Multi-rate Synchronization of Digital Receivers in Software-Defined Radios

Joseph Gaeddert, Haris I. Volos, Drew Cormier, and Jeffrey H. Reed
Mobile and Portable Radio Research Group (MPRG), Wireless@Virginia Tech
Bradley Department of Electrical and Computer Engineering, Virginia Tech
432 Durham Hall, MS 0350, Blacksburg, VA 24061, USA
E-mail: jgaedder@vt.edu

***Abstract*— This paper describes a multi-rate synchronizer that makes use of a polyphase filter bank to simultaneously perform matched-filtering and interpolation to correct for symbol timing offsets observable on a sampled-data receiver. Interpolation between available sample points is achieved by selecting the appropriate filter in the bank which provides the optimal sampling time. Furthermore, carrier phase error is corrected by adding a second control loop to drive a numerically-controlled oscillator before matched filtering, mitigating distortion caused by any carrier offset.**

This design is tractable for burst-mode transmissions where symbol timing and carrier frequency/phase acquisition is necessary for each data frame and allows for symbol timing and carrier offset estimators to be used in conjunction with the loop control architecture provided. Simulations under various timing and carrier impairments are provided.

I. Introduction

Software-defined radios (SDR) aim to move as much digital signal processing (DSP) as close to the antenna as possible, creating a flexible architecture for reconfiguration and adaptability. As a result, SDR provides a versatile platform unavailable to analog systems. Synchronous data communications systems on SDR platforms rely on DSP algorithms to correct for symbol timing and carrier phase impairments observable at the receiver. Physical layer synchronization of symbol timing is required when samples of the received signal are misaligned with the data symbols generated by the transmitter. Several options to align the matched filter output samples are available; signal filtering, however, consumes a significant portion of baseband processing, and, as a result, oversampling the received signal in excess of the Nyquist rate is undesirable. Alternatively, interpolating between available sample points proves to be computationally efficient. Furthermore, the use of a polyphase filterbank as an effective phase shifter in the sampling clock allows for greater flexibility and efficiency in the receiver by computing only those multiplications necessary for matched filtering while simultaneously interpolating to achieve a sample point sufficiently close to the optimum.

Rice and Harris proposed the use of polyphase filter banks for symbol synchronization in digital receivers in [1] and developed loop control architectures for choosing the optimal filter in the bank. This paper extends their work by adding an additional control loop for carrier synchronization after matched filtering. A numerically-controlled oscillator placed before matched-filtering the received signal ensures that pulse distortion due to input carrier offsets is minimized. This design allows for simultaneous tracking of both symbol timing and carrier phase mismatches and is particularly tractable for burst-mode duplexing schemes where carrier phase and symbol timing acquisition needs to be performed on each transmission burst. The loop architecture described in this paper incorporates coarse frequency and symbol timing offset estimators in a natural way, however the discussion is limited to tracking mechanisms and assumes acquisition of timing and carrier frequencies has been established.

Carrier phase synchronization before matched filtering can be corrupted by out-of-band noise and adjacent-channel interference, however matched filtering before decimating can consume valuable processing resources. This motivates an investigation on alternative techniques that combine the two into a single processing block with independent control loops. Common concerns issued to systems operating with two feedback loops can be avoided by realizing that the timing recovery algorithm operates independently of the input phase, so long as the input carrier frequency offset remains sufficiently small. Issues with carrier loop perturbations due to tracking before timing synchronization is achieved is also addressed.

The system was implemented in C++ with minimal third-party software dependencies. Over-the-air testing was performed using Virginia Tech's open-source SCA implementation [2] by wrapping the multi-rate synchronizer DSP inside a set of SCA components.

This paper is organized as follows: Section II gives a brief theoretical background on multi-rate synchronization and discusses the efficacy of using polyphase filterbanks as a mechanism for calculating interpolants; section III discusses the proposed loop control architecture for combining timing and carrier phase synchronization; section IV describes the simulations used in evaluating the system; section V gives notes on its implementation in an SCA platform; Finally, section VI gives some brief concluding remarks.

II. Multi-rate Synchronization Theory

This section is an abbreviated version of a more complete discussion of interpolation in digital-sampled receivers, available in [1] and [3]. We direct the interested reader to these articles for a more thorough investigation on the topic.

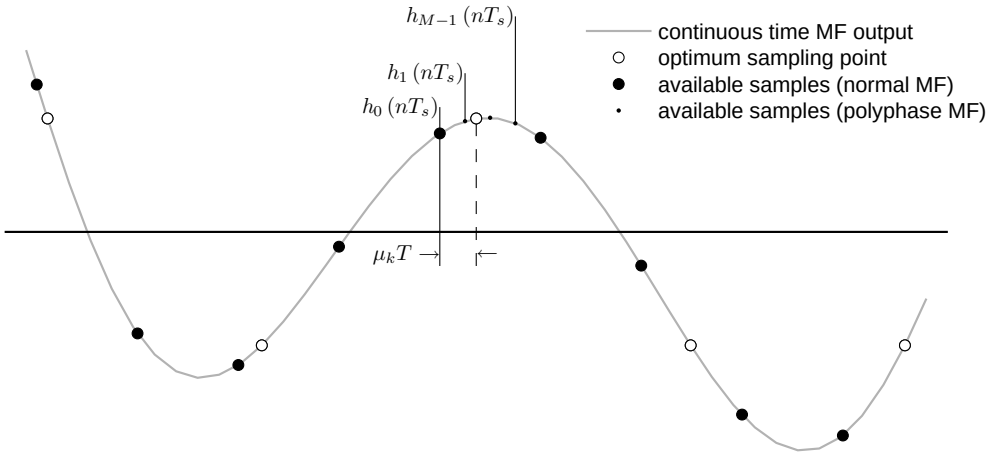

Fig. 1. Matched filter output $y(t)$ and the relationship between available sample points, optimum sample points, and interpolants. In this example the sample rate T_s is approximately twice the symbol rate, however the position of optimum timing slides to the right of the available sample points as time progresses indicating that the actual sampling frequency is slightly greater than 2 samples/symbol. Note that the bank consists of $M = 4$ filters. While none of the samples lie directly on the optimum sampling point, the resolution can be set sufficiently small by increasing M.

A. Continuous and Discrete Time Representations

Let the transmitted analog signal $s(t)$ be the sum of M-ary complex-valued symbols filtered by a pulse shape $p(t)$ spanning $2L$ symbols, each separated by a unit time T. The received signal is assumed to be a delayed version of $s(t)$ corrupted by the addition of white Gaussian noise, $n(t)$, viz.

$$r(t) = s(t - \tau) + n(t) \tag{1}$$

The optimum receiver uses the output $y(t)$ of a matched filter whose impluse response is $h(t) = p(-t)$.

In practice, $r(t)$ is sampled using an analog-to-digital converter (ADC) before filtering in the discrete-time domain. The discretized version of the received signal at a uniform sampling interval T_s is $r(nTs)$ where T/T_s is assumed to be irrational.[1] The received signal samples are applied to an interpolator that computes interpolants $y(kT_i)$ at a reduced rate $T_i = T/N$. The synchronizer must adjust the interval T_i to match the rate at which data symbols are generated by the transmitter. The new samples for an interpolating filter with an impulse response $h_I(t)$ over the timespan $I_1 \leq i \leq I_2$ are given by [3, (6)],

$$y(kT_i) = \sum_{i=I_1}^{I_2} r\left[(m_k - i)T_s\right] h_I\left[(i + \mu_k)T_s\right] \tag{2}$$

where $m_k = \lfloor kT_i/T_s \rfloor$ is the basepoint index and $\mu_k = kT_i/T_s - m_k$ is the irrational fractional interval [3]. Because T_i/T_s is likely irrational, μ_k will change for each interpolant and can take on an infinite number of values. Figure 1 demonstrates the relationship between optimum and available sampling points.

B. Polyphase Filterbanks as a Mechanism for Resampling

By using a polyphase decomposition of a sampled version of the matched filter, a polyphase filterbank can be used to calculate the interpolants in (2). The discrete impulse response of the m-th filter in a bank of M filters is given by [1, Eq. (15)]

$$h_m(nT_s) = h\left(nT_s + \frac{m}{M}T_s\right) \tag{3}$$

The goal of the synchronizer is to choose the filterbank at index m such that the fractional portion $\frac{m}{M}T_s$ is as close to the true timing offset μ_k as possible. As demonstrated in Figure 1, none of the interpolants from the filterbank lie on the optimum output, however increasing the filterbank size can ensure a sufficiently fine resolution.

III. Loop Architecture

Harris and Rice describe three possible loop control architectures in [1], the main differences being the rate at which the loop operates. Keeping the goal of minimizing processing complexity in mind, we chose the architecture operating at 1 sample/symbol. Our adopted system block diagram can be seen in Figure 2. In burst-mode transmission schemes, timing and carrier phase acquisition can occur on several pilot symbols in the frame header.

A. Symbol Timing Tracking Loop

The timing loop consists of a dual pair of matched filter (MF) and derivative matched filter (dMF) polyphase filterbanks which produce an output once every symbol period.[2] The resulting outputs of the MF and dMF are fed into a timing error generator, discussed below. The error signal is filtered with a second-order low-pass loop filter $G(z)$ before feeding the filterbank control. Auxiliary control via the "shift/skip/stuff" mechanism described in [4] is provided to accommodate the incommensurate relationship between the data and sample clocks.

[1]This will hold true for all communications links whose transceiver hardware ADCs are driven by independent clock sources.

[2]Two of each MF and dMF banks are necessary as the in-phase and quadrature signals are processed independently.

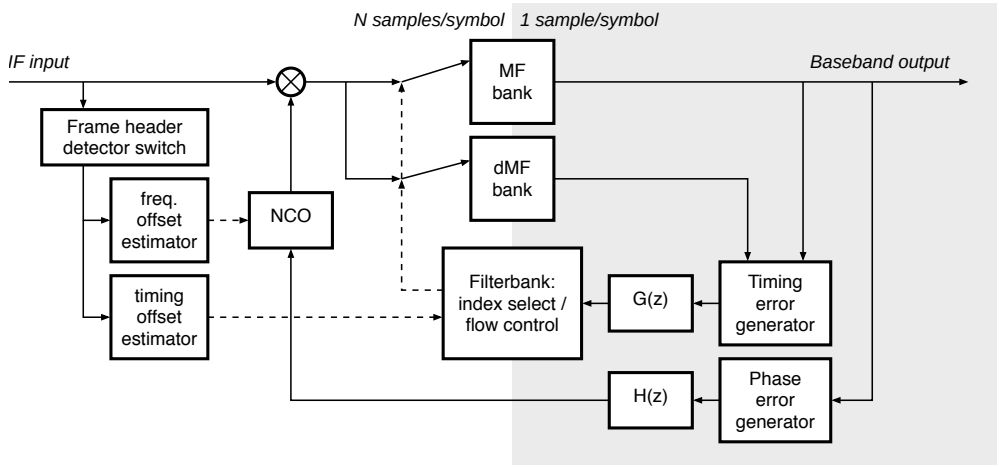

Fig. 2. System block diagram operating at 1 sample/symbol

This system bases its timing error generator on the maximum likelihood timing error detector [5] which relies on the response of the derivative to the matched filter. The ML estimator, therefore, relies on sampling the derivative of the matched filter impulse response,

$$\dot{h}_m(z) = \frac{\partial h_m(z)}{\partial z},\ m \in [0, M-1] \tag{4}$$

Mengali provides an efficient means for approximating $\dot{h}_m(z)$ from the MF coefficients [5, (7.4.57)], also utilized by [1], viz.

$$\dot{h}_m(z) \approx h_{m+1}(z) - h_{m-1}(z), \quad m \in [1, M-2] \tag{5}$$

$$\dot{h}_0(z) \approx h_1(z) - h_{M-1}(z) \tag{6}$$

$$\dot{h}_{M-1}(z) \approx h_{M-2}(z) - h_0(z) \tag{7}$$

This structure allows for only two polyphase filter stages and can operate at a minimum sampling rate.

The error averaged over many symbols provides feedback for controlling the filterbank index. As demonstrated by the S-curve for the maximum-likelihood timing error detector depicted in [5, Figure 7.13], early sampling results in a negative error, while late sampling results in a positive error. When the filtered error signal is negative, the control loop will choose successively lower filterbank indices, enabling less delay. When the filterbank index pointer underflows, the input and output clocks are aligned by repeating the last input sample. Conversely, a positive error will increase the filterbank index pointer, enabling successively more delay. When the filterbank index pointer overflows, the next input sample is skipped.

B. Carrier Phase Tracking Loop

The carrier loop operates in a similar fashion to the timing phase recovery loop; a generated phase error signal is fed through a second-order low-pass loop filter $H(z)$ that drives an NCO. One of the driving motivations for combining carrier phase recovery with symbol timing is that matched filtering can be performed before generating the phase error, thus reducing out-of-band interference and noise applied to the NCO.

Several possibilities exist for generating phase error signals where timing information is known. For systems operating at 1 sample/symbol, perhaps the most obvious solution is a decision-directed loop when knowledge of the modulation scheme utilized is known. For M-PSK signals, a Costas Loop-like structure can be used [6]. For M-QASK signals, decision-based boundaries can be used to direct the phase error generator.

IV. Simulation Results

For all simulations the received signal is generated at 2 samples/symbol and without the addition of noise. The timing loop filter and carrier loop filter operate with a noise bandwidth of approximately 0.5% of the symbol rate and 1% of the sampling frequency, respectively. A set of 32-stage polyphase filterbanks are used in the timing loop for matched filtering. The timing error generator is an approximation to the maximum-likelihood error detector. Table I lists additional parameters used in the simulations.

A. Simulation #1: Symbol Timing Phase Step Response

The first simulation demonstrates the response of the synchronizer when a timing step offset of $T/4$ is introduced into the system. The results can be found in Figure 3.

One very important thing to note is that the carrier phase recovery loop depends on the accurate knowledge of symbol timing. Due to the initial error in symbol timing, the phase error detector results in a non-zero NCO control value in the carrier recovery loop, despite there being no initial carrier phase error in the simulation. Although the carrier loop recovers quickly, the system could make use of a timing lock mechanism to disable the carrier phase control loop before timing synchronization is achieved.

TABLE I
SIMULATION PARAMETERS

Parameter	**Simulation #1**	**Simulation #2**	**Simulation #3**
Modulation scheme	8-PSK	16-QASK	QPSK
Sampling phase offset	$0.25T_s$	$0.1T_s$	0
Sampling frequency offset	0	0	$0.004/T_s$
Carrier phase offset	0	$\pi/5$	0
Carrier frequency offset	0	0	$0.005/T_s$

B. Simulation #2: Timing & Carrier Phase Step Response

The second simulation introduces both carrier and a timing phase offsets under a 16-QASK modulation scheme. The results can be found in Figure 4. Notice how the NCO control settles on the phase offset, $\pi/5$, introduced into the system only after timing synchronization is achieved.

C. Simulation #3: Timing & Carrier Frequency Step Response

Figure 5 depicts the results of a simulation in which both a timing and a carrier frequency offset are introduced. The sampling discrepency is 0.4% of the sampling frequency; one sample out of every 250. The sample clock period T_s is slightly greater than $2T$, and as a consequence the loop control slides through the filterbank by decreasing its index, resulting in progressivly more delay to match the data rate. Notice that the timing control makes one complete sweep through the filterbank approximately every 250 symbols. Additionally, the carrier loop tracks to the frequency offset by linearly increasing the NCO control.

Because the carrier loop, which operates at 1 sample/symbol, depends so highly upon the accuracy of the matched filter, acquisition of timing is critical to this system.

V. SOFTWARE IMPLEMENTATION

In order to demonstrate the ability for this system to operate in an actual wireless environment, the synchronizer was implemented as a set of open-source C++ libraries and wrapped inside a software-defined radio.

The Software Communications Architecture (SCA), developed by the JTRS program of the US Department of Defense, is an open architecture aimed to alleviate interoperability issues in SDR technology. It emphasizes modular software reuse in radio applications by establishing a component-based architecture. Many development tools have been released to facilitate the implementation of SDRs because of the widespread application and adaption of the SCA. We utilized the Open Source SCA Implementation::Embedded (OSSIE), a freely available implementation of the JTRS SCA Core Framework developed at Virginia Tech, for testing our synchronizer in an SDR platform [2].

The filterbank libraries were wrapped inside SCA components and tested within a simple wireless application. The component was successful in recovering carrier phase and symbol timing information before passing the symbols to the

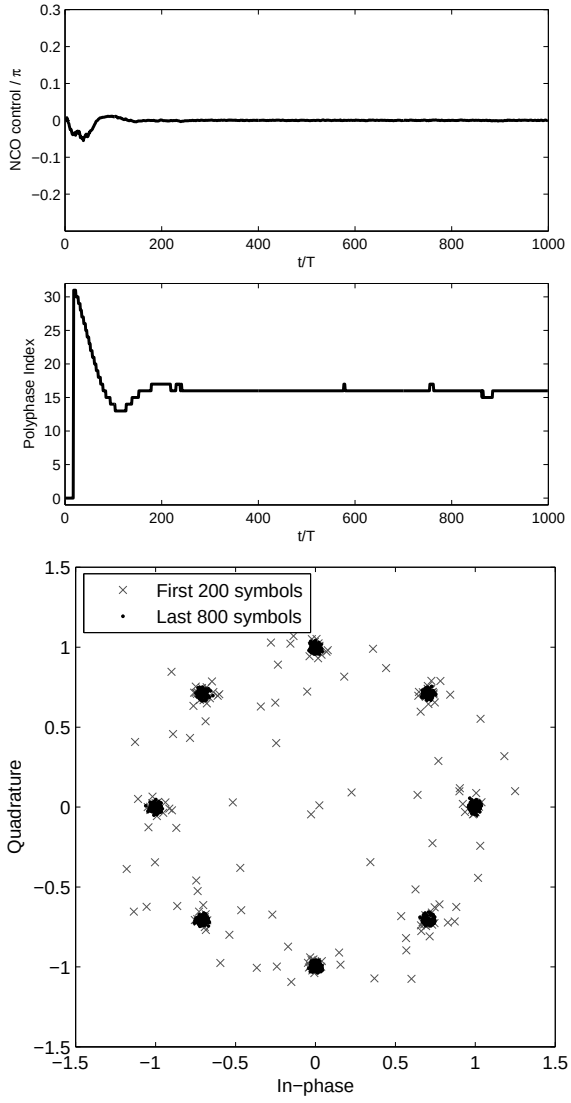

Fig. 3. Simulation #1 results: symbol timing phase step response

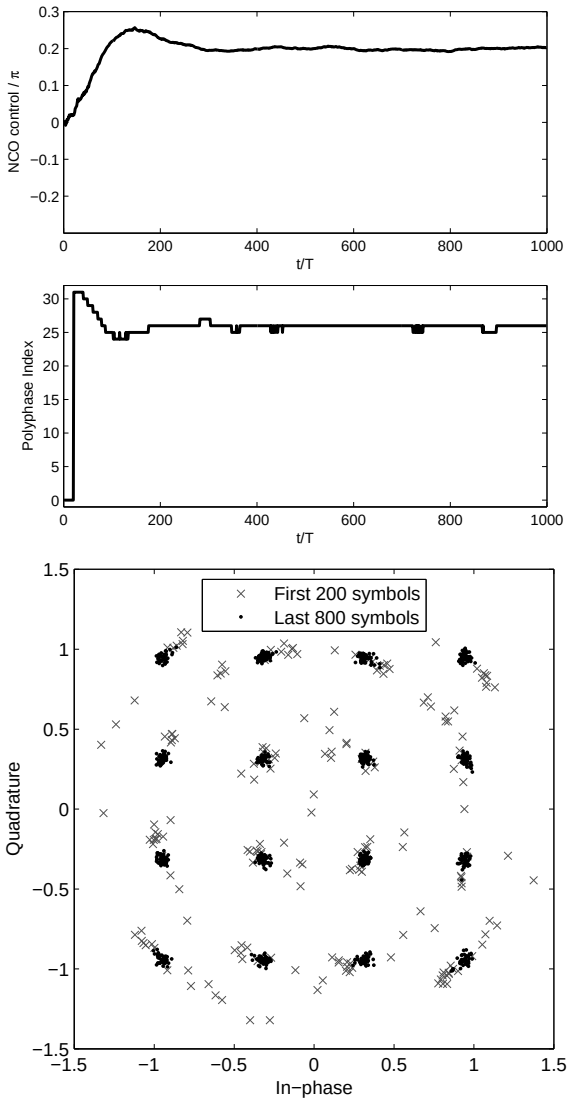

Fig. 4. Simulation #2 results: carrier & timing phase step response

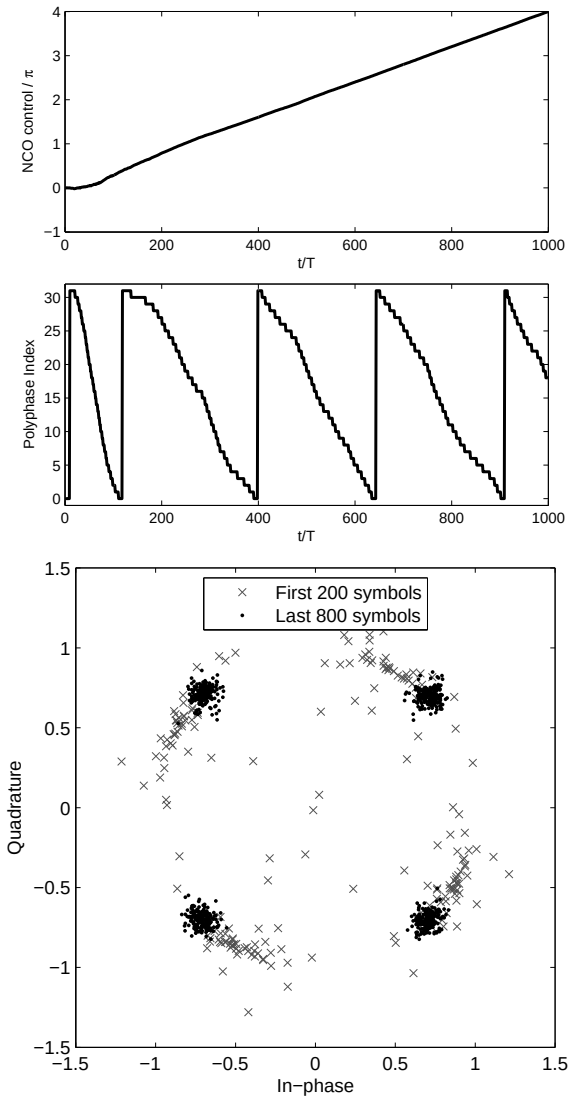

Fig. 5. Simulation #3 results: symbol timing and carrier frequency step response

demodulator. The synchronizer allowed for coherent over-the-air reception of M-PSK and M-QASK modulation schemes.

VI. Conclusions

In this paper we have augmented the work presented in [1] by extending the synchronizer to incorporate a phase recovery tracking loop operating at 1 sample/symbol and demonstrate its ability to synchronize to a number of modulation schemes. The use of polyphase filterbanks as interpolators provides several advantages for low-complexity receivers operating at a minimal sample rate. Integrating a carrier tracking loop to the system provides a tractable solution to burst-mode transmission schemes where periodic acquisition of carrier frequency and symbol timing information is necessary. Three simulations demonstrating the ability for the synchronizer to track to various phase and frequency offsets under different modulation schemes were provided. Additionally an implementation of the synchronizer using an open source implementation of the SCA for SDR platforms was tested.

References

[1] F. J. Harris and M. Rice, "Multirate Digital Filters for Symbol Timing Synchronization in Software Defined Radios," *IEEE Journal on Selected Areas of Communications*, vol. 19, no. 12, pp. 2346–2357, December 2001.

[2] [Online]. Available: http://ossie.wireless.vt.edu/

[3] F. M. Gardner, "Interpolation in Digital Modems–Part I: Fundamentals," *IEEE Transactions on Communications*, vol. 41, no. 3, pp. 501–507, March 1993.

[4] M. Rice and fred harris, "Loop Control Architectures for Symbol Timing Synchronization in Sampled Data Receivers," in *MILCOMM Proceedings*, vol. 2, October 2002, pp. 987–991.

[5] U. Mengali and A. N. D'Andrea, *Synchronization Techniques for Digital Receivers (Applications of Communications Theory)*, 1st ed. Springer, 1997.

[6] H. C. Osborne, "A Generalized "Polarity-Type" Costas Loop for Tracking MPSK Signals," *IEEE Transactions on Communications*, vol. COM-30, no. 10, pp. 2289–2296, October 1982.