL with Finite Impulse Responses," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 1995,

[13] J. Vuori, "Implementation of a Digital Phase-Locked Loop Using Cordic Algorithm," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 1996, 164-167.

[14] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Transactions on Electronic Computers*, pp. 330–334, 1959.

[15] C.-T. Wu, W. Wang, I.-C. Wey, and A.-Y. Wu, "A Frequency Estimation Algorithm for ADPLL Designs With Two-Cycle Lock-in Time," in *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, 2006, pp. 4082–4085.

[16] F. M. Gardner, *Phaselock Techniques, 3rd edition*. New York, Chinchester, Brisbane, Toronto: John Wiley & Sons, Inc., 2005.

[17] V. F. Kroupa, *Phase Lock Loops*. John Wiley & Sons Ltd, 2003.

[18] ——, *Direct Digital Frequency Synthesizers*. IEEE Press, 1998.

[19] S. L. Hahn, *Hilbert Transforms in Signal Processing*. Artech House on Demand, 1996.

[20] M. Kumm and M. S. Sanjari, "Digital Hilbert Transformers for FPGA-based Phase-Locked Loops," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, 2008, pp. 251–256.

[21] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, New Jersey, United States: Prentice Hall Inc., 1975.

[22] J. Wilson, A. Nelson, and B. Farhang-Boroujeny, "Parameter Derivation of Type-2 Discrete-Time Phase-Locked Loops Containing Feedback Delays," *IEEE Trans. Circuits Syst. II*, vol. 56, no. 12, pp. 886–890, Dec. 2009.

[23] H. Unbehauen, *Regelungstechnik I*. Vieweg, 2005.

[24] K. I.N. Bronštein, *Handbook of Mathematics*, 2004.

[25] J. Stensby, "An Approximation of the Pull-Out Frequency Parameter in a Second-Order PLL," in *Proceeding of the Thirty-Eighth Southeastern Symposium on System Theory*, 2006, pp. 75–79.

[26] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays, 3rd Edition*. Berlin: Springer, 2007.

[27] A. Guntoro, P. Zipf, M. Glesner, and H. Klingbeil, "Status Report "Reconfigurable Computing Systems for Digital High-frequency Control of Heavy Ion accelerators" - Extending the Phase Detector module: Magnitude and Phase Information," Technical University Darmstadt, Tech. Rep., 2006.

[28] H. Nicholas and H. Samueli, "An Analysis of the Output Spectrum of Direct Digital Frequency Synthesizers in the Presence of Phase-Accumulator Truncation," in *Proceedings of the 41st Annual Frequency Control Symposium*, 1987, pp. 495–502.

[29] Y. Linn, "A Methodical Approach to Hybrid PLL Design for High-Speed Wireless Communications," in *Wireless and Microwave Technology Conference, 2006. WAMICON '06. IEEE Annual*, Dec. 2006, pp. 1–9.

Martin Kumm was born in Fulda, Germany, in 1980. He received the Dipl.-Ing. degree in electrical engineering from the University of Applied Sciences of Fulda, Germany and from the Technical University of Darmstadt, Germany in 2003 and 2007, respectively. From 2003 to 2009 he was with GSI Darmstadt, Germany, working on digital RF control systems for particle accelerators. He is currently working toward the Ph.D. degree in the Digital Technology group at the University of Kassel, Kassel, Germany. His present research interests are digital signal processing, filter circuits and computer aided design, all in the context of FPGA devices and VLSI technology.

Harald Klingbeil was born in Frankfurt, Germany, in 1968. He received the Dipl.-Ing. degree and the Dr.-Ing. degree in electrical engineering, both from the Technical University in Darmstadt in 1992 and 1997, respectively. From 1997 to 2001, he worked in automotive industry in the area of telematics, navigation and multimedia systems. Since 2002, he is with GSI where he currently leads the RF department. His current research interests are digital RF control systems for particle accelerators.

Peter Zipf (M'05) received the B.A. degree in computer science from Kaiserslautern University, Kaiserslautern, Germany, in 1994, and the Ph.D. degree (Dr.-Ing.) from University of Siegen, Siegen, Germany, in 2002. Until 2009, he was Postdoctoral Researcher at the Department of Electrical Engineering and Information Technology, Darmstadt University of Technology (Technische Universität Darmstadt), Darmstadt, Germany.

He currently holds the chair for Digital Technology at University of Kassel, Kassel, Germany. His research interests include dynamically reconfigurable computing, embedded system and system-on-chip design, as well as design methodologies and CAD tools for reconfigurable systems.