Digital Modulation and Detection for Communication Links

EE 340: Prelab Reading Material for Experiment 7

September 21, 2016

The message (or data) signals that are represented by discrete levels are called digital signals. These signals are commonly expressed using binary digits (or bits). Digital communication involves transmission and reception of digital signals, and the techniques used in the process are called digital modulation and detection techniques, respectively [1].

Why Digital Communication: In general, signals in the digital format are easier to store, transmit and process, compared to analog signals that are more susceptible to getting corrupted by noise and distortion. However, digital bit streams are not transmitted directly, particularly in case of wireless transmission. Typically they are first modulated over a carrier (and the modulated signal is actually analog in nature).

Digital Modulation Formats: As we know, the frequency components of a square wave are not band limited due to sharp transitions in it. Therefore, we cannot directly use data streams to transmit over a bandpass channel. One has to use band-limited analog representation of the bits to modulate over a carrier wave. One can have distinct amplitudes, frequencies or phases per bit or group of bits, leading to various modulation techniques as discussed below.

(a) Amplitude Shift Keying (ASK): The amplitude of the carrier wave is changed with the bit values. The simplest form of ASK is On-Off-Keying (OOK), in which the carrier wave is transmitted for bit '1' and nothing is transmitted for bit '0'.

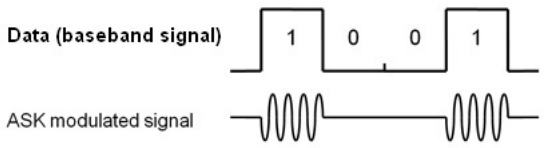


Figure 1: Amplitude Shift Keying (ASK)

(b) Frequency Shift Keying (FSK): In FSK the bit values are mapped to distinct frequencies at the transmitted (while the carrier amplitude is kept constant).

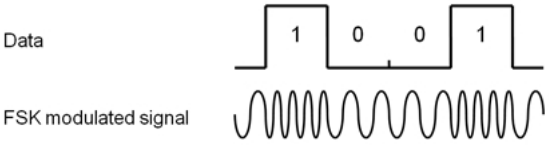
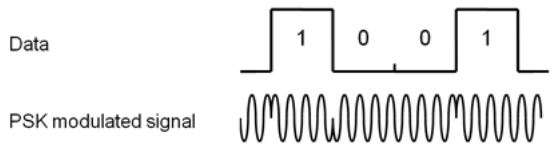


Figure 2: Frequency Shift Keying (FSK)

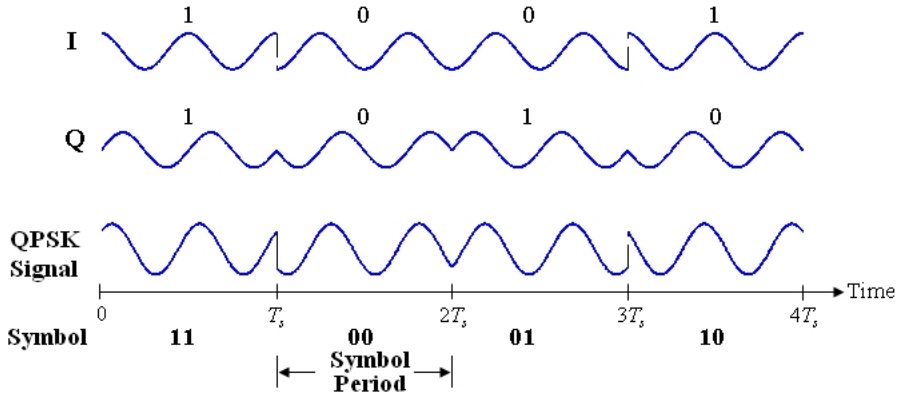
(c) Phase Shift Keying (PSK): In PSK, the bit values are mapped to distinct phases of the transmitted carrier wave (while the carrier amplitude is kept constant).

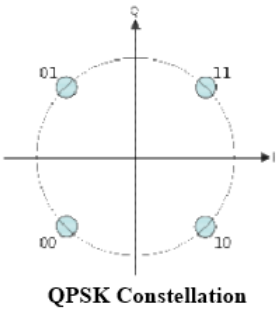


BIT	PHASE
0	0^0
1	180^0

Figure 3: Binary Phase Shift Keying (BPSK)

In QPSK (Quadrature Phase Shift Keying), we use 4 possible phases of the carrier, depending on 2 consecutive bits. As shown in Fig. 4, this can be achieved by independently multiplying each of the I and Q channel carriers, i.e. $\cos(\omega_c t)$ and $\sin(\omega_c t)$, with +1 or -1 and adding them. Show yourself that this scheme results in the carrier with possible phase values of –135°, –45°, +45° and +135°.





Symbol	I, Q Logic Levels	I, Q Amplitude Levels	Carrier Phase
00	0, 0	-1, -1	-135^0
01	0, 1	-1, 1	$+135^0$
10	1, 0	1, -1	-45^0
11	1, 1	1, 1	$+45^0$

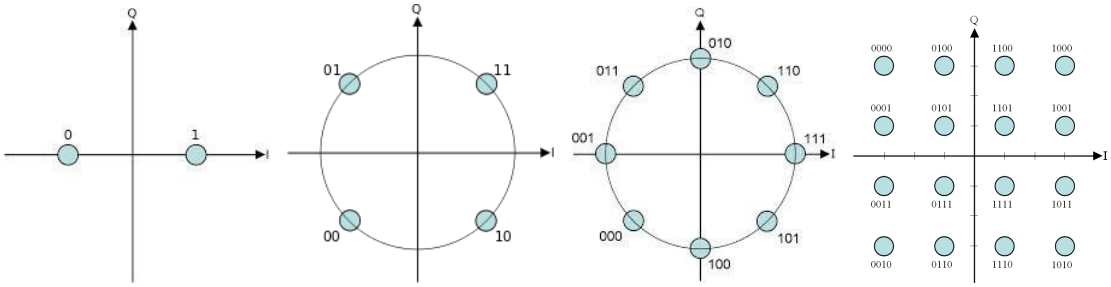
Figure 4: Quadrature Phase Shift Keying (QPSK)

Figure 4 shows the QPSK signal, its symbol table and constellation. Two message bits represent one symbol in QPSK modulation, and duration of transmission of one symbol is called symbol period (T_s). The corresponding symbol rate is $1/T_s$, and therefore the corresponding bit rate for QPSK is $2/T_s$. In general, the carrier phase can take M distinct values, for which the modulation scheme is called M-PSK. For example, in 8-PSK the carrier phase takes the values 0°, 45°, 90°, 135°, 180°, 225°, 270° and 315 ° and each symbol represents 3-bits.

(d) Quadrature Amplitude Modulation (QAM): In general, the in-phase (I) and quadrature phase (Q) components of the carrier can independently be modulated with distinct amplitude levels to obtain M-QAM (which has M possible symbol values and represent $\log_2 M$ bits per symbol). QPSK is also called 4-QAM. In 16-QAM, I and Q components of the carrier are each independently multiplied by 3, -1, +1 or +3 and then added to get the 16-QAM symbol constellation shown in Fig. 5.

The PSK and QAM signals are generally represented on the complex plane to show the constellation diagram (in-phase component amplitude is represented on the x-axis and quadrature-phase component

amplitude is represented on the y-axis). The baseband I and Q signal amplitudes are sampled (one same each for one symbol) at the centre of the symbol period and added on the x-y plane to show the signal constellation. The constellation diagrams for a few of the modulation schemes are shown below:



BPSK QPSK (or 4-QAM) 8-PSK 16-QAM

Figure 5: Constellation diagrams of some QAM and PSK modulation formats.

A Digital Communication Link: A very simplified version of a digital communication link is shown in the Fig. 6. The raw bits to be transmitted are first grouped to form symbols which are mapped to I and Q amplitudes. These 'amplitude waveforms' are like square waves, which occupy infinite bandwidth because of sharp transitions. Hence, the waveforms have to be passed through low pass filter to smoothen these transitions. However, low-pass filtering also results in inter-symbol-interference (ISI).

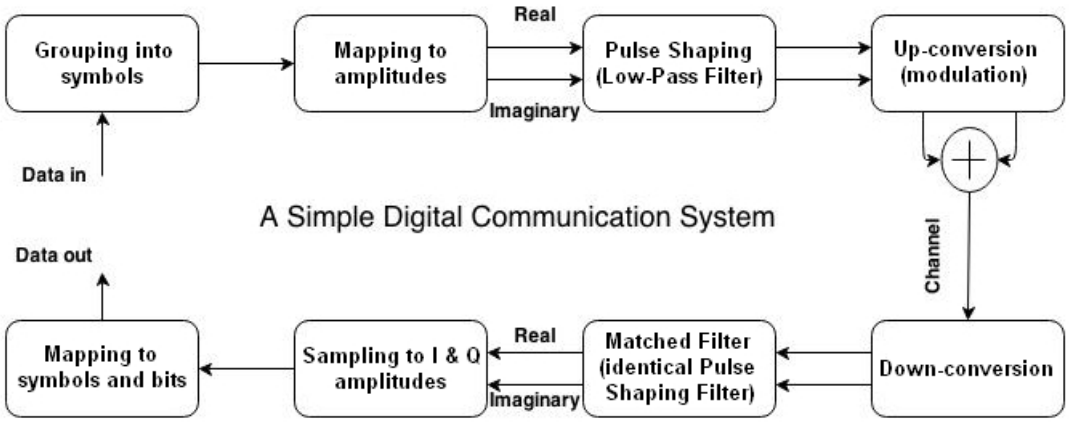


Figure 6: A simplified model of a digital communication system.

Pulse Shaping Filter: The pulse shaping filter' is designed to achieve low-pass filtering of the signal while simultaneously minimizing ISI. A Raised Cosine Filter (RCF) can achieve this (you can read more about it at https://en.wikipedia.org/wiki/Raised-cosine_filter if interested; it will be discussed later in your theory classes).

At the receiver, the sequence is reversed for demodulation of the signal, as shown again in Fig. 6. A low pass filter used at the receiver after down-conversion (recall Figure 1(b) of prelab material of Lab 4). If this filter is identical to the pulse shaping filter used at the transmitter side (i.e. if it is a 'Matched Filter'), Signal-to-Noise Ratio is maximized (you can try to prove it yourself; it will also be discussed later in your theory classes). Therefore, to achieve both matched filtering and for minimizing ISI, identical Root-Raised Cosine Filters (RRCFs) can be used at the transmitter and the receiver sides (the product or cascade of two identical RRCFs results in an RCF, which has zero ISI at ideal sampling points).

In GNU-Radio, the pulse shaping is implemented using FIR filters, tap coefficients of which are generated using the filter design tool of the software. Identical tap coefficients are used in 'Polyphase Arbitrary Resampler' block in the transmitter and 'Polyphase Clock Synchronizer' at the receiver for matching the transmitter and receiver pulse shaping filters.

Appendix: Blocks to be used in the experiment

Some of the blocks that will be used in the GNU-Radio based digital communications experiments are listed below:

Random Source
It generates random message symbols (non-negative integers) of values in a range [minimum, (maximum1)]. Repeat must be on to generate message symbols continuously.

Chunks to Symbols
It maps a stream of chunks (groups of k-bit symbols) to a stream of float or complex constellation points (or amplitudes). In the block, the symbol table can be given to directly map the symbol values to their corresponding real or complex amplitudes.

Polyphase Arbitrary Re-Sampler
To create a signal from these symbols that can be visualized as a "waveform", multiple samples are required to represent the symbol. The Polyphase Arbitrary Re-Sampler does this job. The average number of samples representing the waveform for one symbol period is called the "samples-per-symbol" or the *sps*. For example, if our incoming signal is coming at a symbol rate of 100 kSymbols/s, and we are sampling it at 1 MHz sample rate, the $sps = 10$. If the sample-rate is changed by up-sampling or downsampling the signal in GNU radio, the *sps* will change accordingly, but not the symbol rate.

The Polyphase Arbitrary Re-Sampler block implements the pulse shaping filter, which is specified using FIR taps in its properties. In this experiment, we have used Root Raised Cosine (RRC) filter defined by *tx_taps*. This block maps the complex amplitudes (assuming ONE sample represents one symbol at the input) and passes each input through the RRC filter with desired samples-per-symbol (*sps*). Typically you should have an sps of at least two for baseband signals to comfortably satisfy the Nyquist Criterion (especially if the pulse shaping filter is not a brick wall filter). Carrier modulated signals must have much larger *sps* compared to that of the baseband signal (why?).

Frequency Locked Loop (FLL)
Recall an earlier discussion (for example, in Section 1.1 of Lab 2 prelab reading material), in which it was mentioned that the transmitter and receiver frequencies may have an offset, as they are derived from different reference crystal oscillators. For example, the transmitter may be using $f_c = 399.99$ MHz, whereas the receiver may be tuned to $f_c = 400.01$ MHz (in absolute sense). Without correcting for the relative offset between the transmit and the receive frequencies, it is difficult to receive symbols correctly. The FLL block is mainly used for carrier frequency offset correction using a band edge filter at a coarse level. Finer frequency adjustment is handled by the Costas loop (or some other techniques).

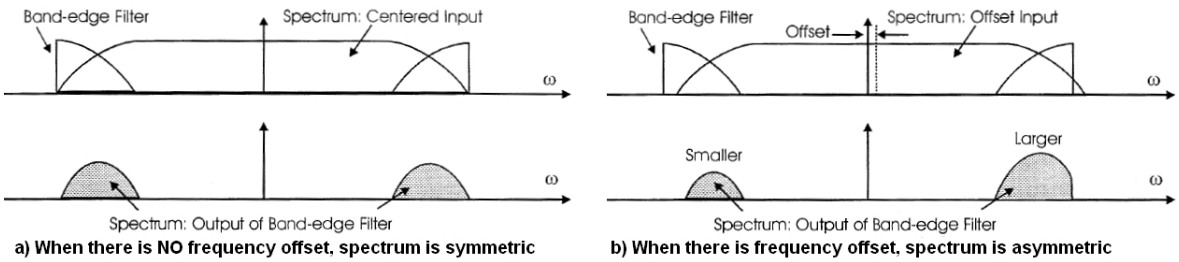


Figure 7: Working of a FLL Band Edge Filter (Courtesy: Fredric J. Harris, Multirate Signal Processing for Communication Systems)

In this block, as shown in Fig. 7, the products of the input spectrum and the spectrum of the band edge filter are monitored. If there is a frequency offset, the output of band edge filter will be asymmetric. The receiver frequency is adjusted to achieve symmetric PSD at the band-edge filter output. Some of the block parameters to be used in it include:
• *Samples per symbol:* It should be already known to you by now.
• *Filter Roll off Factor:* This parameter sets the roll off factor that is used in the pulse shaping filter

(defined in your filter design toolbox).
• *Prototype Filter Size:* This sets the number of taps in the band-edge filters. This should be about the same number of taps used in the transmitter's pulse shaping filter. A large number of taps will result in a large delay in frequency estimation and will not be able to track any fast changes. Choosing a value between 30 and 70 is usually good (you can use 55).
• *Loop Bandwidth:* Sets the loop filter bandwidth in feedback control loop for frequency adjustment. High loop bandwidth implies that the FLL can track frequency fluctuations very quickly, but can make the tracker unstable. Small loop bandwidth means that the FLL takes a longer time to adjust to the frequency error, but it is more stable. Choose its value between $\pi/100$ to $\pi/50$ (in rads/sample).

Costas Loop
This block is used for estimation of residual carrier frequency offset (that could not be removed by the FLL) and carrier phase offset, and their compensation. This block calculates the residual frequency and phase error in the signal and provides the required compensation so that the transmitted constellation is retrieved back correctly at the output. You will implement Costas Loop for frequency & phase offset estimation and correction yourself in Experiment 10.
For the Costas Loop block to work in GNU-radio, SPS of the signal at its input should be 1, i.e. the Costas Loop should be applied after the Polyphase Clock Synchronizer (having *'output sps = 1'*), which is discussed next. The order of the Costas Loop is the number of symbol levels (for example 4 for QPSK and 8 for 8-PSK).

Polyphase Clock Synchronizer for Data Timing Recovery
In addition to the offset in the carrier frequency, an offset in the data (symbol) clock rate ($1/T_s$) and phase also exists. For example, the symbol rate at the transmitter side may be 100 kSymbols/sec (and corresponding data clock be 100 kHz), whereas at the receiver, the sampling data clock may be 100.01 kHz. Also, the sampling clock edges have to be aligned to the centre of symbol periods to sample symbols with best SNR. This operation of clock synchronization and alignment is carried out by the Polyphase Clock Synchronizer block [4]. You will implement clock timing synchronization yourself in hardware in Experiment 8.

You can give the complex output samples from this block to the Scope Sink in the X-Y mode (with *output sps=1*) to observe the received symbol constellation.

CMA Equalizer
Copies of the transmitted wireless signal arrive at the receiver from multiple paths due to various reflectors, such as buildings and trees, with different delays and amplitudes, which result in ISI (intersymbol-interference) at the receiver. The equalizer block is used for removing the ISI by inverting the channel response. You will be studying more about the 'Constant Modulus Algorithm (CMA)' based equalizer in Lab 9 and implementing it yourself.

However, in this experiment, you will directly use the ready-made CMA block to compensate for multipath propagation effects, for receiving wirelessly transmitted PSK signal.
Block parameters to be used:
Num. Taps: 10; Modulus: 1; Gain: 0.0001; Samples per Symbol: *sps*

References

[1] https://en.wikipedia.org/wiki/Modulation#Digital_modulation_methods

[2] https://en.wikipedia.org/wiki/Phase-shift_keying

[3] http://www.trondeau.com/examples/2014/1/23/pfb-channelizers-and-synthesizers.html

[4] http://gnuradio.org/redmine/projects/gnuradio/wiki/Guided_Tutorial_PSK_Demodulation