Design of an Online Clock Recovery System Using PLL

Mohammad Hassan Ghaed, Reza Kazemi
School of Electrical Engineering
Sharif University of Technology, Tehran, Iran
email: mhaghaed@yahoo.com

Abstract

This paper deals with the problem of clock recovery in communication links. Concerning the fact that in the digital world most of the data is transmitted in serial format, a lot of systems send the signal and the clock simultaneously. In this paper, an innovative method for extracting the clock in such systems is introduced. The method relies on the a PLL-like feedback loop that can lock on the input clock with in outstandingly short time. The method has proven to be quite effective in simulations. The applications vary from computer networks to wireless communication. Moreover, It can be quite useful in instrumentation purposes in which the link is the bottleneck of system stability.

Keywords Clock Recovery, PLL , Phase Locked Loop, Communication Systems

1 Introduction

Clock recovery is one of the most active research areas in digital communication. The process of extracting the clock from a signal itself is of immense importance when fast transmission of digital data is desired. PLLs are one of the most powerful methods for such applications in which error tolerance and convergence time are important concerns of the designers. In this paper we introduce a PLL-based method for acquiring the clock in digital communication.
In section 2 we start with an introduction to PLLs and their formulation. Section 3 Describes the clock recovery problem and restrictions. In section 4 the use of PLLs for clock extraction is described. The algorithm is put into practice in simulations in section 5. Finally, section 6 draws conclusions of the paper.

2 Phase Locked Loops

Throughout years PLLs have proven to be one of the most effective tools in various applications. Although quite intricate in theory, PLL's nature can be understood qualitatively.
The block diagram of a simple PLL is demonstrated in Figure 1. Its consists of a nonlinear feedback loop that locks the output on the input frequency. As implied by the diagram, the PLL consists of 4 essential components:

1. **Phase Detector:** The phase detector measures the difference between the input and output phase. Whenever the output signal lags a bit, its output mounts and activates the loop
2. **VCO:** It generates a sinusoid whose frequency variance from its center frequency is proportional to its input:

$$\omega_{out} = \omega_c + k_v v_(t) \quad (1)$$

3. **Loop Filter:** The loop filter smoothes the output of the PLL. The performance of the PLL relies on the loop filter used in the loop.
4. **Frequency Divider:** The divider makes the output frequency n-times the input frequency.

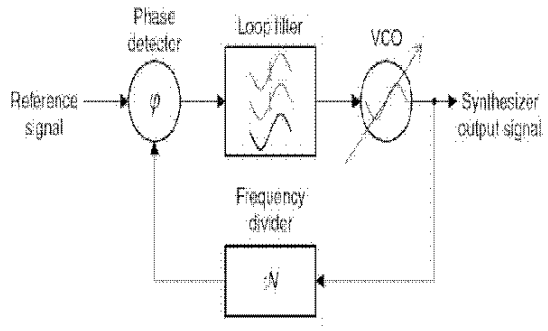

Figure 1: Block Diagram of a Simple PLL

The functionality of the PLL can be described as follows: once the input frequency undergoes a change in its frequency, the phase detector detects a change and its output increases. As a result, the loop increases the VCO's frequency in such a way that the phase difference is compensated.
Let the VCO reference signal be

$$v_{r(t)} = A_r \cos(w_{r(t)} + \phi_{r(t)}) \tag{2}$$

Also, assume that

$$v_{o(t)} = A_o \cos(w_{c(t)} + \phi_{o(t)}) \tag{3}$$

$$\phi_{o(t)} = 2\pi \int^{t} v_{in(\lambda)} d\lambda \tag{4}$$

Where $v_{in(t)}$ and ω_c are the input signal and the center frequency of the VCO respectively.
For simplification we can consider the detector's output as:

$$e_{(t)} = f_{(\theta_{r(t)}\theta_{o(t)}+\alpha)} \tag{5}$$

In which θ_r and θ_o are the are the phases of the two inputs of the phase detector. For instance, for a phase detector whose output is the product of its two inputs we will have:

$$e_{(t)} = k_d \sin(\Delta\omega t - \phi_{o(t)} + \phi r(t)) \tag{6}$$

Note that:

$$\phi_{o(t)} = 2\pi \int^{t} v_{in(\lambda)} d\lambda \tag{7}$$

Now, if we define:

$$m_{(t)} = \Delta\omega t - \phi_{o(t)} + \phi r(t) \tag{8}$$

by differentiation the resulting equation would be:

$$m'_{(t)} = \Delta\omega - \phi'_{o(t)} + \phi' r(t) \tag{9}$$

If we assume that $\phi r(t)$ is constant, we'll have

$$e_{(t)} = \sin(m(t)) \tag{10}$$

and the following differential equation is derived:

$$m'_{(t)} + 2\pi k_d \sin(m(t)) = \Delta\omega \tag{11}$$

This equation doesn't have an explicit solution, but if we use the estimation:

$$\sin(m_{(t)}) \simeq m_{(t)} \tag{12}$$

then a first order differential equation with the following solution is derived:

$$m_{(t)} = m_{(t_0)} e^{2\pi k_d (t-t_0)} \quad ; \quad t >= t_0 \tag{13}$$

3 Clock Recovery

Today, communication systems are experiencing a fast transition to broadband era. In the world of broadband signaling, parallel link face acute problems such as difficult routing and high EMI. Consequently, serial transmission has become the dominant type of data transfers. Many of these serial methods send data combined with the clock. In other words, it's the receiver's responsibility to distinguish clock frequency and phase.
Figure 2 shows the structure of a serial link. Note that the serializer converts the parallel data to serial link which is appropriate for the link. Later at the receive side, this serial data is de-serialized and the clock is extracted. The purpose of this paper is to introduce a method for this extraction. In order for the receive side to find out the clock frequency of the data stream, the stream's frequency must remain approximately constant in long-term.
There are two options for digital transmission: unipolar and bipolar. Since unipolar signals have a DC

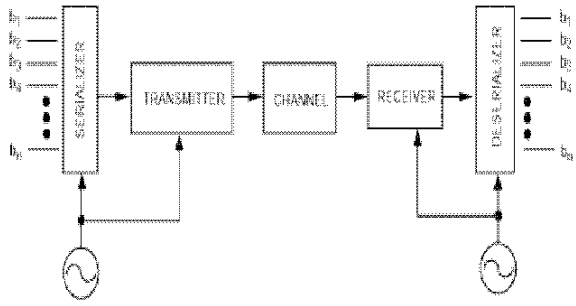

Figure 2: Block Diagram of a Serial Link

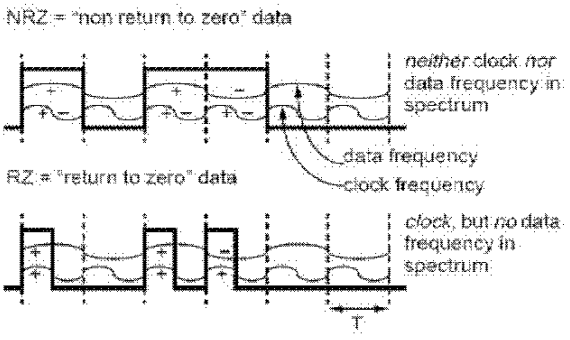

Figure 3: NRZ and RZ signals

component, they consume some additional energy. As a result bipolar transmission is the prominent style for sending digital data.

From another point of view, the link signal can be either "non returning to zero" or "returning to zero". These two signals are exhibited in Figure 3. As far as the spectrum of the signal is concerned, bipolar NZR signals don't have a peak at their clock frequency. Nevertheless, if a nonlinear operator acts on it, its clock frequency stands out in the spectrum. Figure 4 demonstrates the spectrum of such a signal before and after being squared.

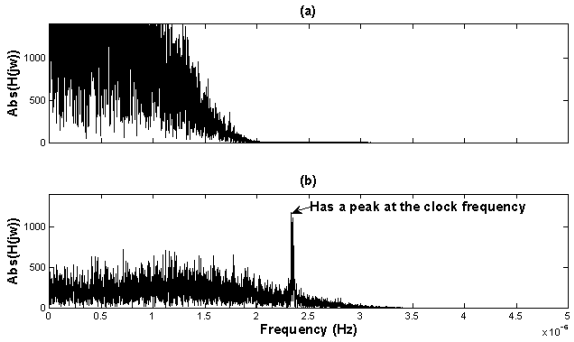

Figure 4: a. The Spectrum of Bipolar NZR Signal Before Being Processed by a Square-Law Device b. The Spectrum After Being Processed by such a Device

4 Using PLLs to Recover the Clock

The method described here takes advantage of two PLL loops in order to detect the clock signal of a data stream. As we know, the optimum signal in a lossy transmission line is the one with sinc-like pulses. This minimizes the low-pass effect of the channel. Concerning this fact, the received signal has local maxima at its optimum sampling times. In other words, if t_k is an optimum sampling time:

$$|y_{(t_k-\delta)}| \approx |y_{(t_k+\delta)}| < |y_{(t_k)}| \tag{14}$$

So, a lagging clock has a property that:

$$|y_{(t_k-\delta)}| > |y_{(t_k+\delta)}| \tag{15}$$

If we feed this difference back to the input of a PLL, it can modify its phase until the phase of the generated clock matches that of the input signal. Figure 5 evinces the block diagram of this method.
In the next section we will simulate the block diagram in order to probe its performance.

5 Simulation and Results

In order to examine the clock recovery method, a communication system model is created as seen in

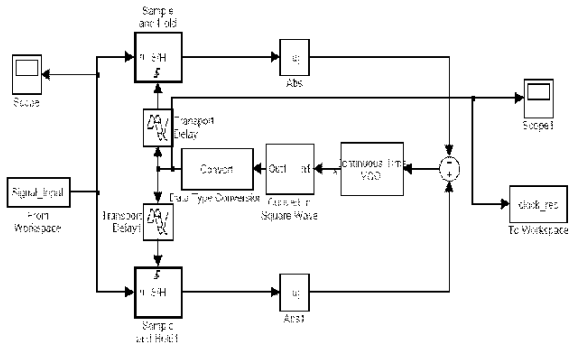

Figure 5: The Block Diagram of PLL-Based Clock Recovery Module

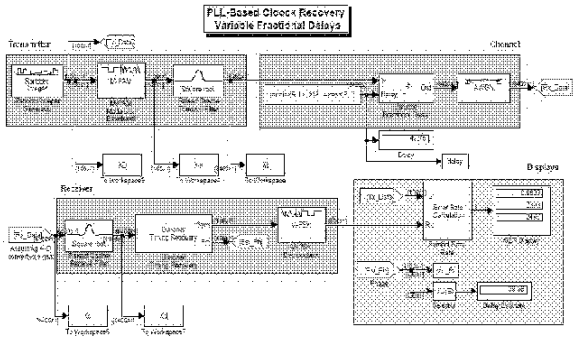

Figure 6: The Communication System Used to Benchmark the CDR Method

Figure 6 in SIMULINK. The signals created in this model are shown in Figure 7. Note that the link signal was reshaped to a sinc-like pattern in order to minimize the effect of low-pass link.

At the receiver side, the signal is processed by the pre-mentioned algorithm. The output of the algorithm is shown in Figure 8. Although the detected and transmitter frequency are not even close to each other, the approach successfully adapts to the signal be generating the same frequency as the transmitter after 400 cycles.

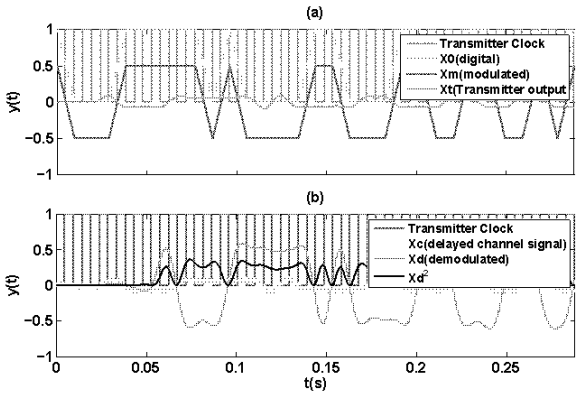

Figure 7: a.The Signals Present at the Transmitter Side b.The Signals Present at the Receive Side

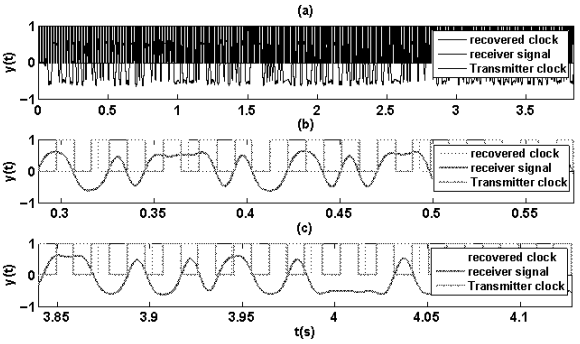

Figure 8: a.The Change of PLL Clock in 400 Cycles b.The Initial Status of the PLL in the First 30 Cycles c.The Match of the Output Frequency with Transmitter Clock after 400 Cycles

6 Conclusion

An approach for solving the problem of clock recovery was presented in this paper. The solution proved to be quite effective and fast-converging in simulations. The flexibility of the algorithm is of utmost importance since in reality, we deal with cases in which no information of the line bitrate is provided and the program must lock on the input frequency no matter what its initial frequency was. The method is especially useful in high-bandwidth links that require fast response to slight variations in the frequency of incoming signal due to link delays.
The most outstanding advantage of the abovementioned approach is the balance between complexity and efficiency. While demonstrating spectacular performance in simulations, the informative block diagram of Figure 5 clearly exhibits where does this efficiency come from. This makes upgrading the algorithm easier.

Acknowledgement

The author would like to thank Mr. Amir Hejazi that gently provided important help about communication aspects of SIMULINK.

References

[1] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, 4th edition, 2002

[2] A. Oppenheim and R. Schafer , *Discrete Signal Processing*, Prentice Hall, 2nd edition, 1999

[3] A. B. Carlson, P. B. Crilly and J. C. Rutledge *Communication Systems*, McGraw-Hill, 4th edition, 2002

[4] E. Arthurs and H. Dim, "On the Optimum Detection of Digital Signals in the presence of White Gaussian Noise in ," IRE Trans. Commun. Systems, Vol. CS-10, pp. 336-372

[5] F. M. Gardner *Phaselock Techniques*, Wiley, 2th edition, 1979

[6] E. W. Kamen and B. S. Heck *Fundamentals of Signals and Systems Using MATLAB*, Prentice-Hall, 1997

[7] J. G. Proakis and M. Salehi *Contemporary Communication Systems Using MATLAB*, Pacific Grove, 1998