Task 5: Offset Quadrature Phase Shift Keying (OQPSK)

University of Windsor
Department of Electrical and Computer Engineering
ELEC 4190 – Digital Communications

Note: Answer all questions in this document. Support all answers by attaching MATLAB and Simulink files. This lab is mostly adapted from materials provided by the authors of *Digital Communications: A Discrete-Time Approach*.

Submission: Submissions should be through Brightspace. There is a 24-hour grace period after the due date without a penalty. Late submissions and email submissions will not be accepted.

1. Introduction:

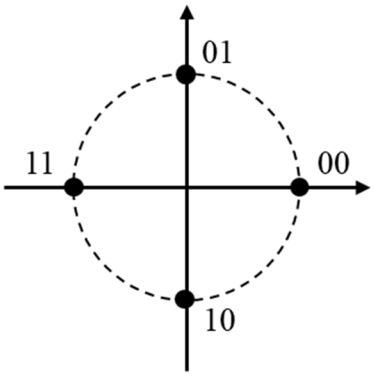
In this exercise, you will design an OQPSK system. OQPSK is a variant of QPSK modulation using four different phases to transmit. It is sometimes called staggered QPSK (SQPSK). Like QPSK, the OQPSK constellation consists of four points equally spaced on a circle. The main difference is that the quadrature component is delay half a symbol-period relative to the in-phase component in OQPSK. By offsetting the timing of the odd and even bits by one bit-period, or half a symbol-period, the in-phase and quadrature components will never change at the same time. This limits the phase-shift to no more than 90° at a time, compared to 180° in QPSK. In other words, OQPSK avoids phase trajectories through the origin which, in turn, is a good thing when a non-linear RF power amplifier is used.

For the OQPSK system, you will design the modulator and detector using blocks from Simulink, DSP System, and Communications System Toolboxes. You will first test the detector you designed by constructing a modulator to produce a test signal. Then, you will use the detector to process some given modulated data that contain a secret message each.

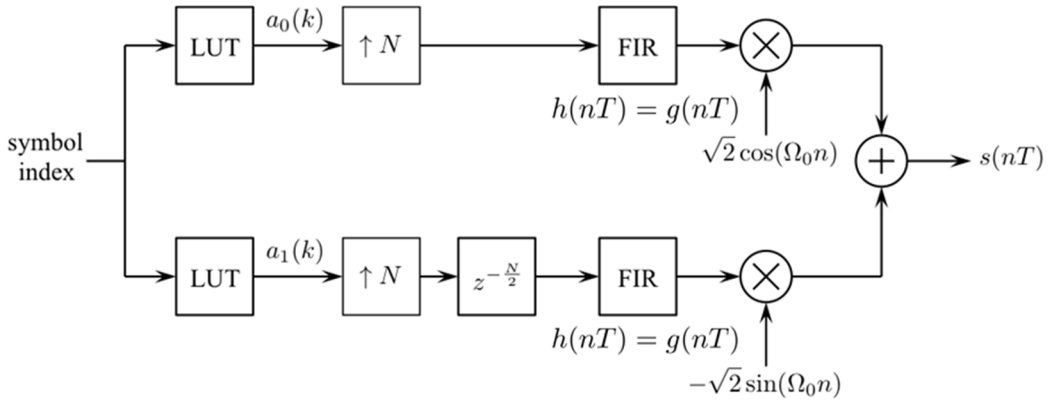
2. OQPSK System Design:

The following procedure steps you through this design process.

1. Design the modulator shown below to meet the following specifications:
   - Normalized sample rate: 24 samples/bit
   - OQPSK signal representation:

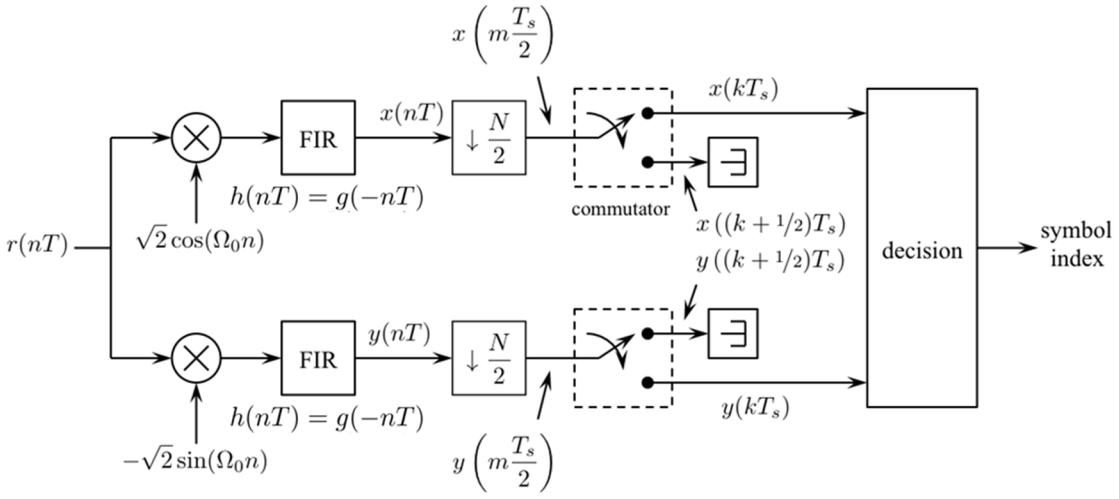


- Average energy: 9
- Normalized carrier frequency: 0.25 cycles/sample
- Carrier phase: 0 degrees
- Pulse shape: SRRC (30% excess bandwidth, span = 16 symbols)
- Symbol clock offset: 0
- Input message: the four-symbol sequence 1 2 0 3



2. Next, add an AWGN Channel block to simulate the channel noise. Set the SNR per symbol to 40 dB and the initial seed to your student number.

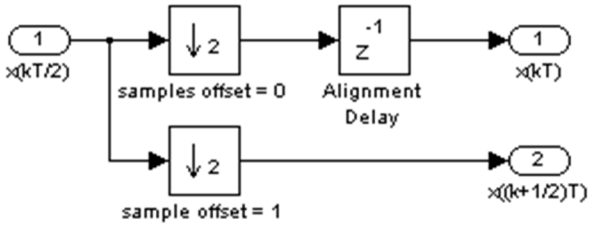
3. Now, design the detector shown below.



Note: The matched filter outputs are downsampled to two samples/symbol (or one sample/bit) in synchronism with the symbols. The samples are commutated as shown. After commutation, the clock rate is one sample/symbol, the even- and odd-indexed samples appearing in parallel. Detection is based on the even-indexed samples from the I-channel and the odd-indexed samples from the Q-channel. The unused samples are used for carrier phase.

The data samples arrive at the input of the commutator at a rate equivalent to 2 samples/symbol and are output in two parallel streams at a symbol rate equivalent to 1 sample/symbol. The commutator may be constructed in Simulink using a pair of downsample blocks with different

sample offsets and a delay as shown below. This delay is required to properly align the even- and odd-indexed samples.



4. Connect the output of your modulator to the input of the channel.

5. Connect the output of the channel to your detector.

6. Connect the output of your detector to a To Workspace block (be sure to open the Properties Dialog Window and set the Save format to matrix) and a Scope block.

7. Set the simulation parameters as follows:
   - Simulation Time:
     - Start Time: 0.0
     - Stop Time: **(16+4+1)×24-1**
   - Solver Options:
     - Type: Fixed-step
     - Solver: discrete (no continuous states)
     - Fixed step size: auto
   - Tasking and sample time options:
     - Periodic sample time constraint: Unconstrained
     - Tasking mode for periodic sample times: SingleTasking

   Note: the stop time is computed as follows: 16 = 2×(span/2) which is the delay of the pulse shaping filter plus the delay of the matched filter; 4 = the number of data symbols; 1 = the delay introduced by the modulator and commutator; 24 = the number of samples/symbol. We subtract 1 because the start time is t = 0.0.

8. Run the simulation and plot the demodulator input and the matched filter output on the same set of axes using a Scope block. The simulation produces 16 downsampled matched filter outputs; the last four correspond to the sequence 1 2 0 3. Check the values in the workspace to see if they agree with input sequence 1 2 0 3.

9. Adjust the offset of the Downsample block and Stop Time to obtain the proper values.

3. Subtask 1:

1. Run the oqpskdata.p script in MATLAB Command Window (i.e., `>> oqpskdata(uid)`) where `uid` is your student number. This step should create a file named oqpsksignal.mat in the same directory. The total number of symbols in the input message K will be printed (i.e., the number of ASCII characters is 2K/7).

2. Replace the modulator and channel blocks with a From File block and set the Filename to oqpskdata.mat and the sample time to 1.

3. Set the simulation parameters as follows:
   - Simulation Time:
     - Start Time: 0.0
     - Stop Time: **(16+K+1)×24-1**
   - Solver Options:
     - Type: Fixed-step
     - Solver: discrete (no continuous states)
     - Fixed step size: auto
   - Tasking and sample time options:
     - Periodic sample time constraint: Unconstrained
     - Tasking mode for periodic sample times: SingleTasking

4. Run the simulation.

5. The detector produces approximately **17+K** symbol estimates. The last K of these correspond to the ASCII characters of the message. Determine the message using either a MATLAB script or an ASCII Table.

6. Submit the following:
   - Two Simulink files: one file for the system design and one file for the subtask.
   - A summary report with screenshots of the output of each block in both files and the secret message along with any codes used.