**U19EC046 | DCOM | LAB 10**

**Date: 08-11-2021**

**AIM**

To study and simulate Quadrature Phase Shift Keying (QPSK).

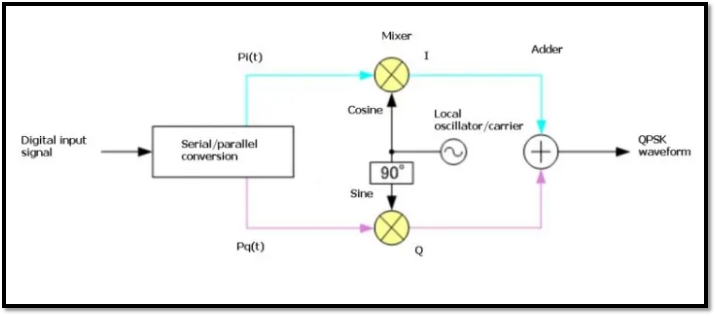
**THEORY**

ASK, FSK and BPSK transmit one bit per symbol and hence carrier is assumed to have one of the two possible states to transmit 1 0r 0.

Quadrature Phase Shift Keying (QPSK) is a form of Phase Shift Keying which transmits two bits per symbol.

Since it transmits two bits per symbol there are four possible combinations and thus there is four different phases.

* For 𝝿/4 QPSK, the four different phases are 45, 135, 225, 315.
* QPSK symbols are not represented by 0 or 1 but it is represented as 00, 01, 10 and 11.
* QPSK carry twice as much information as ordinary PSK using the same bandwidth.
* QPSK is used for satellite transmission of MPEG2 video, cable modems, videoconferencing, cellular phone systems, and other forms of digital communication over an RF carrier.

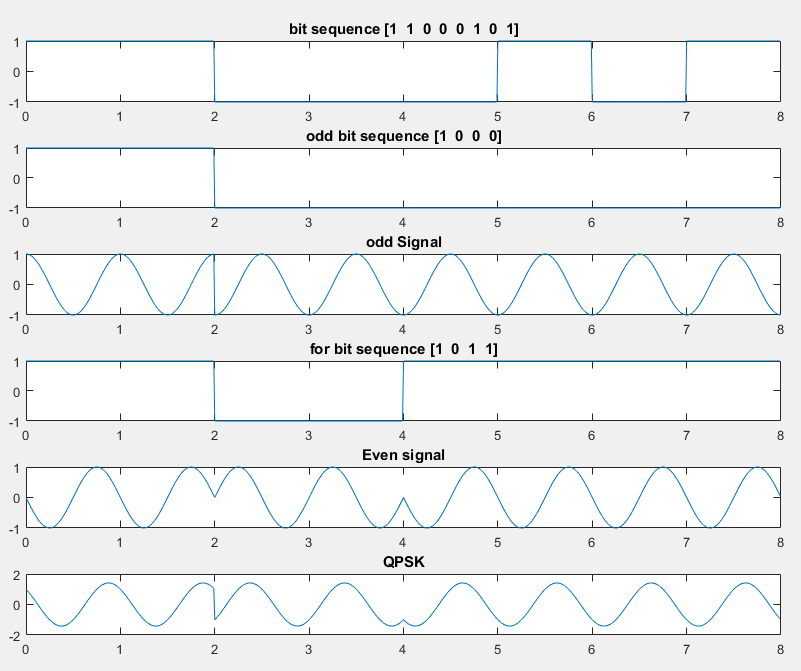


QPSK modulation circuit consists of a bit-splitter, 2-bit serial to parallel converter, two multipliers, a local oscillator, and a summer. At the transmitter input, the message signal bits are separated as even bits and odd bits using a bit splitter. The even and odd components of bit sequence are known as Quadrature and In phase signals respectively. Then, Quadrature and In phase signals are modulated by its corresponding carrier. Here, the Local Oscillator is used for generating the carrier waveform. After modulation, modulated Quadrature and In-phase signals are given to the summer to generate QPSK signal.

**MATLAB CODE**

|  |
| --- |
| ***//waveGen.m***  **function op = waveGen( bits, strechCoefficient )**  **op = cell(1, 3);**  **dt = 0.01;**  **t = 0 : dt : length(bits)\*strechCoefficient - dt;**  **op{1} = polarNRZ(bits, strechCoefficient);**  **op{2} = t;**  **end**  **function op = polarNRZ( bits, strechCoefficient )**  **one = ones(1, 100\*strechCoefficient);**  **zero = zeros(1, 100\*strechCoefficient) - 1;**  **graph = [];**  **for i = 1:length(bits)**  **if(bits(i)==1)**  **graph = [graph one];**  **else**  **graph = [graph zero];**  **end**  **end**  **op = graph;**  **end**  ***//main.m***  **clc; clear all;**  **bits = input('Enter bit sequence :');**  **if isempty(bits)**  **bits = randi([0 1], 1, 8);**  **end**  **bitOdd = bits(1: 2: end);**  **bitEven = bits(2: 2: end);**  **op = waveGen(bits, 1);**  **polarNrz = op{1};**  **t = op{2};**  **opOdd = waveGen(bitOdd, 2);**  **polarNrzOdd = opOdd{1};**  **opEven = waveGen(bitEven, 2);**  **polarNrzEven = opEven{1};**  **sin\_ = -sin(2\*pi\*t);**  **cos\_ = cos(2\*pi\*t);**  **oddSignal = polarNrzOdd.\*cos\_;**  **evenSignal = polarNrzEven.\*sin\_;**  **QPSK = oddSignal + evenSignal;**  **subplot(6, 1, 1)**  **plot(t, polarNrz)**  **title(['bit sequence [' num2str(bits) ']'])**  **subplot(6, 1, 2)**  **plot(t, polarNrzOdd)**  **title(['odd bit sequence [' num2str(bitOdd) ']'])**  **subplot(6, 1, 3)**  **plot(t, oddSignal)**  **title(['odd Signal'])**  **subplot(6, 1, 4)**  **plot(t, polarNrzEven)**  **title(['for bit sequence [' num2str(bitEven) ']'])**  **subplot(6, 1, 5)**  **plot(t, evenSignal)**  **title(['Even signal'])**  **subplot(6, 1, 6)**  **plot(t, QPSK)**  **title(['QPSK'])** |

**OUTPUT**



**CONCLUSION**

In this experiment, we have generated the Quadrature Phase Shift Keying (QPSK) signal form message bits.