Digital Receiver Design

Ke Tang
16823384

[Introduction]:

The conventional analog radio receiver has been in use for nearly a century. With increasing development of communication science, developing digitalized receiver for stable and reliable communication is more and more important to reduce complexity in hardware. In earlier period, digital filter is constrained by performance of ADC so that digitalization can be only implemented after orthogonal demodulation which causes high phase error. To offer significant benefits and reliability in performance, density and cost, more advanced discrete-time architecture can be applied to design the digital receiver in components such as carrier phase synchronizer and symbol timing synchronizer. In this report, several discrete-time techniques are used in different components and Matlab simulation will indicate that digital receiver using purely discrete-time processing is feasible and reliable for modern hardware implementation.

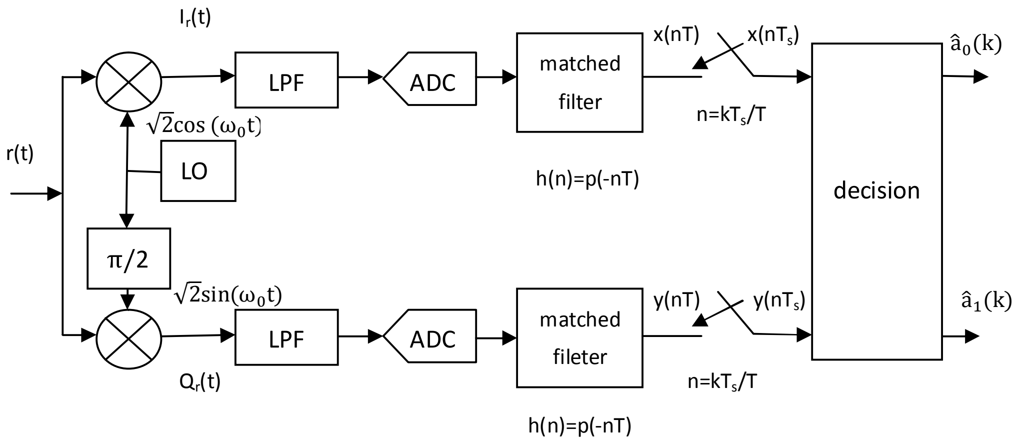
[Design details]:

Normally, a digital receiver must have an Analog-to-Digital Converter (ADC) component sampling the continuous signal to get discrete signal, a discrete-time matched filter used to get signal stream and a determiner to decide which point is closest to the signal space projection in the constellations.

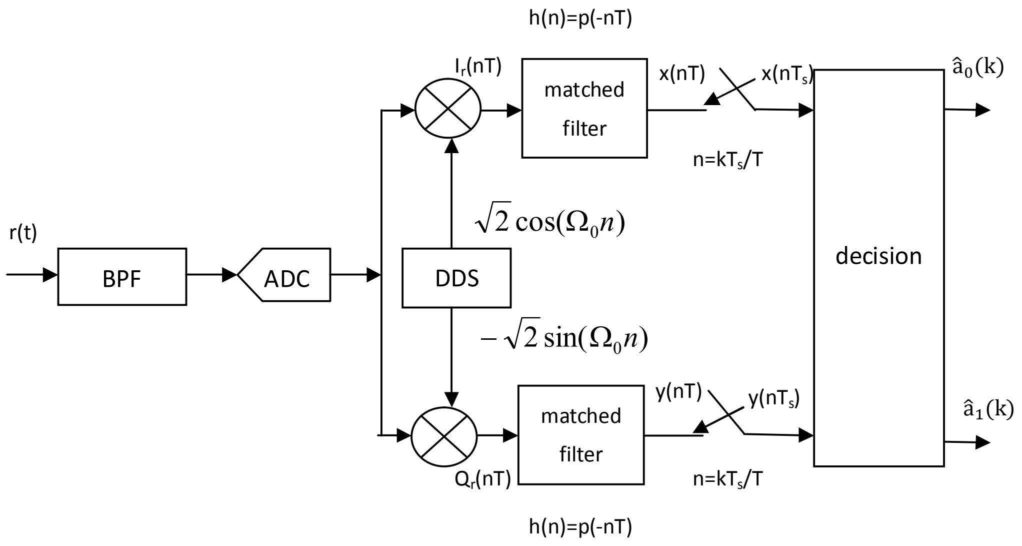
1. ADC placement:
To produce samples for discrete-time processing, ADC component should be pushed as more front as possible to include fewer analog components. There are two basic options for ADC placement: I/Q sampling and IF sampling.

I/Q sampling first performs the quadrature mixing in the continuous-time domain using a pair of mixers and quadrature sinusoids generated from a local oscillator. Then the outputs of the mixers are low-pass filtered to remove double-frequency components and sampled by a pair of ADCs before discrete-time matched filter. The advantage of this method is that ADC can operate at a relatively low clock rate and can achieve higher quantization precision. But manufacturing tolerances limits the accuracy of the parameter value of the components used to construct the continuous-time circuits. Imperfection in the mixers may cause DC offsets, local oscillator leakage may be radiated from the antenna and it’s impossible to achieve preserve proper amplitude and phase relation between I/O components.

However, IF sampling which samples IF signal by a single ADC can solve most of the problems. It makes the generation of quadrature sinusoids and the corresponding mixing operation entirely in the discrete-time domain but ADC must operate at a much higher clock rate so that quantization precision is lower and much signal processing must operate at a higher clock rate. In this design, I will use the second approach and illustration of these two method are displayed as follows:



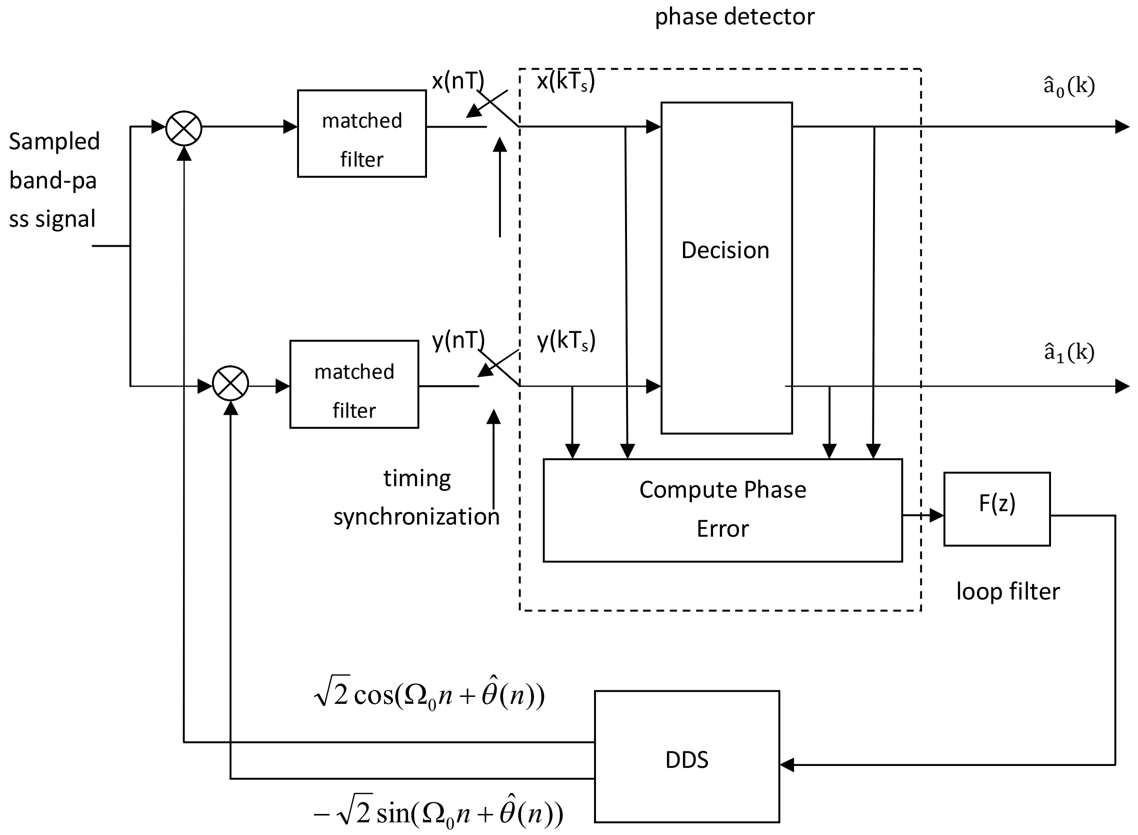
I/Q baseband sampling



IF band-pass sampling

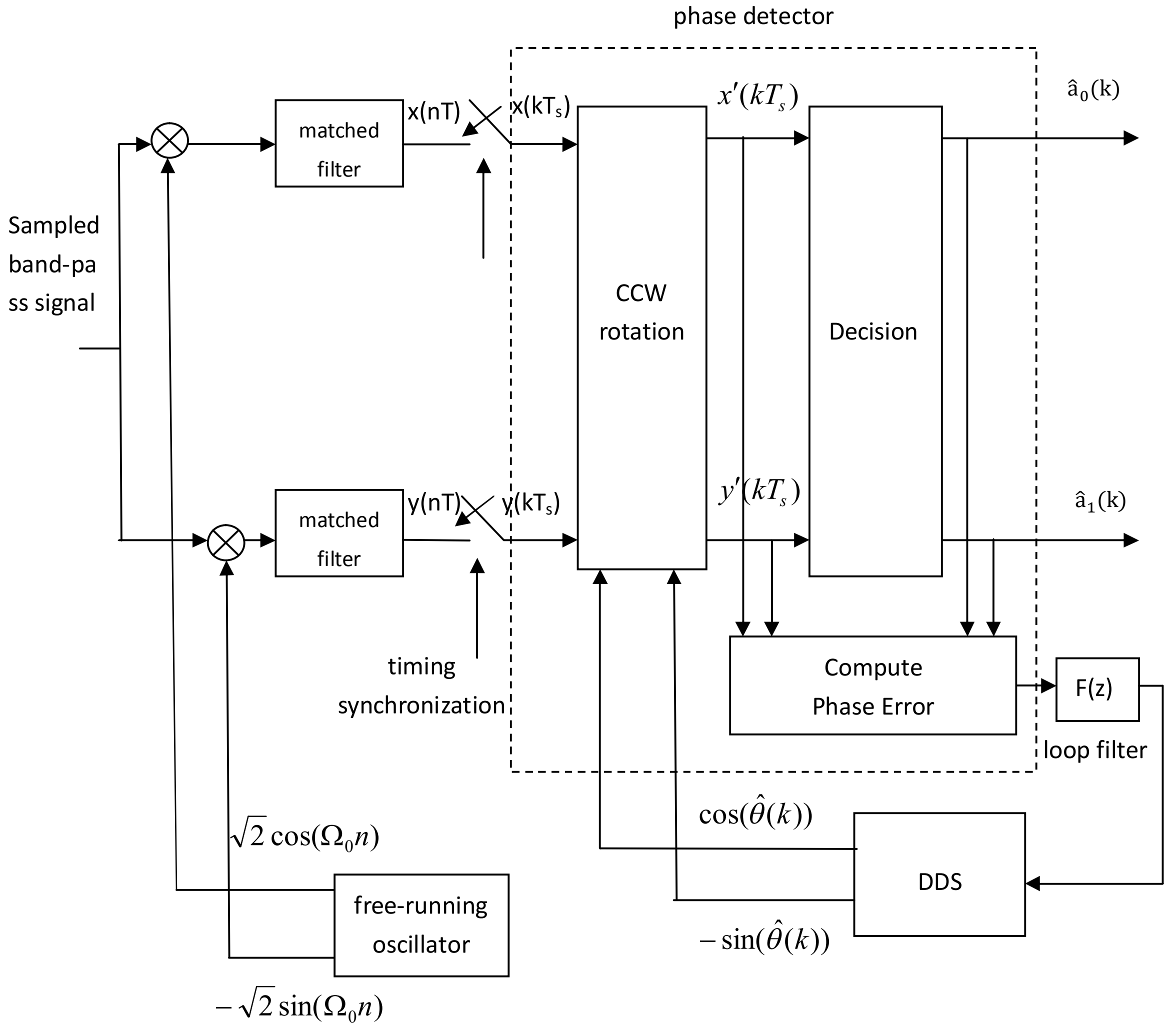
2. Quantization: A real ADC has finite precision amplitude and is composed a sample-and-hold component and a quantizer. A uniform quantizer with b bits partitions the dynamic range $2X_m$ into 2b discrete levels. The width of each level is $\Delta = \frac{2X_m}{2^b} = \frac{Xm}{2^{b-1}}$ and the quantized output is $x_q(nT) = k\Delta X_m$. Thus, the error signal between original signal and quantized signal is $e(n) = x(nT) - x_q(nT)$ and can be treated as quantization noise. When the discrete-time levels increase, the quantization precision and the signal-to-quantization-noise ratio will increase and the error signal will be reduced. But raising discrete-time levels will also result in more complex hardware. Consequently, the tradeoff must be considered between quantization precision and complexity of implementation.

3. Carrier Phase Synchronization:
The role of carrier phase synchronization is to estimate the carrier phase in order to reduce carrier phase error because a carrier phase error will cause a rotation in the signal space projection which may affect the decision. Basically, the synchronizer is based on the phase-locked loop or PLL and forces the local oscillators in the detector to oscillate in both frequency and phase with the carrier oscillator used at the transmitter. One approach mimics continuous-time processing that adjusts the phase of the quadrature sinusoids used to translate the received samples from band-pass to I/Q baseband. The error signal is the difference between signal space projections and the data symbols (data estimates) and the PLL locks when the error signal is forced to zero. However, this is a multi-rate system. The phase detector operates at 1 sample/symbol whereas the DDS operates at N samples/symbol. As a result, an upsample operation must be inserted which is not desirable.



Carrier phase synchronization using phase adjusted quadrature sinusoids

The other approach uses quadrature sinusoids with a fixed frequency and phase to perform the translation from band-pass to I/Q baseband. Carrier phase offset compensation is performed by rotating the downsampled matched filter outputs to remove rotation due to carrier phase offset. The carrier phase PLL is similar to the previous one and locks when the phase error generated by de-rotated signal space projections and data symbols (or data symbol estimates) is forced to zero. Because the sampled matched filter outputs form a discrete-time sequence, this approach is a purely discrete-time approach which is what we anticipate. In addition, the PLL operates at 1 sample/symbol rather than N samples/symbol in the first approach.

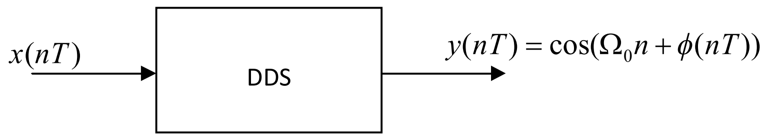


Carrier phase synchronization using a post-matched filter de-rotation operation

If the phase error detector uses the symbol estimates to compute the phase error, the resulting PLL is a decision-directed loop. Alternatively, if the phase error is computed using knowledge of transmitted data symbols, the PLL is a data-aided PLL. In this receiver design, I will use decision-directed PLL.

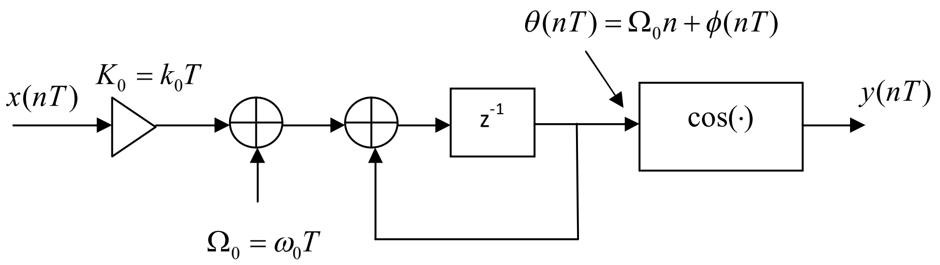
DDS is short for Direct Digital Synthesizer is a practical method for producing a reference frequency whenever extremely precise frequency resolution and fast clocking speed are required. For continuous-time sinusoid $\cos(\omega_0 + \phi(t))$ with time-varying $\phi(t)$, the instantaneous frequency = $\omega_0 + \frac{d}{dt}\phi(t)$ rad/s so that frequency is the time derivative of the phase. The instantaneous phase is denoted as $\theta(t) = \omega_0 + \phi(t)$ and is used for a continuous-time voltage controlled oscillator (VCO) which uses input x(t) to compute the instantaneous excess frequency and produces sinusoidal output y(t). The relationship between instantaneous excess phase $\phi(t)$

and the input x(t) is $\phi(t) = k_0 \int_{-\infty}^{t} x(\tau) d\tau$. As for T-spaced samples of input x(nT), the samples of output will be $y(nT) = \cos(\omega_0 nT + \phi nT))$ and the instantaneous excess phase will be $\phi(nT) = k_0 T \sum_{k=-\infty}^{n-1} x(kT)$. The instantaneous frequency of discrete-time sinusoid is changed by adjusting the amplitude of the samples x(nT). The block diagram is as follows:



$$\phi(nT) = k_0 T \sum_{k=-\infty}^{n-1} x(kT)$$

Block diagram showing the input/output relationship



a conceptual block diagram emphasizing the signal processing

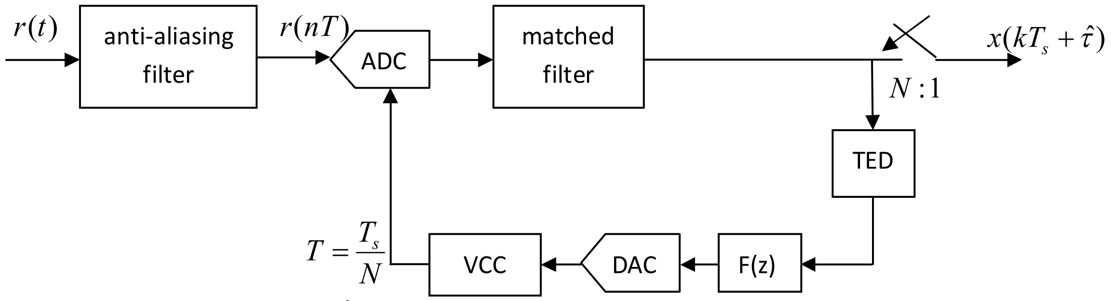
For the CCW rotation component, the sampled matched filter outputs are de-rotated by the estimated carrier phase offset $\hat{\theta}(k)$. The DDS provides carrier phase estimate as $\cos(\hat{\theta}(k))$ and $-\sin(\hat{\theta}(k))$ to produce the de-rotated signal space projection. The error signal is proportional to the uncompensated phase error $\theta - \hat{\theta}(k)$. Thus, the loop locks when $\hat{\theta}(k)$ equals to θ.

The phase error detector can be implemented in two ways: heuristic phase error detector and maximum likelihood phase error detector. The heuristic phase error detector directly computes the difference between the phase angle of the de-rotated matched filter outputs and the phase angle of the transmitted constellation point, while maximum likelihood phase error detector selects a phase estimate that minimizes the energy in the phase error and do not need such arctangent operations.

5. Symbol Timing Synchronization:

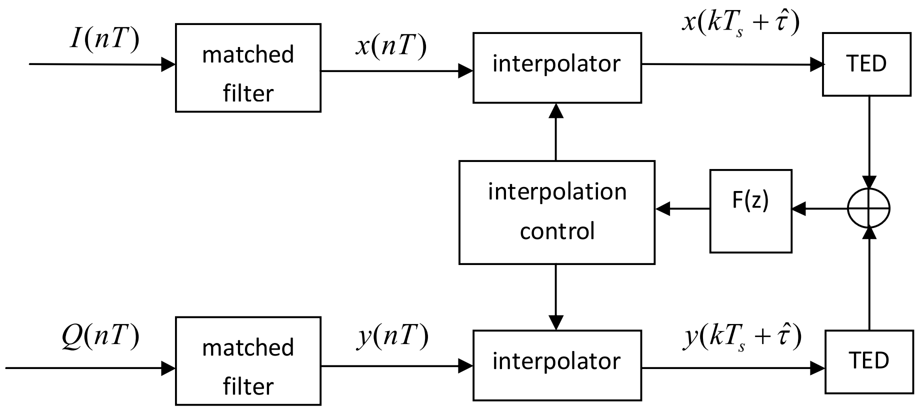
Symbol timing synchronization is the process of estimating a clock signal that is aligned in both phase and frequency with the clock used to generate the data at the transmitter. The clock must be extracted from the noisy received waveforms that carry the data and the clock signal is used to identify the instants when the matched filter output should be sampled. A timing error will cause the signal space projections scattered and then affect the decision more significantly when they are closer to their boundaries.

In discrete-time detector, the goal of symbol timing is to produce N samples at the matched filter outputs during each symbol interval such that one of the samples is aligned with the maximum eye opening. Two basic approaches can be used to solve this problem. The first one, similar to continuous-time detector, uses the timing error to adjust the phase of the voltage controlled clock (VCC) that triggers the ADC.



A hybrid continuous-time/discrete-time approach to symbol timing synchronization

Although this approach can produce samples that are aligned in both phase and frequency with the data clock, it needs a feedback path to the continuous-time parts having hardware overhead of transferring from digital to analog domain via DAC and analog filter. Moreover, VCC will contribute higher levels of phase noise. Thus, the second approach, which estimates the time delay solely from the asynchronous samples at the output of the matched filter, can address those issues. For this approach, role of the symbol timing synchronization becomes to move the samples to the desired time instants, called interpolation. This type of timing synchronization is performed using a discrete-time PLL composed of timing delay detector (TED), loop filter and interpolation control. The architecture is as follows:



A discrete-time approach to symbol timing synchronization

The TEDs produce an error signal once every symbol based on the current timing estimate and the discrete-time error signal is updated at the symbol rate. The output of the TED is a function of the interpolated matched filter outputs and the data symbol (or their estimates). There are many types of TEDs such as Maximum Likelihood Timing Error Dector(MLTED), Early-Late Timing Error Detector(ELTED), Zero-Crossing Timing Error Dector(ZCTED) and Gardner Timing Error Detector(GTED). In this design, I will use MLTED, which uses the sign-corrected slope of the eye diagram for the error signal. The error signal for the decision-directed TED is

$e(k) = \hat{a}_0(k)\dot{x}(kT_s + \hat{\tau}) + \hat{a}_1(k)\dot{y}(kT_s + \hat{\tau})$ and the data symbol estimates are

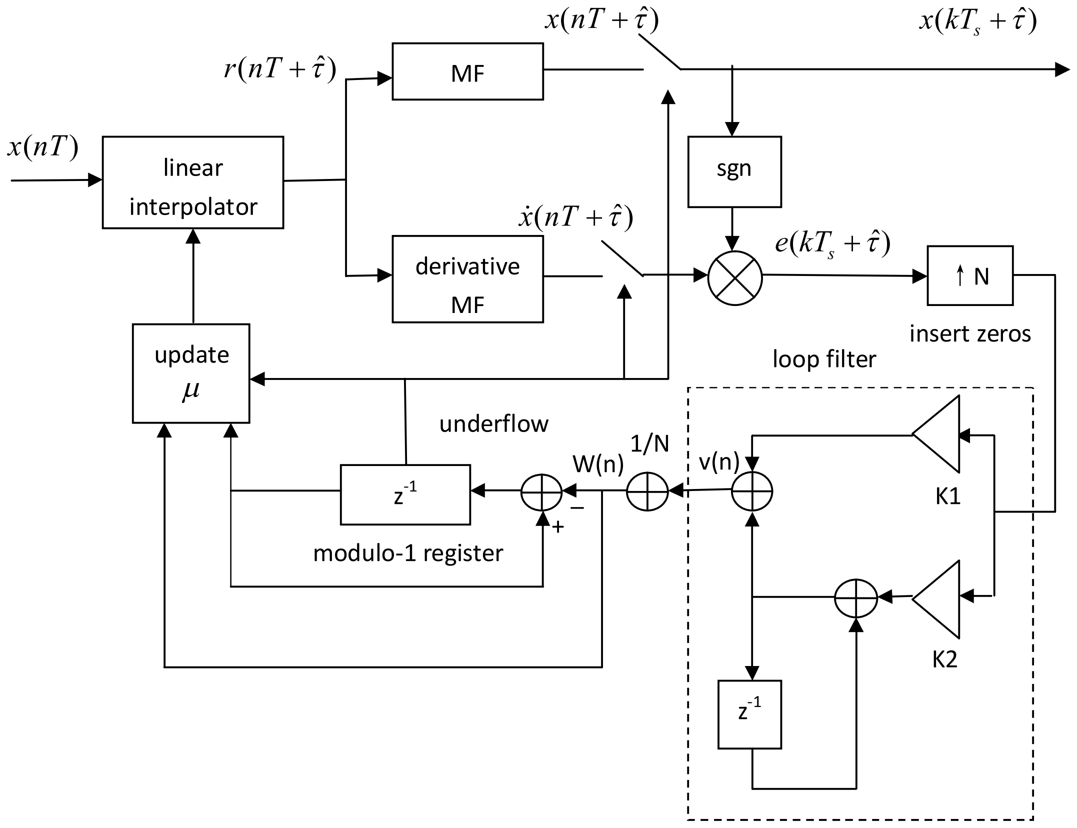
$\hat{a}_0(k) = \operatorname{sgn}\{x(kT_s + \hat{\tau})\}, \hat{a}_1(k) = \operatorname{sgn}\{y(kT_s + \hat{\tau})\}$.

For the interpolation part, the desired interpolant can be obtained by computing a weighted sum of available samples. When the interpolant is between x(nT) and x((n+1)T), the sample index n is the basepoint index m(k) and the fraction of a sample time greater than m(k)T is the fractional interval $\mu(k)$. For asynchronous sampling, the sample clock is independent of data clock used by the transmitter so the sampling instants are not synchronized to the symbol periods. When the symbol timing PLL is in lock and the interpolants are desired once per symbol, $T_I = T_S$. The ideal interpolation filter is IIR but its use poses an often unacceptable computational burden, especially when the fractional interval changes. Thus, in this design, I will use a popular class of FIR interpolating filter, piecewise polynomial interpolation. To simplify this process, the first degree polynomial will be used to approximate the interpolants, which can be denoted as

$x((m(k) + \mu(k))T) = \mu(k)x((m(k)+1)T) + (1-\mu(k))x(m(k)T)$.

The purpose of the interpolator control block is to provide the interpolator with the k-th basepoint index m(k) and the k-th fractional interval $\mu(k)$. Two common used methods for interpolation control are a counter-based method and a recursive method. Here, I will use a counter-based method, Modulo-1 Counter Interpolation Control. For the case where interpolants are required for every N samples, interpolation control can be accomplished using a modulo-1 counter designed to underflow every N samples where the underflows are aligned with the basepoint indexes. The T-spaced samples of matched filter input are clocked into the matched filter with the same clock used to update the counter. A decrementing modulo-1 counter can be used to simplify the computation of the fractional interval. The counter decrements by 1/N on average so that underflows occur every N samples on average. The loop filter output v(n) adjusts the amount by which the counter decrements. This is done to align the underflows with the sample times of the desired interpolant. When operating properly, the modulo-1 counter underflows occur a clock period after the desired interpolant. Then the fractional interval may be computed directly from the contents of the modulo-1 counter on underflow. The counter value satisfies the recursion $\eta(n+1) = (\eta(n) - W(n))$ mod 1, where $W(n) = 1/N + v(n)$ is the counter input. When the decrementing counter underflows, the index n is the basepoint index m(k). Incorporating the modulo-1 reduction produces $\eta(m(k)+1) = 1+\eta(m(k)) - W(m(k))$.

According to the relationship $\frac{\mu(m(k))}{\eta(m(k))} = \frac{1-\mu(m(k))}{1-\eta(m(k)+1)}$, $\mu(m(k)) = \frac{\eta(m(k))}{W(m(k))}$. So, the underflow period of the decrementing modulo-1 counter is $\frac{1}{W(n)} = \frac{N}{1+Nv(n)}$. When in lock, v(n) is zero on average and the decrementing modulo-1 counter underflow period is N samples on average. During acquisition, v(n) adjusts the underflow period to align the underflow events with the symbol boundaries. The architecture of the whole symbol timing synchronization is illustrated as below (only the inphase signal part is displayed):



Symbol timing synchronization for inphase signal using a linear interpolator and an proportional-plus-integrator loop filter

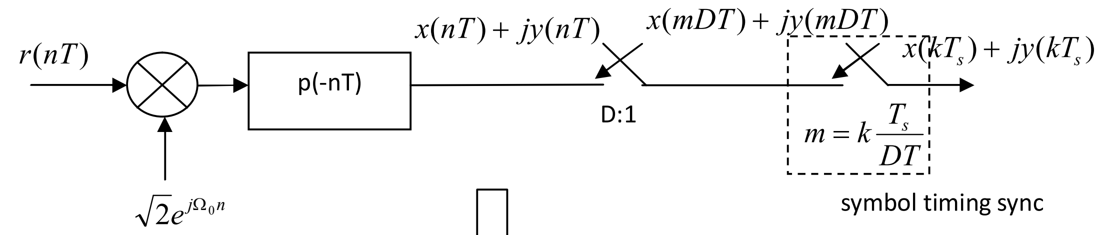
6. Possible improvement:

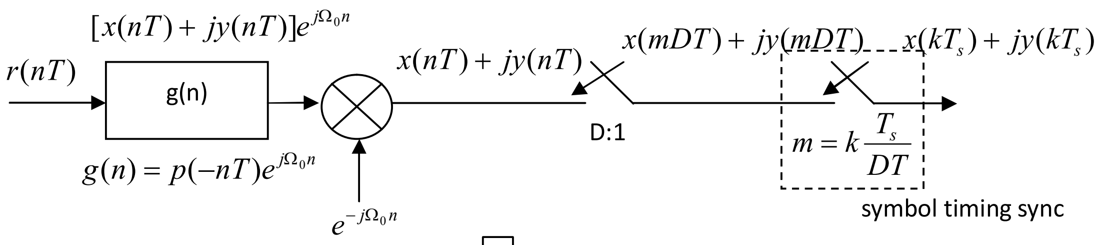
Most receivers are required to select the energy in a specified bandwidth centered around a desired frequency and reject the spectral content outside the desired frequency band. The process to isolate the desired bandwidth centered around the desired carrier frequency is channelization. For discrete-time QAM demodulator, two techniques can be used for channelization. Assume that the desired signal is centered at Ω_0 rad/sample, the first kind, also called superheterodyne receiver, uses multiplication by the complex exponential $e^{-j\Omega_0 n}$ to center the spectrum of the desired signal at baseband and a real-valued low-pass filter $h(n)$ to eliminate the unwanted spectral content. The other kind, also called RTF channelizer, uses a complex-valued band-pass filter with coefficients $g(n) = h(n)e^{\Omega_0 n}$ to eliminate the unwanted spectral content first, followed by multiplication by $e^{-j\Omega_0 n}$ to translate the output to baseband. These techniques are useful for applications where the sampling is applied directly to RF spectrum. However, for many applications, a local oscillator is adjusted to translate the center frequency of the desired signal to a fixed frequency called the intermediate frequency or IF. As this project focuses on IF digital receiver design, it is assumed that some kind of channelization has been already performed.

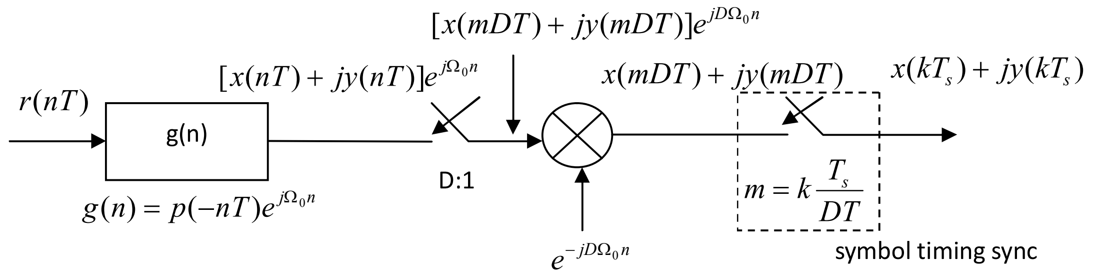
For the demodulator, since we use the architecture of IF sampling, the IF sample rate should be selected to satisfy the sampling theorem, that is, the sample rate is at least twice the highest frequency of the IF signal. If the sample rate is four times the intermediate frequency, that is, $F_s = 4f_0$ samples/s, then $\Omega_0 = 2\pi f_0 T = \frac{2\pi f_0}{F_s} = \frac{\pi}{2}$ so that the quadrature sinusoids assume the 0, ± 1 values and the quadrature frequency translation can use sign-alternations instead of multiplications. Thus, the properties of multirate processing can be used to reduce the complexity of a QAM demodulator.

To simply the design, it is assumed that the IF sample rate aliases the signal to the quarter sample rate and is a multiple of the symbol rate. The sampled IF signal can be represented as $r(nT) = I_r(nT)\sqrt{2}\cos(\Omega_0 n) - Q_r(nT)\sqrt{2}\sin(\Omega_0 n)$, where $\Omega_0 = \pi/2$. With complex spectral shift, $r(nT)\sqrt{2}e^{j\Omega_0 n} = I_r(nT) + jQ_r(nT) + [I_r(nT) - jQ_r(nT)]e^{-j2\Omega_0 n}$. The matched filter will remove the double frequency term and produces the complex-valued baseband output $x(nT) + jy(nT)$. Moreover, the output can be downsampled by D because its bandwidth has been reduced by the matched filter. The resulting downsampled signal is $x(mDT) + jy(mDT)$. The symbol timing synchronizer, operating on $x(mDT) + jy(mDT)$, will produce a sequence at 1 sample/symbol from which the desired signal space projections are produced after rotation by the carrier phase offset. To enable multirate processing, first of all, we have to replace the low-pass matched filter with the complex-valued band-pass filter: $g(n) = p(-nT)\sqrt{2}e^{j\Omega_0 n}$. Thus, the output of band-pass matched filter can be expressed as

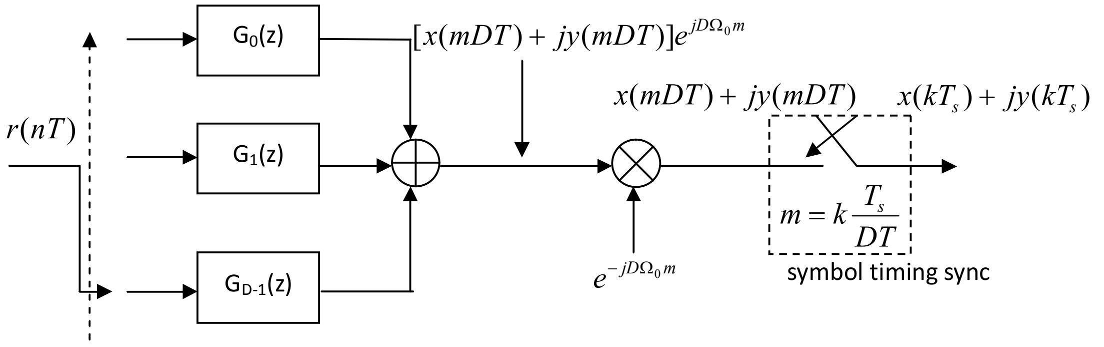
$[x(nT) + jy(nT)]e^{j\Omega_0 n}$. Then, we move the downsample operation forward and place it right at the output of the band-pass filter so that the complex exponential oscillates at a frequency $D\Omega_0$ rad/sample. The development process is shown below:







If $D\Omega_0 \bmod(2\pi) = 0$, the donwnsample operation aliases the band-pass filter output directly to baseband and the final frequency translation is not required. A commonly used choice for D=4. As the structure begins with the filter-downsample operations, an efficient method for performing this task is to use polyphase filterbank.

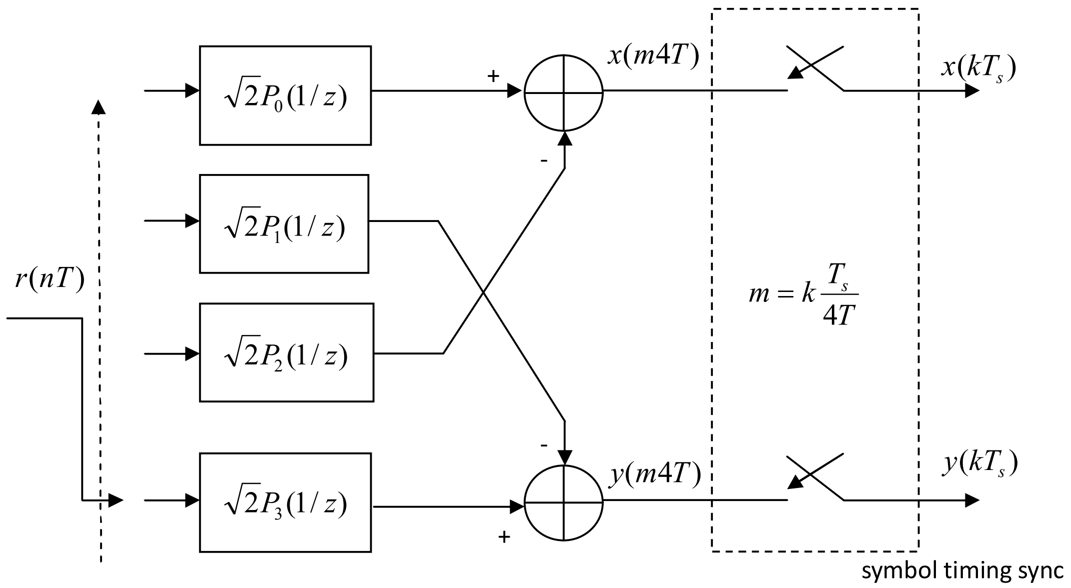


When D=4, the four-stage polyphase partition of G(z) can be simply expressed as

$G(z) = G_0(z^4) + z^{-1}G_1(z4) + z^{-2}G_2(z4) + z^{-3}G_3(z4)$, where

$$\begin{aligned}
G_0(z4) &= \sqrt{2}p(0) + \sqrt{2}p(-4)e^{j4\Omega_0}z^{-4} + \sqrt{2}p(-8)e^{j8\Omega_0}z^{-8} + \cdots \\
&= \sqrt{2}[p(0) + p(-4)z^{-4} + p(-8)z^{-8} + \cdots] \\
&= \sqrt{2}P_0(z^{-4}) \\
G_1(z^4) &= \sqrt{2}p(-1)e^{j\Omega_0} + \sqrt{2}p(-5)e^{j5\Omega_0}z^{-4} + \sqrt{2}p(-9)e^{j9\Omega_0}z^{-8} + \cdots \\
&= j\sqrt{2}[p(-1) + p(-5)z^{-4} + p(-9)z^{-8} + \cdots] \\
&= j\sqrt{2}P_1(z^{-4}) \\
G_2(z4) &= \sqrt{2}p(-2)e^{j2\Omega_0} + \sqrt{2}p(-6)e^{j6\Omega_0}z^{-4} + \sqrt{2}p(-10)e^{j10\Omega_0}z^{-8} + \cdots \\
&= -\sqrt{2}[p(-2) + p(-6)z^{-4} + p(-10)z^{-8} + \cdots] \\
&= -\sqrt{2}P_2(z^{-4}) \\
G_3(z^4) &= \sqrt{2}p(-3)e^{j3\Omega_0} + \sqrt{2}p(-7)e^{j7\Omega_0}z^{-4} + \sqrt{2}p(-11)e^{j11\Omega_0}z^{-8} + \cdots \\
&= -j\sqrt{2}[p(-4) + p(-7)z^{-4} + p(-11)z^{-8} + \cdots] \\
&= -j\sqrt{2}P_3(z^{-4})
\end{aligned}$$

The Noble identities are applied to move the dowsample-by-4 operation to the input side of these filters. Since the input is real, the combination of the outputs of filters $G_0(z^4)$ and $G_2(z^4)$ is $x(4mT)$ whereas the combination of the outputs of filters $G_1(z^4)$ and $G_3(z^4)$ is $y(4mT)$. The final system is as follows:



[Simulation and analysis]:

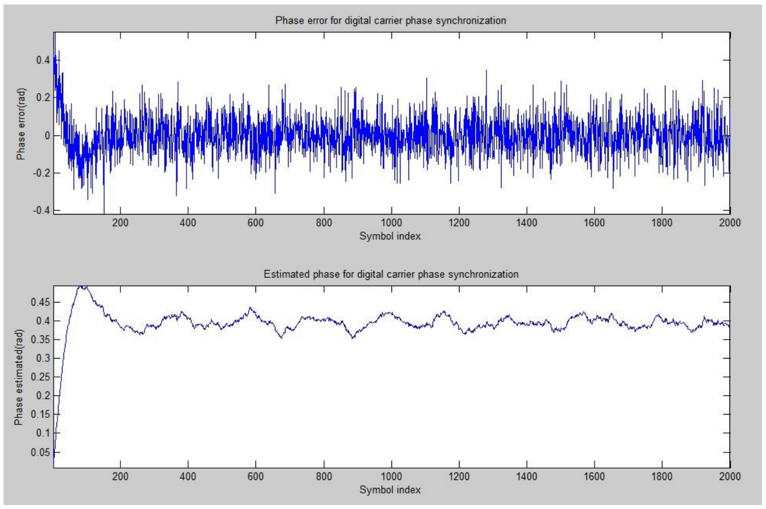
For simulation, the parameters of the system are chosen as follows:

Parameters	Values
Symbol rate	1symbols/s
IF sample rate	40 samples/symbol.
Number of symbols	2000
Modulation method	4QAM
Initial phase offset	22.5^{o}
Carrier frequency	10Hz
Pulse shape	Raised cosine pulse
SNR	5dB
Quantization levels	8

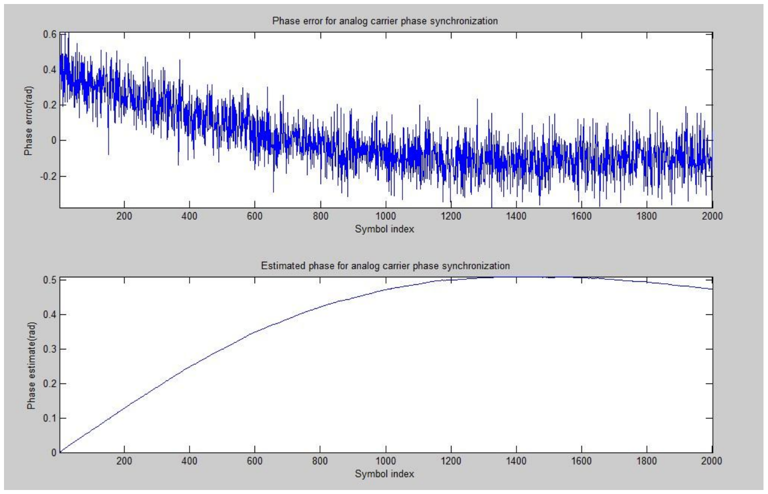
1. Carrier phase synchronization:

The simulation for both analog and digital method is shown below:

Digital carrier phase synchronization



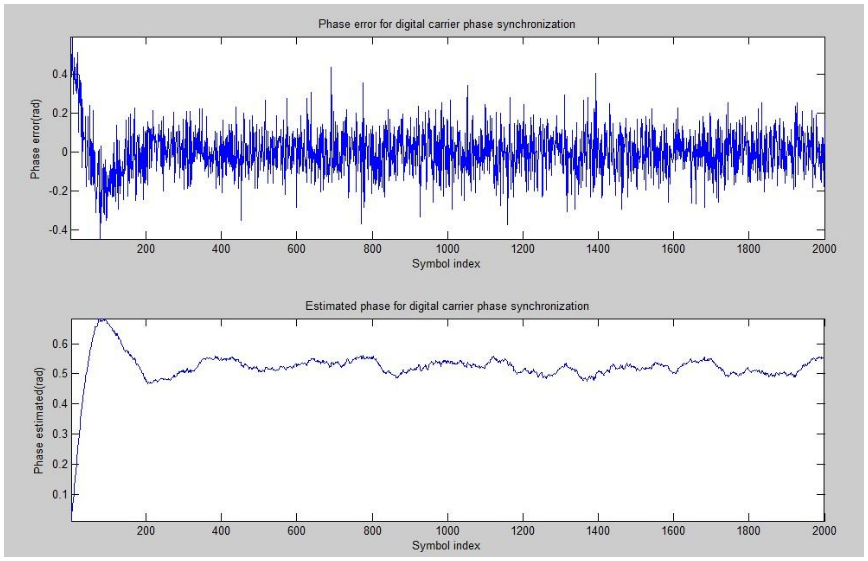
Analog carrier phase synchronization

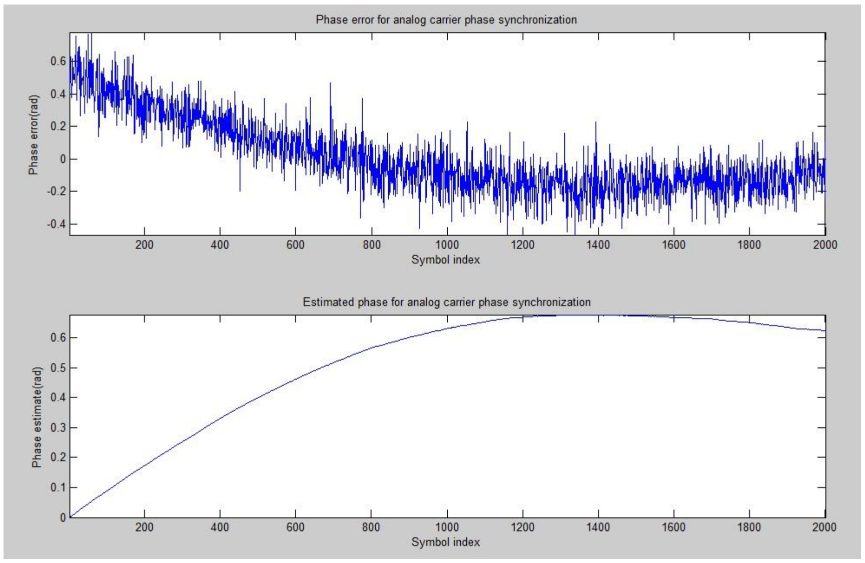


As we can see, the digital method quickly synchronizes the phase without upsampling operation but the estimated phase is not stable while the analog method is a little slower and requires a upsampling component but can get more stable estimated phase. So we should use digital method when we need to fast synchronization in instantaneous communication and use analog method when the accuracy of phase synchronization is required. Both methods use phase error and DDS to generate the estimated reference phase but the difference is that digital method compensates the phase offset to CCW rotation component while analog method compensates the phase offset to the local oscillator for demodulation.

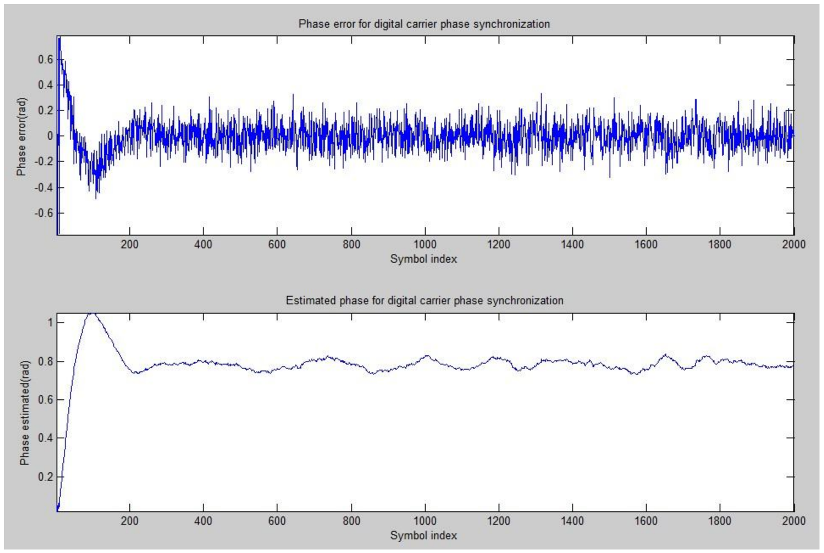
When the initial phase offset increases, the phase error and estimated phase are shown below:

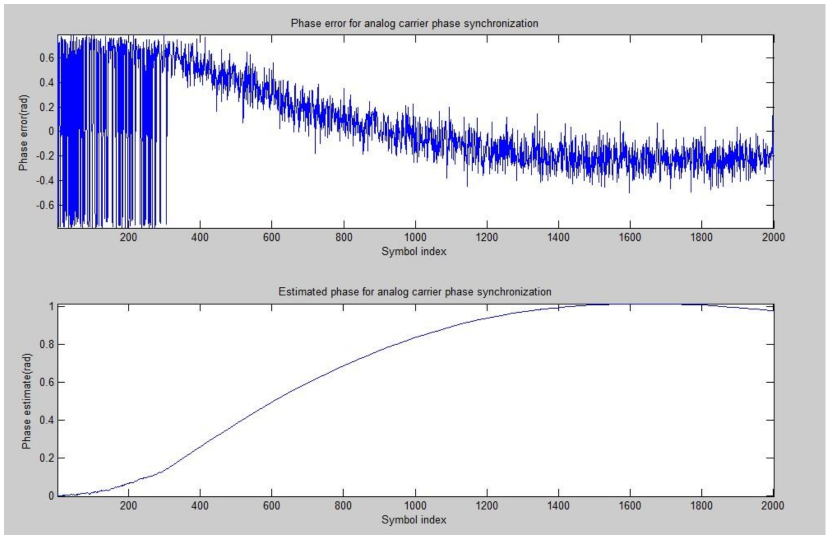
intial phase offset=30^o





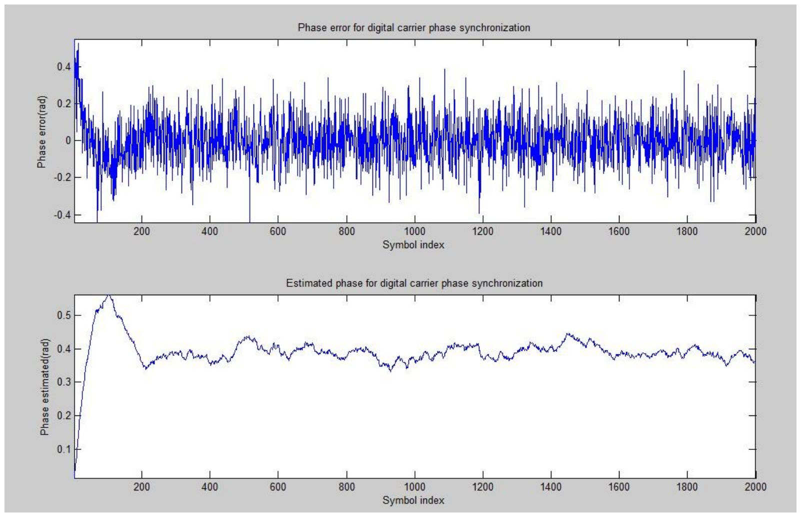
intial phase offset=45^o

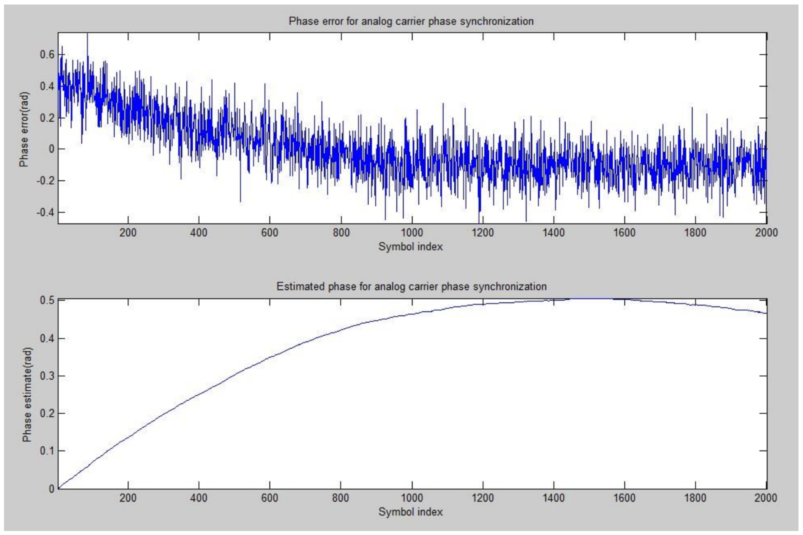




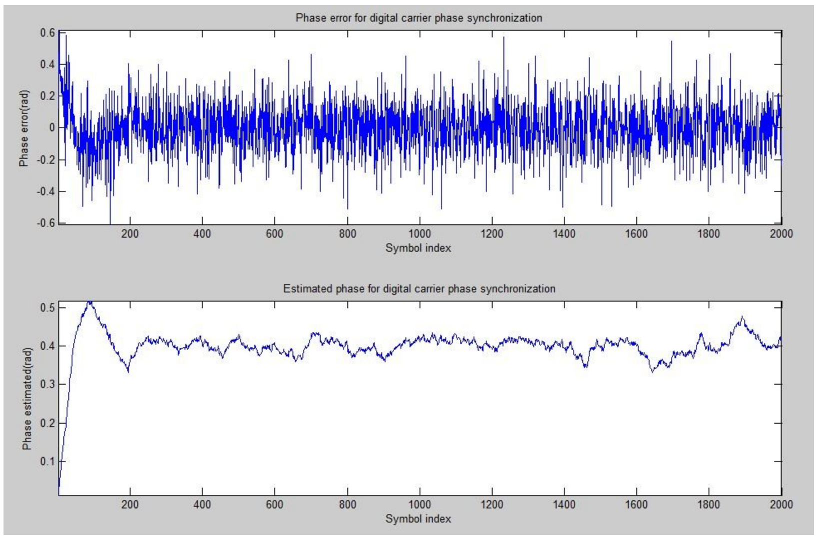
When the SNR decreases, the phase error and estimated phase are shown below:

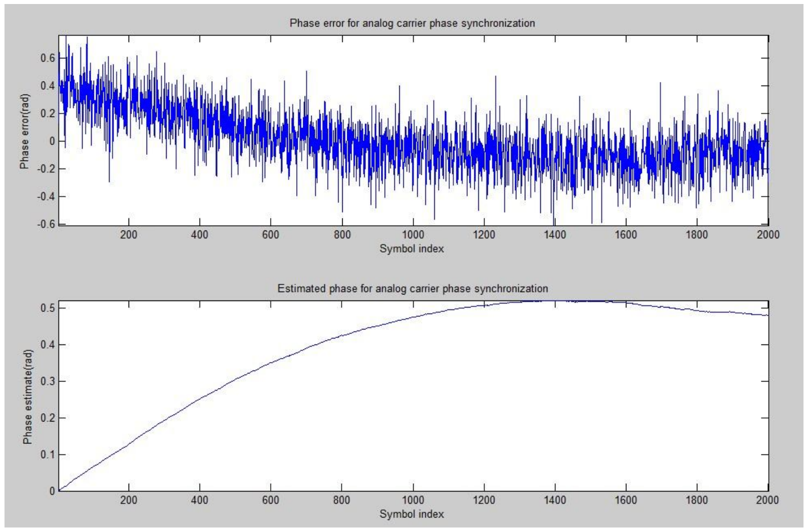
SNR=3dB




Phase error for analog carrier phase synchronization
Phase error(rad)
Symbol index
Estimated phase for analog carrier phase synchronization
Phase estimate(rad)
Symbol index

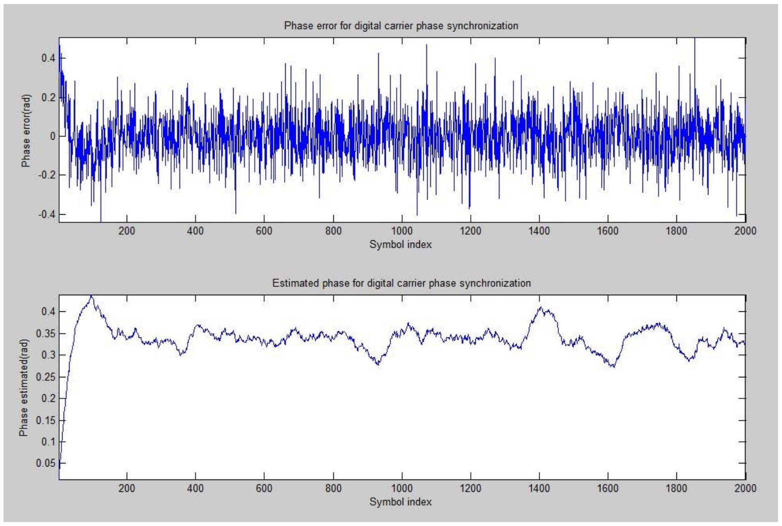
SNR=1dB

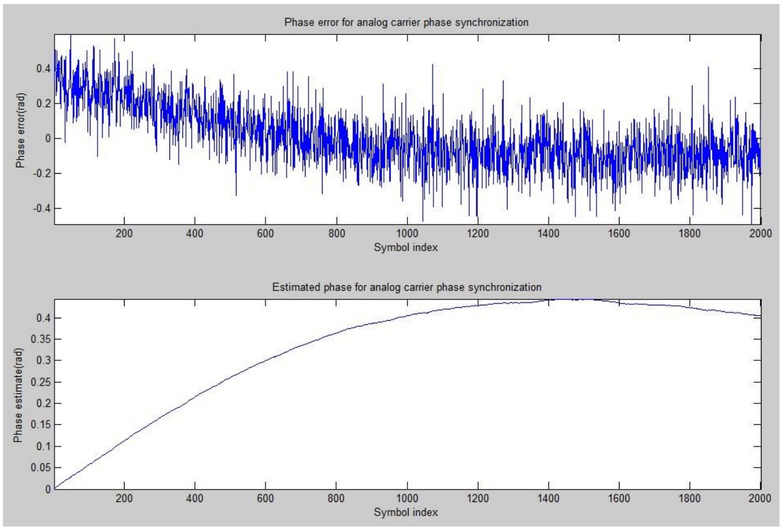

Phase error for digital carrier phase synchronization
Phase error(rad)
Symbol index
Estimated phase for digital carrier phase synchronization
Phase estimated(rad)
Symbol index


Phase error for analog carrier phase synchronization
Phase error(rad)
Symbol index
Estimated phase for analog carrier phase synchronization
Phase estimate(rad)
Symbol index

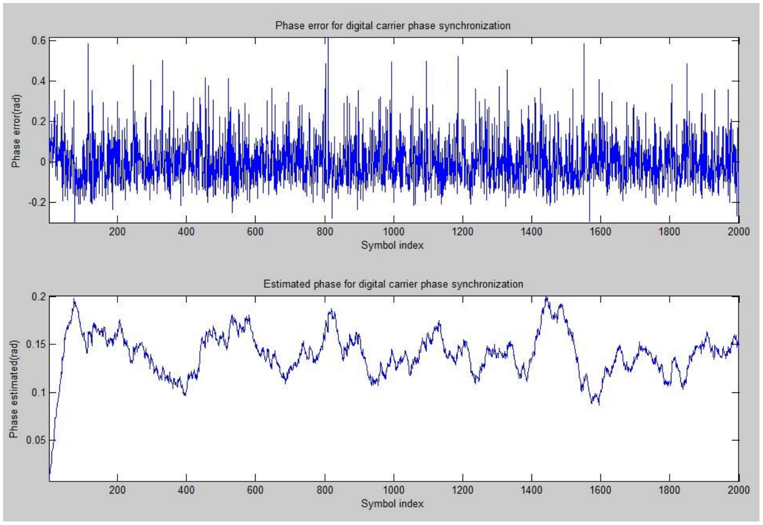
When the quantization levels increase, the phase error and estimated phase are shown below:

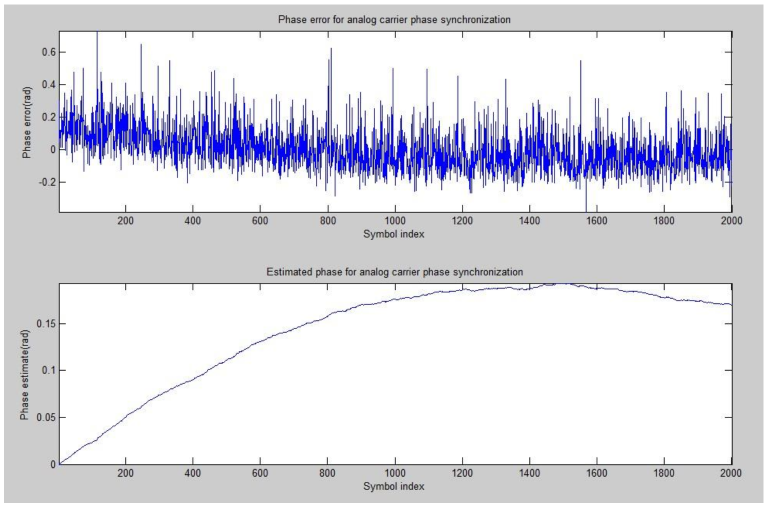
b=2





b=1





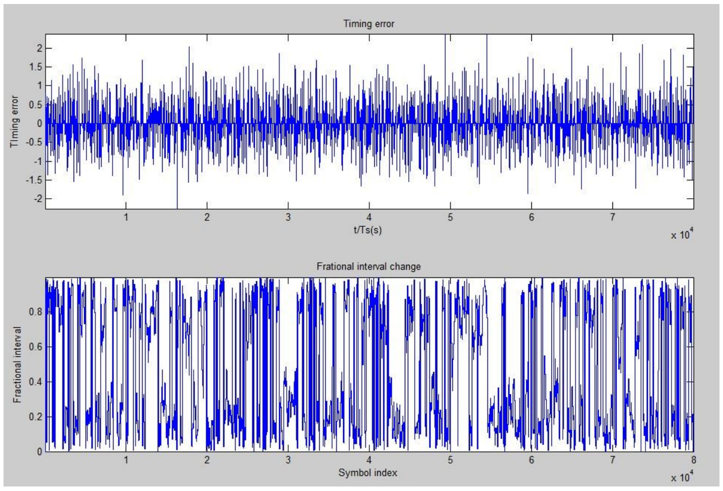
So, according to those results shown below, when the phase offset increases, digital method is not affected much but analog method becomes more inaccurate and slower. When SNR increases or quantization levels decrease, analog method doesn't change much but digital method becomes more unstable and inaccurate.

2. Symbol timing synchronization

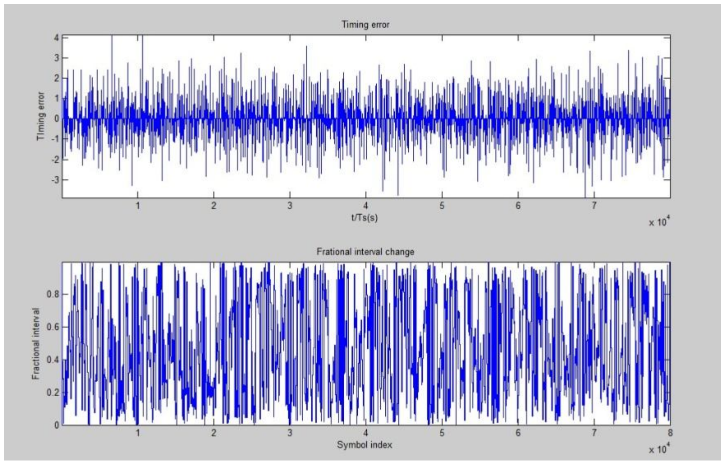
As mentioned before, an analog-like method has many advantages which lead to transport delay, hardware overhead, and higher levels of phase noise but it can produce samples that are aligned in both phase and frequency with the data clock. The digital method, which addresses these issues using asynchronous sampling rate, can reduce the complexity of hardware implementation. However, the major disadvantage of this approach is interpolation jitter when the fractional timing error accumulates and eventually become unity. Thus, when the data bits must be retransmitted over a synchronous link to some other destination, this interpolation jitter is especially problematic so that in this case, we had better use analog method instead of digital method for reliable transmission.

When the SNR decreases, the performance of digital method is shown below:

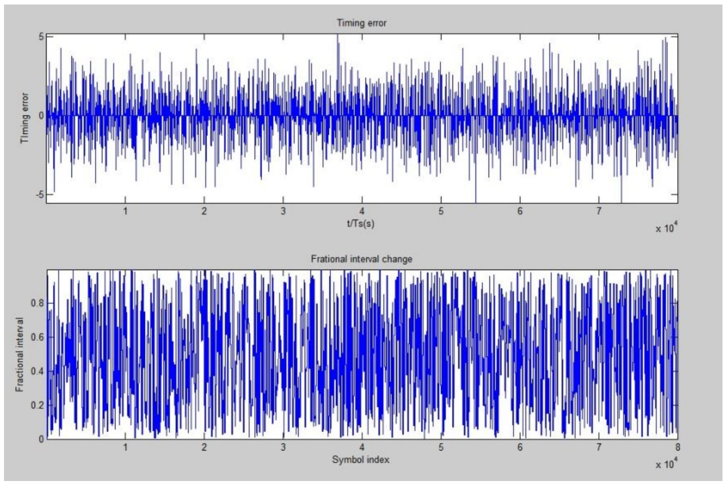
SNR=10dB



SNR=5dB

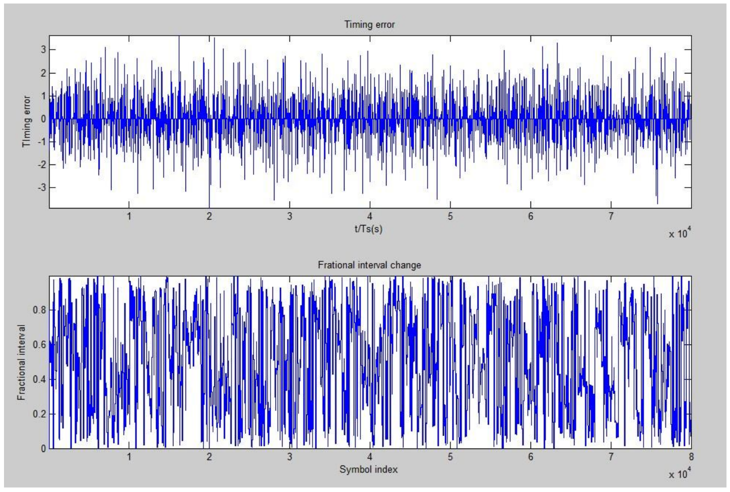


SNR=1dB

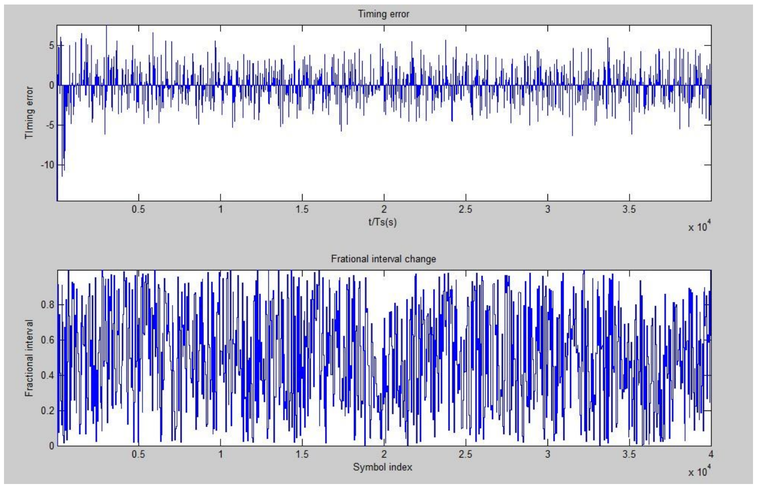


When the sampling rate increases, the performance is shown below:

sampling rate=40 samples/symbol

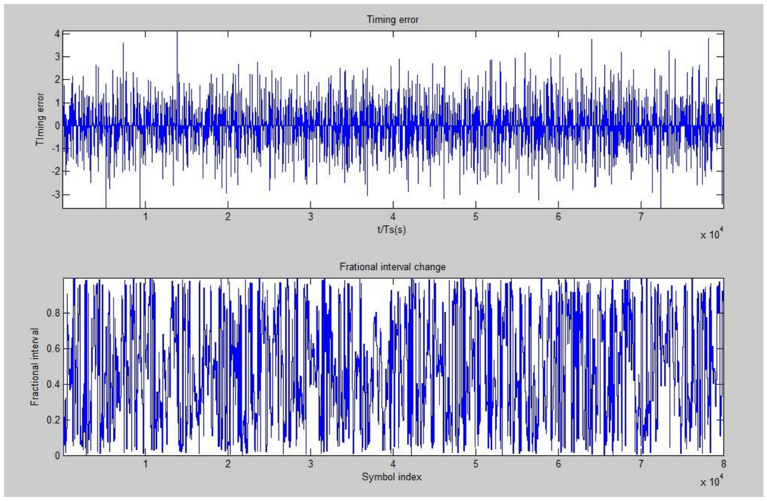


sampling rate=20 samples/symbol

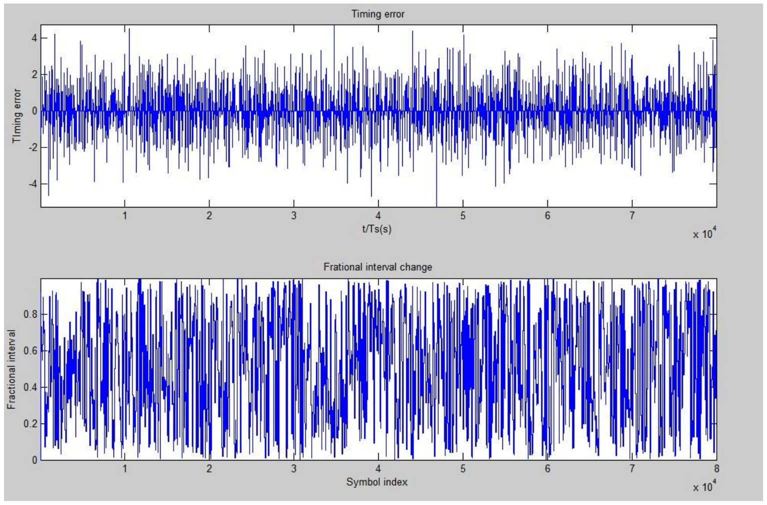


When the quantization levels decrease, the performance is shown below:

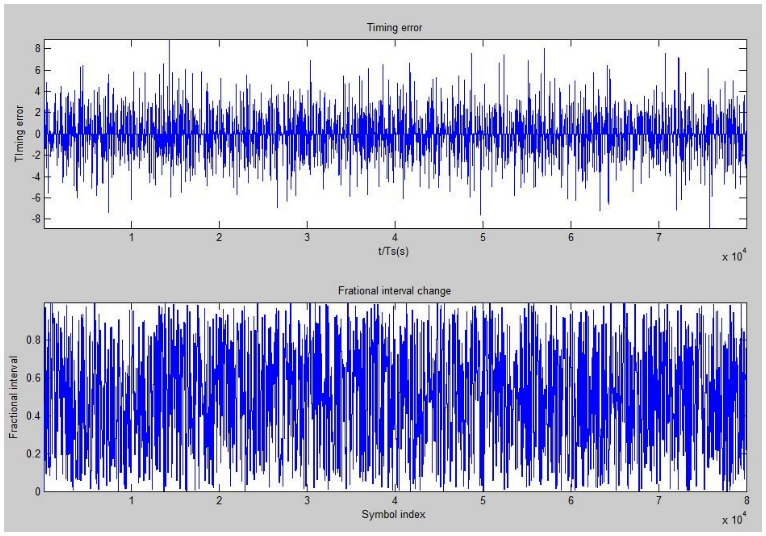
b=3



b=2



b=1

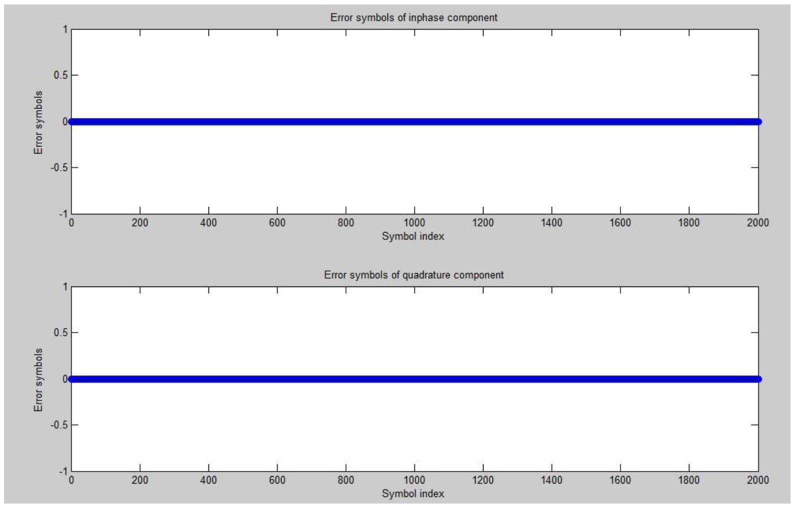


From the results displayed above, we can see that the performance of symbol timing synchronizer is affected by SNR, sampling rate and quantization levels. When SNR, sampling rate or quantization levels decrease, the timing error will become larger and the performance of the symbol timing synchronization is getting worse.

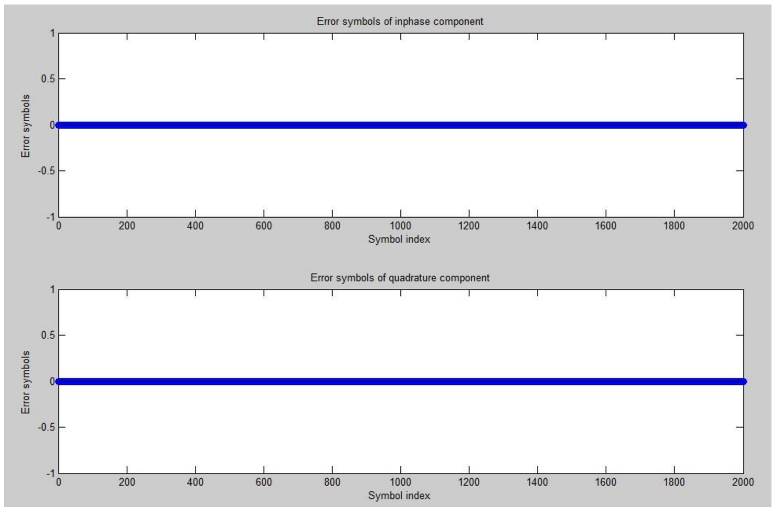
3. Probability of error:

To measure the whole digital system performance, the probability of error may be a crucial factor that should be considered. Here, I consider three factors: initial phase offset, SNR and quantization levels.

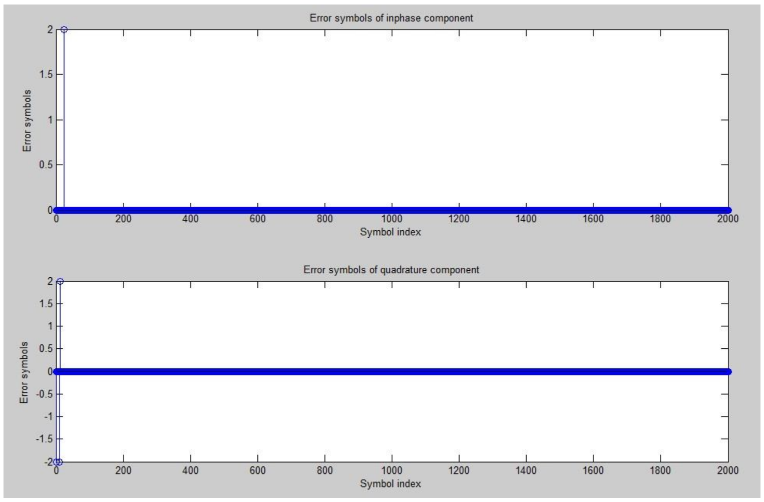
SNR=5dB, quantization levels=8, initial phase offset=22.5^o



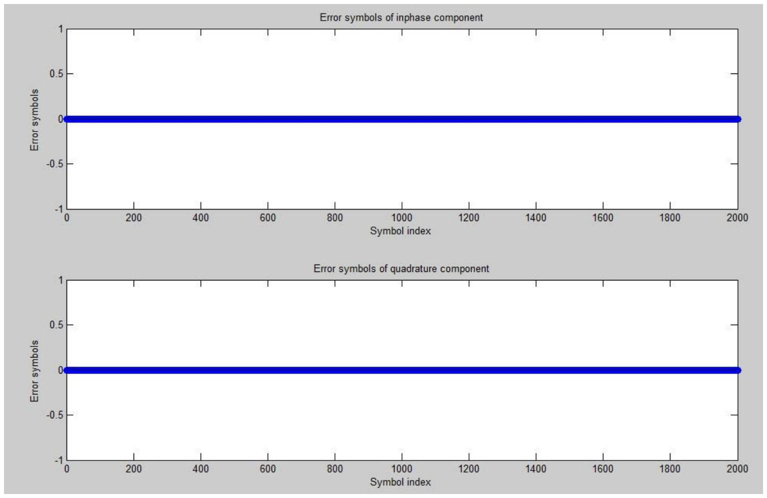
SNR=5dB, quantization levels=8, initial phase offset=30^o



SNR=5dB, quantization levels=8, initial phase offset=45^o

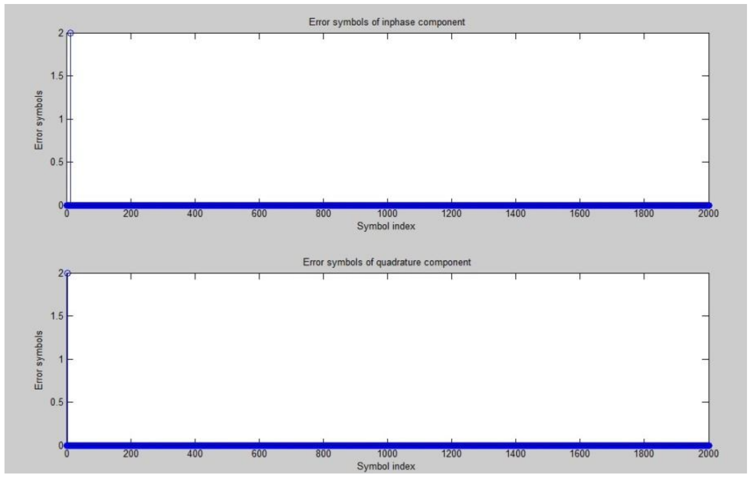


SNR=3dB, quantization levels=8, initial phase offset=30^o

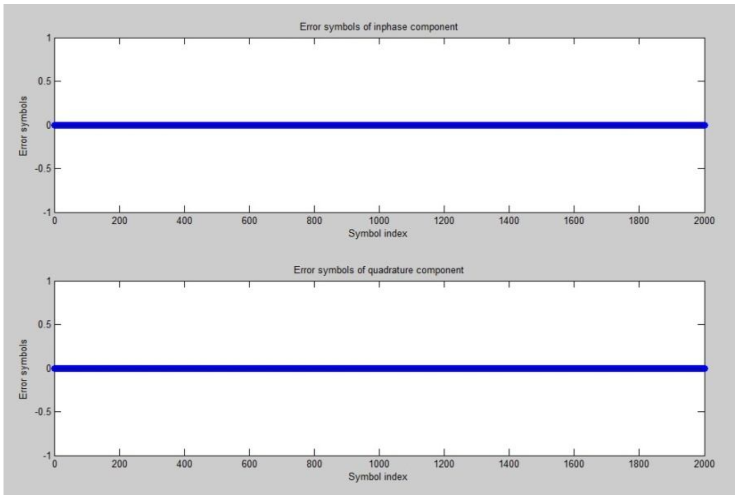


SNR=1dB, quantization levels=8, initial phase offset=30^o

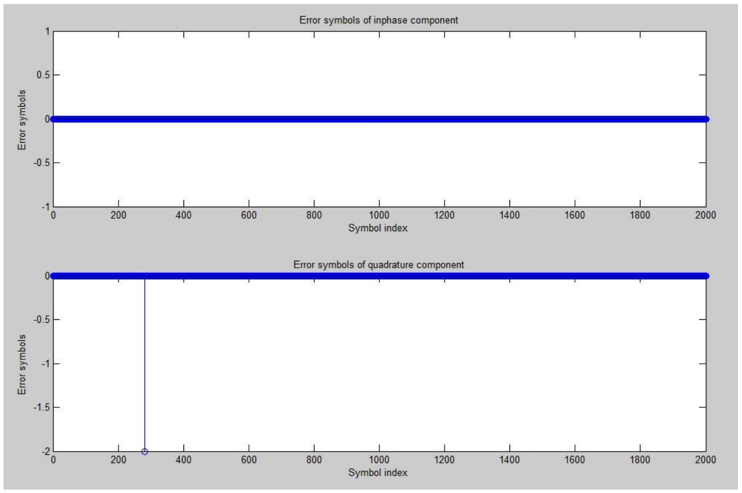
reliable.



SNR=3dB, quantization levels=4, initial phase offset=30^{o}



SNR=3dB, quantization levels=2, initial phase offset=30^{o}



From the results shown above, we can see that when the initial phase offset increases to some degree, or either of SNR and quantization levels decreases to some extent, the error symbols start to occur. But in general, the performance of the digital receiver is pretty good, stable and reliable.

[Conclusion]:

From this project, I've learned how to design a receiver and how to use discrete-time technique to improve the performance and reliability. To overcome the disadvantages of traditional receiver, some specific techniques starting from pure digital domain have to be developed. First of all, ADC can be placed to the intermediate frequency and an IF sampling operation can be applied to reduce the complexity. Then, to eliminate the upsampling operation in analog-like receiver, the carrier phase synchronization can use a fixed frequency and phase to translate the signal to baseband and compensate the carrier phase offset by rotating the downsampled matched filter outputs to remove the extra rotation. Moreover, the symbol timing synchronization can be achieved using interpolation, interpolation control and timing error detector, which can mitigate the effect caused by hybrid design and fix the IF sampling rate. To further improve the system, multi-rate processing and channelization may be applied to simplify the design and provide more convenient solution.

During the design and simulation, I found out that when one designs a system, he must know very well how every part works and how they communicate with each other. Newly added component should not affect the performance of others, otherwise I need to start over from the initial state. Communication system is complicated and has many options to choose so that decisions are very important for every step. Besides mathematical deduction and theory support, we have to carry out different kinds of experiments and simulation to explore the possible solutions or check our original design. For this time, through many times of trials, I chose the optimal set of parameters to maximize the system performance while maintaining the complexity of hardware implementation, which is crucial for both industry and academy.

Before designing the digital receiver, I reviewed the fundamental knowledge about digital signal processing and probability, which seem easy but are useful for theoretical proving and deduction. Then, I studied core parts of a communication system: modulation, demodulation, decision, signal space projection, carrier phase synchronization and symbol timing synchronization, which offer me sufficient information for the system design. At last, some advanced techniques such as multi-rate processing are explored and I am so excited about those creative ideas.

Finally, simulation is always the best way to check out the performance and find out the problems. Matlab provides plenty of built-in functions about signal processing and communication so that we can easily and quickly use those tools to simulate a near real communication environment. The process of simulating, finding problems, fixing problems and simulating again is a very good approach to either do research or work on a project. In this project, this approach is very useful and gives me a lot of information to modify the system and make it better. In future work and study, I will still insist on this efficient method and benefit from it as well.