Software PLL for Power-line Interference Synchronization: Implementation and Results

Dobromir Petkov Dobrev and Tatyana Dimitrova Neycheva

Abstract – **Power-line interference is a common disturbing factor in almost all two-electrode biosignal acquisition applications. Many filtering procedures for mains interference elimination are available, but all of them are maximally effective when the filter notches are positioned exactly at the power-line harmonics, i. e. when the sampling rate is synchronous with the power-line frequency. Moreover, various lock-in techniques, such as automatic common mode input impedance balance, require precise in-phase and quadrature phase references, synchronous with the power-line interference. Recently a design methodology of software PLL for power-line synchronization was published. This paper describes the results of its practical realization.**

Keywords – **Software PLL (SPLL), All-digital PLL (ADPLL), Power-line Synchronization**

I. Introduction

Power-line (PL) interference is a common disturbing factor in almost all biosignal acquisition applications. As a consequence of electrode impedance imbalance and the finite value of the amplifier CMRR, some AC noise remains even when special signal recording techniques are applied (shielding, driven right leg, body potential driving, etc.). A further reduction of PL interference usually is achieved by digital post-filtering. Many algorithms for PL interference suppression are available, starting from simple comb filters [1], to advanced subtraction procedures and lock-in techniques [2, 3], but all of them tend to lose their efficiency when PL frequency differs from its nominal value. Maximal rejection is possible only when the sampling rate is synchronous with the PL frequency, because only at that case, the filter notches coincide with the PL harmonics.

A lock-in technique for input common mode impedance balance was developed [4]. The approach is based on two digitally regulated control loops to maintain resistive and capacitive input common mode impedance balance. The control loops require precise in-phase and quadrature phase references, synchronous with the common mode PL interference. If the synchronization is lost, the negative feedback can become positive, leading to instability problems and lack of convergence.

Recently a design methodology of Software PLL (SPLL) for synchronization with the PL interference was published [5]. It was shown how the PLL loop gain could be evaluated in *s*-domain, and how the *z*-domain transfer function can be derived from its analog prototype using *s* to *z* backward difference mapping. Now, the discussed SPLL is realized on STM32F407 microcontroller, and this paper gives some results of its implementation.

D. Dobrev is with the Institute of Biophysics and Biomedical Engineering – Bulgarian Academy of Sciences, Bl. 105 Acad G. Bontchev Str., 1113 Sofia, Bulgaria,
e-mail: dobri@biomed.bas.bg

T. Neycheva is with the Institute of Biophysics and Biomedical Engineering – Bulgarian Academy of Sciences, Bl. 105 Acad G. Bontchev Str., 1113 Sofia, Bulgaria,
e-mail: tatiana@biomed.bas.bg

II. Overview of SPLL Design

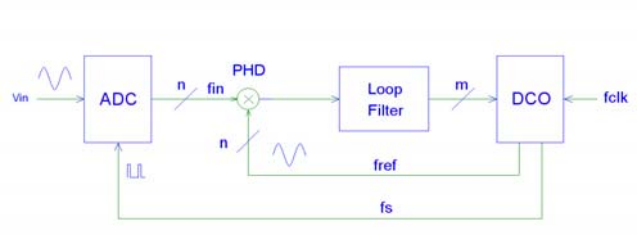


Fig. 1. Software PLL structure

The SPLL structure is shown in *Fig. 1*. It consists of three blocks: Phase Detector (PHD), Loop Filter and Digitally Controlled Oscillator (DCO). The input signal V_{in} is processed in digital form, after ADC it is *n*-bits data stream. The mixer or phase detector (PHD) simply is a multiplier. Square wave mixing greatly simplifies the mixer operation, but could be used only when the loop bandwidth is lower enough in comparison to the generated frequency. Sine wave mixing is preferable when low jitter is a must. The Loop Filter (LF) integrates the data in time, and due to averaging increases the resolution, so the DCO input could be *m*-bits ($m \geq n$) word. For proper processing at low oversampling ratios, the sampling rate f_s must be multiple to the reference frequency f_{ref}. Once the DCO range is defined to cover all variation of the input frequency with reserve, the SPLL design is reduced to the LF design. Thus, the LF must be carefully designed to provide stable system with appropriate settling time.

The loop gain is responsible for the stability of each control system. In the presented SPLL design, the loop gain analysis is derived from its analog PLL prototype, see *Fig.2*. The shown structure contains two integrators, i. e. it represents a second order transfer function. The first pole, at DC, is related to the VCO, which serves as an ideal integrator included in the loop [6, 7]. The second pole, also at DC, is due to the integrator in the Loop Filter, and must be compensated for stability by adding a zero with forward path with a coefficient k_z.

The Phase Detector gain G_{PHD} has dimension $\frac{V}{rad}$, while the VCO gain G_{VCO} has dimension $\frac{rad}{sV}$. The reciprocal value of the product $G_{PHD}G_{VCO}$ has dimension of seconds, and for simplicity can be denoted by a symbol τ_{vco}. So, the

timeconstant τ_{vco} determines the roll-off of the inherent integrator in the loop, hidden in the VCO operation.

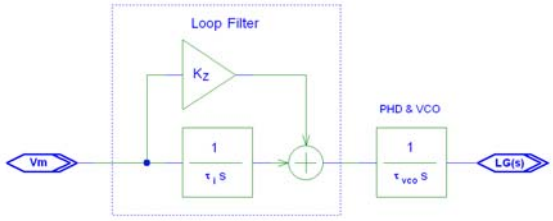


Fig. 2. PLL loop gain in analog prototype

Thus, the transfer function of the shown in *Fig*. 2 loop gain is dimension less, and can be expressed with Eq. (1):

$$LG(s) = \frac{1 + k_z \tau_i s}{\tau_i s} \cdot \frac{1}{\tau_{vco} s} \tag{1}$$

More details on analysis of Eq. (1) can be found in [5]. The goal is to be found the unity gain frequency of the open loop gain, because it determines the bandwidth and stability of the closed loop system. For τ_{vco}=1.3s, τ_i=1s and k_z=8, the open loop unity gain frequency is about 1Hz. Thus, the closed loop bandwidth will be fast enough because a bandwidth of only 0.1Hz is sufficient for tracking PL frequency.

Next, the loop gain in *z*-domain easily can be found using the backward difference mapping of *s*-plane to *z*-plane according the Eq. (2) [5, 8]. Here, $T=1/f_s$ is the sampling interval, reciprocal to the sampling frequency f_s.

$$s = \frac{1 - z^{-1}}{T} \tag{2}$$

In analog PLL usually a third pole is inserted in the LF for high-frequency filtering and reducing VCO jitter. It affects frequencies faraway from 0dB point to maintain stability. In SPLL the best and simple way for high-frequency filtering is by one PL period moving-average filter (averager). The averager effectively cancels all harmonics of PL frequency and will reduce ripples at the VCO input. Adding 1PL period averager will introduce group delay of 10ms in the loop. Evaluated at 1Hz, this delay corresponds to 3.6 degree phase lag. Adding additional delay of one sampling period at f_s=2kHz, or 0.18 degree, the total phase margin will drop by about 4 degree. The conclusion is that 1PL period averager is possible to be added in the loop, and will reduce the phase margin by 4 degree. The modified loop filter by added 1PL period averager is shown in *Fig. 3*.

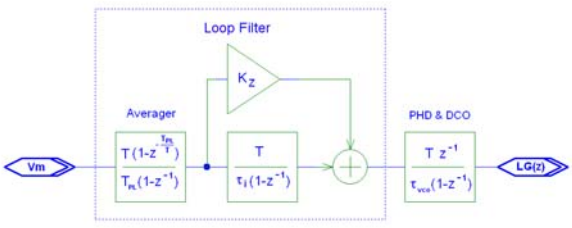


Fig. 3. SPLL loop gain in *z*-domain

The loop gain from *Fig. 3* can be written as Eq. (3):

$$LG(z) = LF(z) \cdot \frac{Tz^{-1}}{\tau_{VCO}(1 - z^{-1})} \tag{3}$$

Where LF(*z*) is the loop filter transfer function, and the second multiplicand is the transfer function of the phase detector and the DCO. The loop filter transfer function LF(*z*) includes 1PL period averager, and can be expressed with Eq. (4):

$$LF(z) = \frac{T(1 - z^{-\frac{T_{PL}}{T}})}{T_{PL}(1 - z^{-1})} \cdot \frac{T + k_z \tau_i (1 - z^{-1})}{\tau_i (1 - z^{-1})} \tag{4}$$

Adding a coefficient $k_i = \frac{T}{\tau_i}$ in Eq. (4), it can be rewritten as Eq. (5):

$$LF(z) = \frac{1 - z^{-\frac{T_{PL}}{T}}}{1 - z^{-1}} \cdot \frac{T}{T_{PL}} \cdot \frac{k_i + k_z (1 - z^{-1})}{1 - z^{-1}} \tag{5}$$

The transfer function expressed in Eq. (5) can be realized with signal flow schematic shown in *Fig. 4*.

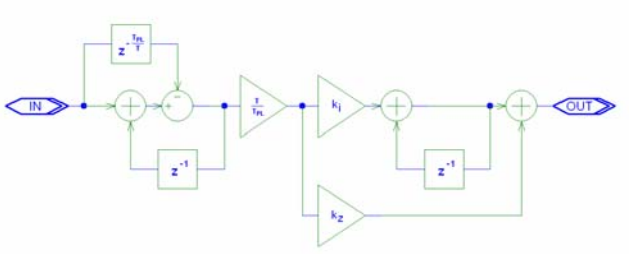


Fig. 4. SPLL loop filter realization

III. Loop Filter Optimization in Matlab

Simulink schematic for loop filter optimization is shown in *Fig.5*. Matlab simulations are run to evaluate the stability of the SPLL and to optimize the LF coefficients k_i and k_z. The goal is to achieve as fast as possible stable response.

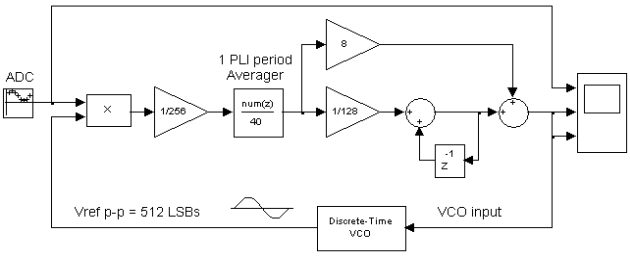
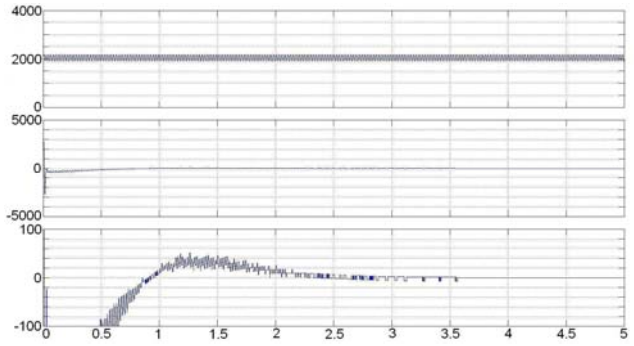


Fig. 5. Simulink schematic for LF optimization

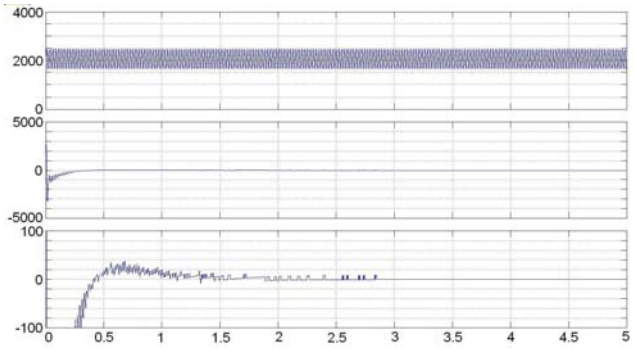
For lower DCO input jitter a sine wave mixing is used. For avoiding floating point multiplications, the DCO generates sine wave with 256LSBs amplitude. The mixer output is divided by 256 to keep the loop gain. The DCO sensitivity is 1mHz/LSB, and 1LSB corresponds to 3V/4096=0.732mV, thus the DCO sensitivity is 1.36Hz/V.

The coefficient k_z is fixed to k_z=8. The coefficient k_i was varying starting from k_i=0.5m, as was shown in [5]. It was found that fast and stable response is achieved when k_i is increased up to k_i=8m, which value easily can be implemented by a division of 128, i. e. $k_i=2^{-7}$=7.8m. The phase margin, when k_i=7.8m, drops to about 65 degree but still preserves good stability.

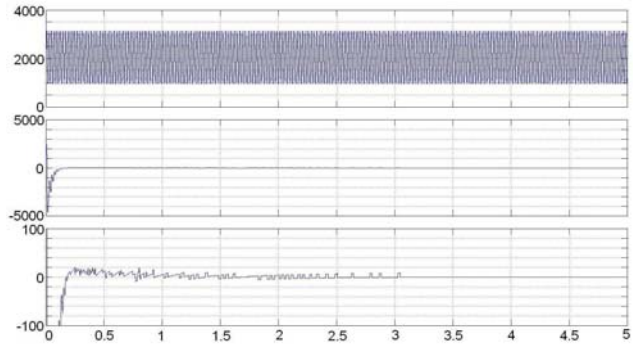
The simulation results are shown in *Fig. 6*. The first trace is the input frequency. The second and the third traces are the DCO input at a different zooms. Because the loop speed depends on the input amplitude, the stability must be checked in all possible variations of the input amplitude, i. e. in its minimum, typical and maximum values.



a) Min. V_{in}=200mV$_{pp}$, f_{in}=50Hz



b) Typ. V_{in}=600mV$_{pp}$, f_{in}=50Hz



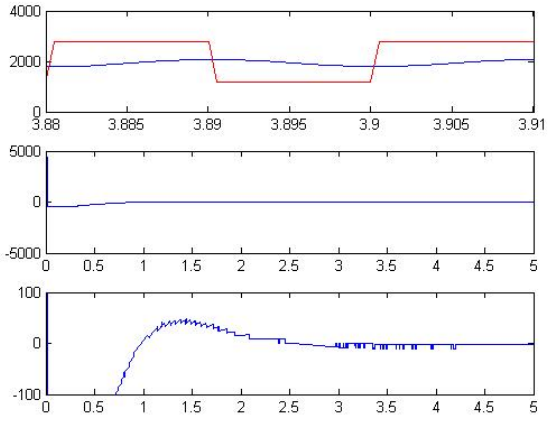
c) Max. V_{in}=1.6V$_{pp}$, f_{in}=50Hz

Fig. 6. Simulink simulation results

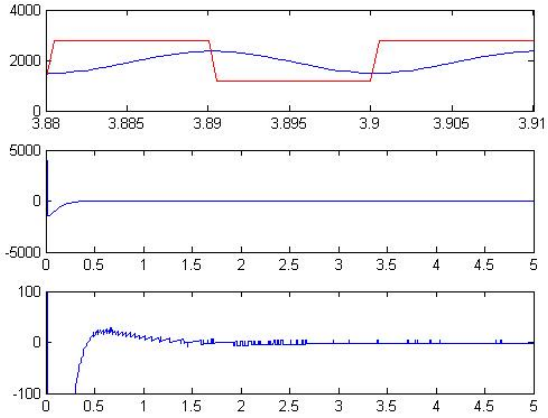
It can be seen that the settling of the DCO input has a small overshoot but the response is stable in all variations of the input amplitude.

IV. Practical Realization and Results

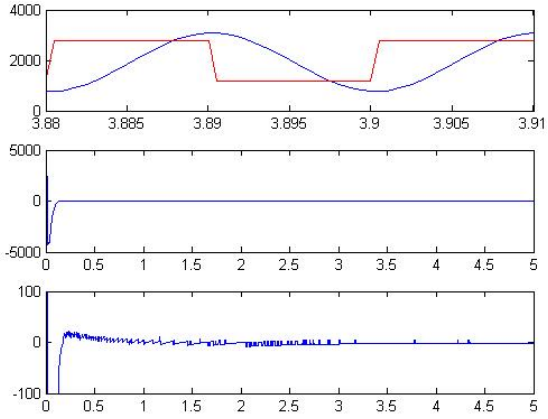
The SPLL is implemented on the microcontroller STM32F407. The microcontroller incorporates a 12-bits ADC which is used for converting the input signal. At the first trace the input frequency, and the generated reference converted in rectangular form are shown. The second and the third traces are the DCO input at a different zooms. It can be seen that the stability shown in the practical results corresponds to the Matlab simulations shown in *Fig. 6*.



a) Min. V_{in}=200mV$_{pp}$, f_{in}=50Hz

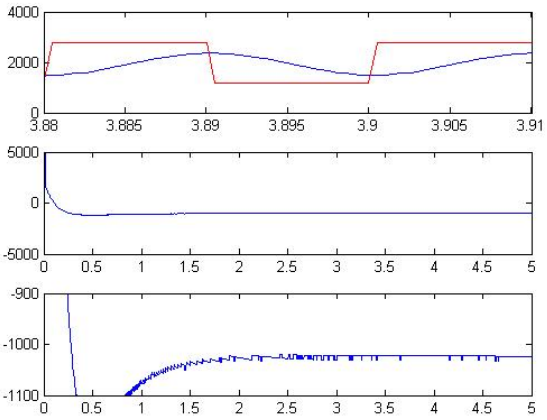


b) Typ. V_{in}=600mV$_{pp}$, f_{in}=50Hz

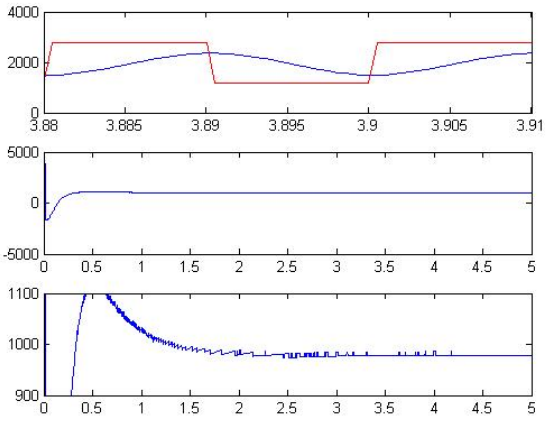


c) Max. V_{in}=1.6V$_{pp}$, f_{in}=50Hz

Fig. 7. Practical results for f_{in}=50Hz



a) Typ. V_{in}=600mV$_{pp}$, f_{in}=49Hz



b) Typ. V_{in}=600mV$_{pp}$, f_{in}=51Hz

Fig. 8. Practical results for f_{in} variation ±1Hz

Practical results when f_{in} vary ±1Hz are shown in *Fig.* 8. The DCO input is settled to about ±1000LSBs. As was noted previously, the loop speed depends on the input signal amplitude. Automatic Gain Control (AGC) of the input signal amplitude could be added in addition for constant settling time [5]. *Fig. 7* and *Fig. 8* show the real operation of the microcontroller. The shown data are in LSBs vs. Time, and 1LSB corresponds to 0.732mV. The data are transferred to PC, and are visualized with Matlab.

From the DCO input shown in the third trace in *Fig. 7* and *Fig. 8*, it can be seen that the loop has stable response in all variation of the input amplitude and frequency from V_{in}=200mV$_{pp}$ to V_{in}=1.6V$_{pp}$, and f_{in}=±1Hz. When the DCO input is settled the generated rectangular reference, derived from the used in the mixer sine wave reference, leads the input sine wave in 90 degree, see the first traces in *Fig. 7* and *Fig. 8*. To minimize the DCO input remaining ripple, the ADC sampling rate is multiple to the generated reference. Thus, the averager, included in the loop filter, is maximally effective in rejection the PL harmonics. Note, that the 1PL period averager is a comb filter with notches at all harmonics of the PL interference. It plays a very important role as a part of the loop filter for lowering the pulsation at the DCO input. Without averager the DCO will operate with higher level of ripples, due to forward path in the LF integrator, see *Fig. 4*.

V. CONCLUSION

The SPLL implementation is discussed, and real practical results of its operation are shown. The results correspond to the made simulations. The DCO input has stable response in all possible variations of the input signal amplitude and frequency. The main advantage of the approach is that the synchronization is done in software, so it has no production cost.

The SPLL purpose is to generate synchronous reference to the common mode PL interference in two-electrode amplification. It is intended for use in ECG signal processing, but can be used after easy adaptation in various digital signal processing applications, where frequency synchronization is needed.

A design methodology of SPLL was described in [5], where it was shown, how the SPLL *z*-domain transfer function can be derived from its analog PLL *s*-domain prototype. The two articles, this and the previous one [5], could be used as a SPLL tutorial because all steps in the design process are considered, from modeling and simulation to final realization and validation. Of course, to understand the subject, the reader should have at least a basic knowledge with feedback control theory, and with *s* and *z* domain transfer functions.

REFERENCES

[1] D. Dobrev, T. Neycheva, N. Mudrov. *Simple High-Q Comb Filter for Mains Interference and Baseline Drift Suppression*, Annual Journal of Electronics, Vol. 3, No 1, pp. 50-52, 2009.

[2] Ch. Levkov, G. Mihov, R. Ivanov, I. Daskalov, I. Christov, I. Dotsinsky. *Removal of Power-Line Interference from the ECG: a Review of the Subtraction Procedure*, Biomed Eng Online 4:50, 2005.

[3] D. Dobrev, T. Neycheva, N. Mudrov. *Digital Lock-in Techniques for Adaptive Power-Line Interference Extraction*, Physiol Meas 29, pp. 803–816, 2008.

[4] D. Dobrev, T. Neycheva. *Digital Lock-in Technique for Input Impedance Balance in Two-electrode Biosignal Amplifiers*, Annual Journal of Electronics, Vol. 7, No 1, pp. 64-67, 2013.

[5] D. Dobrev, T. Neycheva. *Software PLL for Power-line Interference Synchronization: Design, Modeling and Simulation*, Annual Journal of Electronics, Vol. 8, pp. 58-61, 2014.

[6] D. Dobrev. *Use PSpice for Behavioral Modeling of VCOs*, EDN, pp. 102-104, 2002.

[7] W. Li, J. Meiners. *Introduction to PLL System Modeling*, TI Application Journal slyt015, pp. 5-10, 2000.

[8] J. Proakis, D. Manolakis. *Digital Signal Proccesing Principles, Algorithms and Applications*, Prentice-Hall, 1996.