Special Topic - Mini project

MSK Modulator and Demodulator

Implementation in MATLAB/Simulink

Aravind S
PES1201801734
ECE - 4D

Ashvin B
PES1201801245
ECE - 4C

Pranay Mundra
PES1201801166
ECE - 4A

Individual Contributions

- **Aravind S(PES1201801734)**
  - Modulator in MATLAB
  - Filter design for the demodulator in Simulink
  - Implementation of the FFT method in MATLAB and Simulink

- **Pranay M(PES1201801166)**
  - The modulator in Simulink and C
  - Demodulator algorithm design and implementation in Simulink
  - Implementation of Derivative method in Simulink

- **Ashvin B(PES1201801245)**
  - Literature survey
  - Encountered problems in the implementation of Costas loop in Simulink
  - Implementation of Peaks and Valleys method in MATLAB

Problem Statement

To build a Modulator and Demodulator for MSK waves for the following specifications -

- Carrier Frequency of 1800Hz.
- Two tones of 1200Hz(for 0) and 2400Hz(for 1).
- A data rate of 2400 bits per second

Introduction

Minimum Shift Key

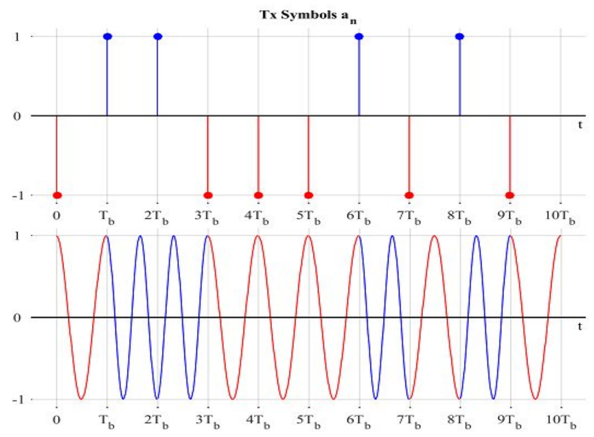
Minimum shift keying, MSK, is a form frequency modulation based on a system called continuous-phase frequency-shift keying.

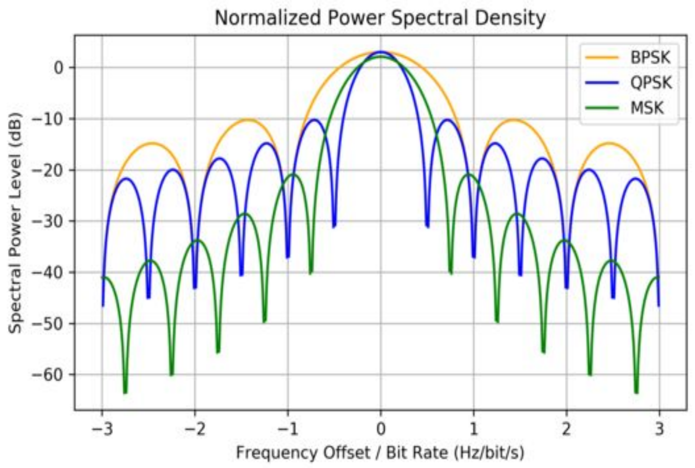
Minimum Shift Keying (MSK) is one of the most spectrally efficient modulation schemes available. Due to its constant envelope, it is resilient to non-linear distortion and was therefore chosen as the modulation technique for the GSM cell phone standard.

Advantages of using MSK

MSK, minimum shift keying has the feature that there are no phase discontinuities and this significantly reduces the bandwidth needed over other forms of phase and frequency-shift keying.

Minimum shift keying, MSK offers advantages in terms of spectral efficiency when compared to other similar modes, and it also enables power amplifiers to operate in saturation enabling them to provide high levels of efficiency.





Tried And Tested Approaches

1) FFT Method

Carrier-wave is broken into sub-parts and FFT is applied to each of them to obtain the frequency.

Drawback:- There was not enough data for FFT to function properly.

2) Peaks and Valleys

Local maxima and minima were found in the carrier-wave and the time difference was measured between these points. A lower difference implies a frequency of 2400Hz and a higher difference corresponds to 1200Hz.

Drawback:- We couldn’t lock the phase as it only works for a phase-locked loop.

3) Derivative method

A reference signal is generated whose derivative is continuously compared with the same of the MSK wave. A larger absolute derivative corresponds to a higher frequency and is used to classify the bit value.

Drawback:- Requires a phase-locked loop until the first transition is found.

4) Filtering method

A low-pass Bessel filter is used to segregate the two frequency components of the MSK wave and is subsequently classified into bit values depending on the where the transitions(in the form of peaks surrounding troughs) occur.

Drawback:- Minor instability around transition points.

Modulator

Modulation is the process of varying one or more properties of a periodic waveform, called the carrier signal, with a modulating signal that typically contains information to be transmitted.

MSK being a form of Frequency Shift Keying, the frequency of the carrier wave was varied according to the digital data to be transmitted. The carrier frequency would have to be increased id the input is 1 and decreased if the input is 0.

This implies that the output MSK would be of the form:

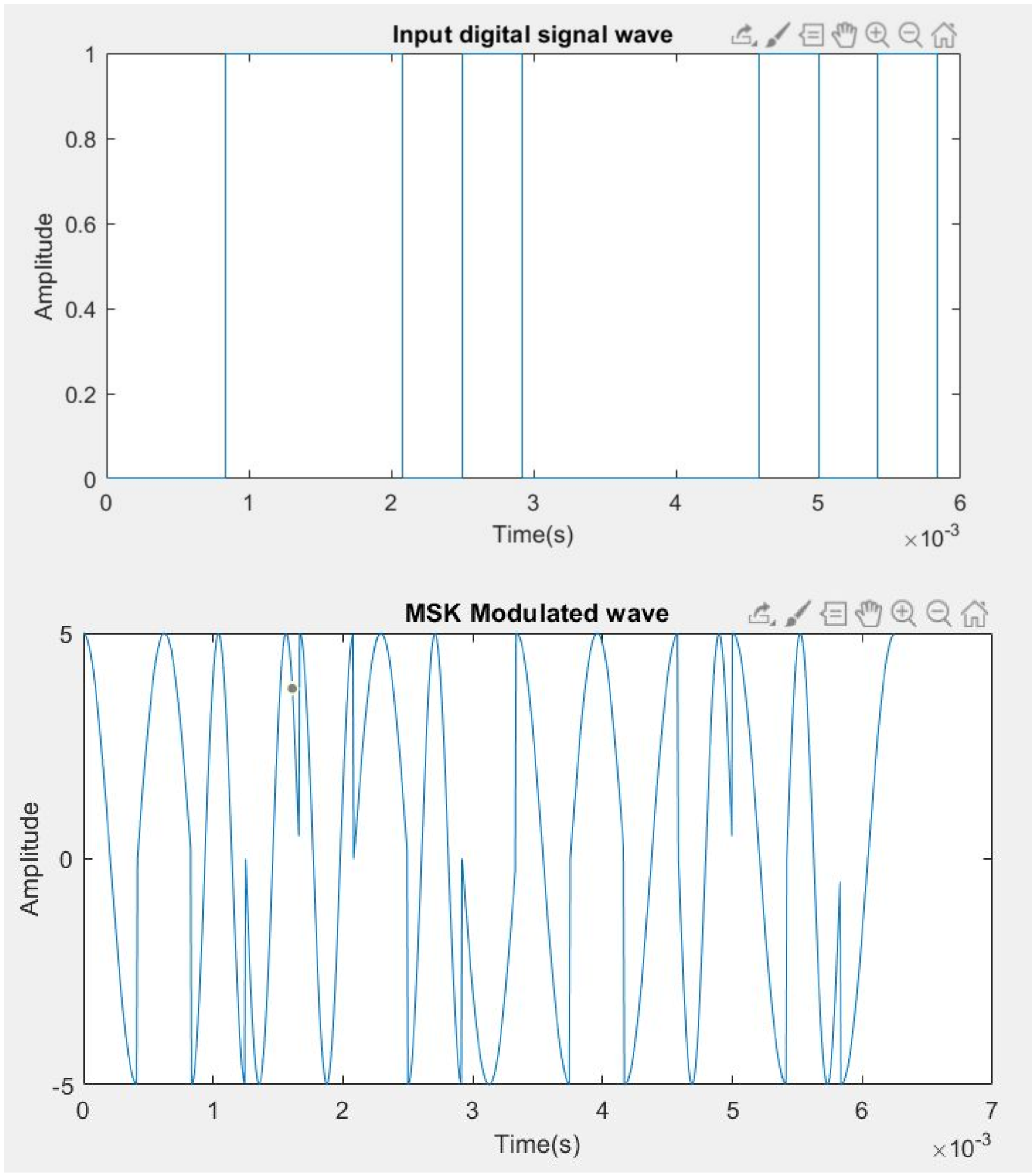
$$s(t) = A\cos 2\pi F_i t = A\cos\left[2\pi\{F_c \pm \Delta_F\}t\right]$$

Where

$$0 \leq t \leq T_b$$

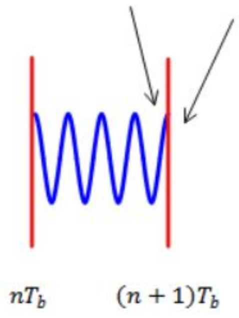
Here, △F is the frequency change. The value of △F will be 600Hz. So the carrier frequency is either increased to 2400Hz or decreased to 1200Hz which are the tones for 1 and 0 respectively.

When this was implemented, the output was as follows:



As can be seen from the above graphs, there are phase discontinuities in the output wave. Hence a phase correction has to be implemented.

This phase correction was obtained by equating the phase of the wave at both sides of $t=(n+1)T_b$ as shown in the figure below:



Which implies:

$$2\pi\frac{a_n R_b}{4}(t-nT_b)+\theta_n\Big|_{t=(n+1)T_b} = 2\pi\frac{a_{n+1}R_b}{4}(t-(n+1)T_b)+\theta_{n+1}\Big|_{t=(n+1)T_b}$$

On solving this, we get:

$$\theta_n = \theta_{n-1} + a_{n-1}\frac{\pi}{2}$$

Where a_{n-1} is the previous bit of data and θ_{n-1} is the previous phase correction.

This is the necessary phase correction.

Thus, the final MSK signal is generated as:

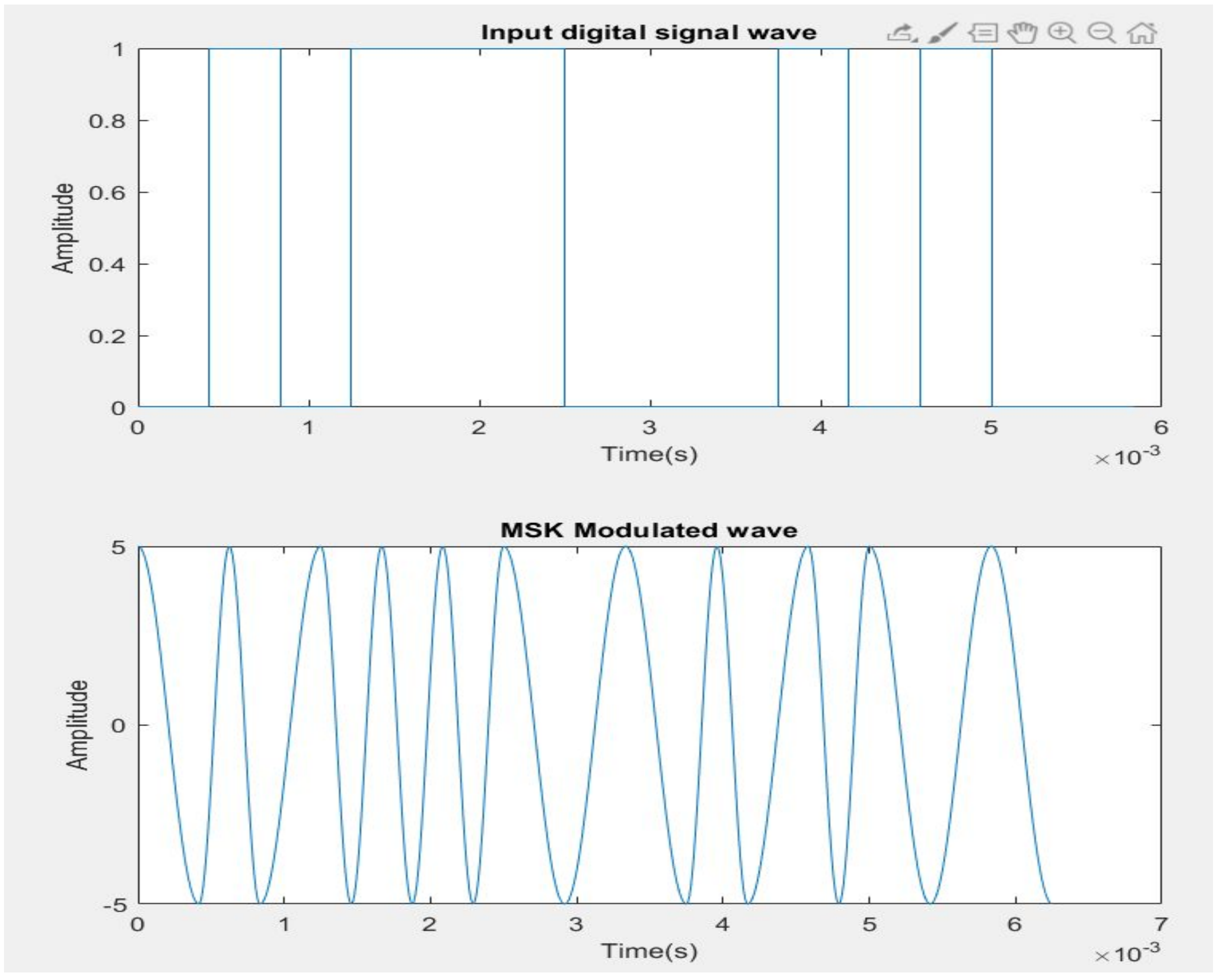
$$s(t) = \cos\left[2\pi F_c t + 2\pi \frac{a_n R_b}{4}(t - nT_b) + \theta_n\right]$$

Where,

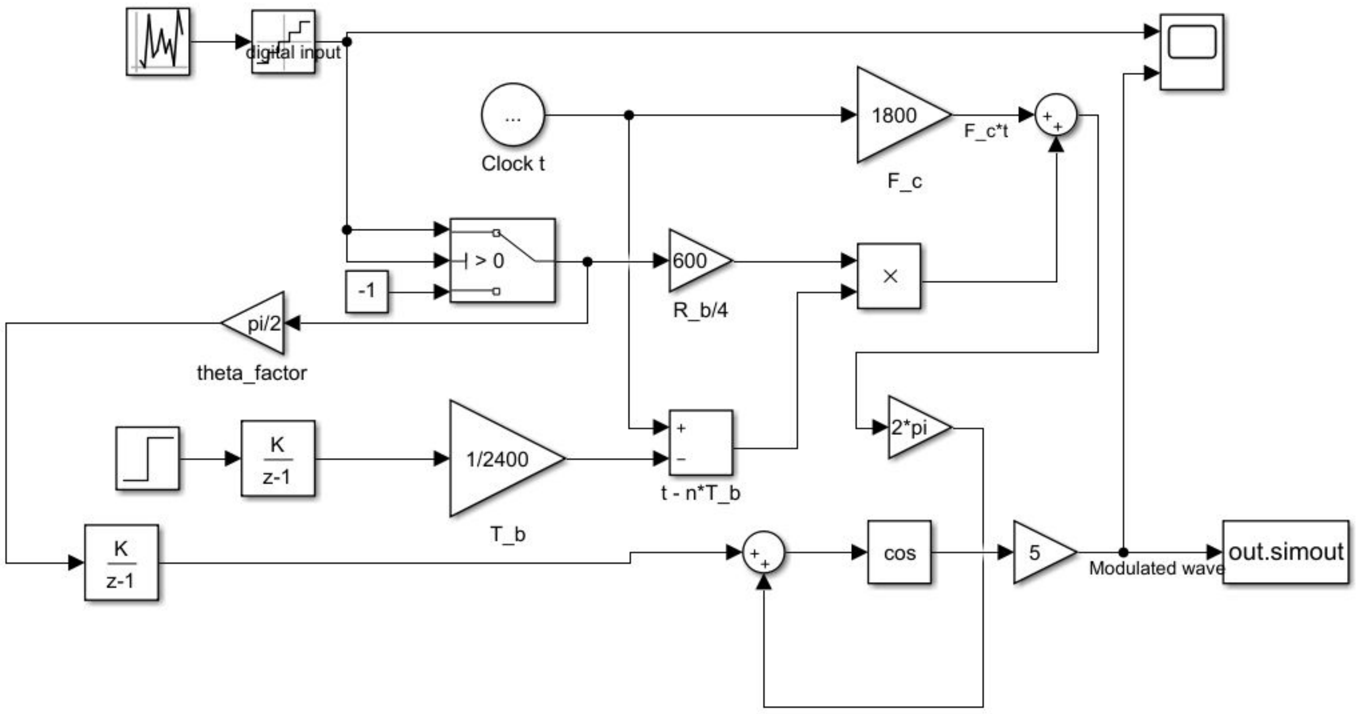
$$\theta_n = \theta_{n-1} + a_{n-1}\frac{\pi}{2}$$

The modulator was first implemented in MATLAB and then in Simulink. The MATLAB code can be found in the annexure (a).

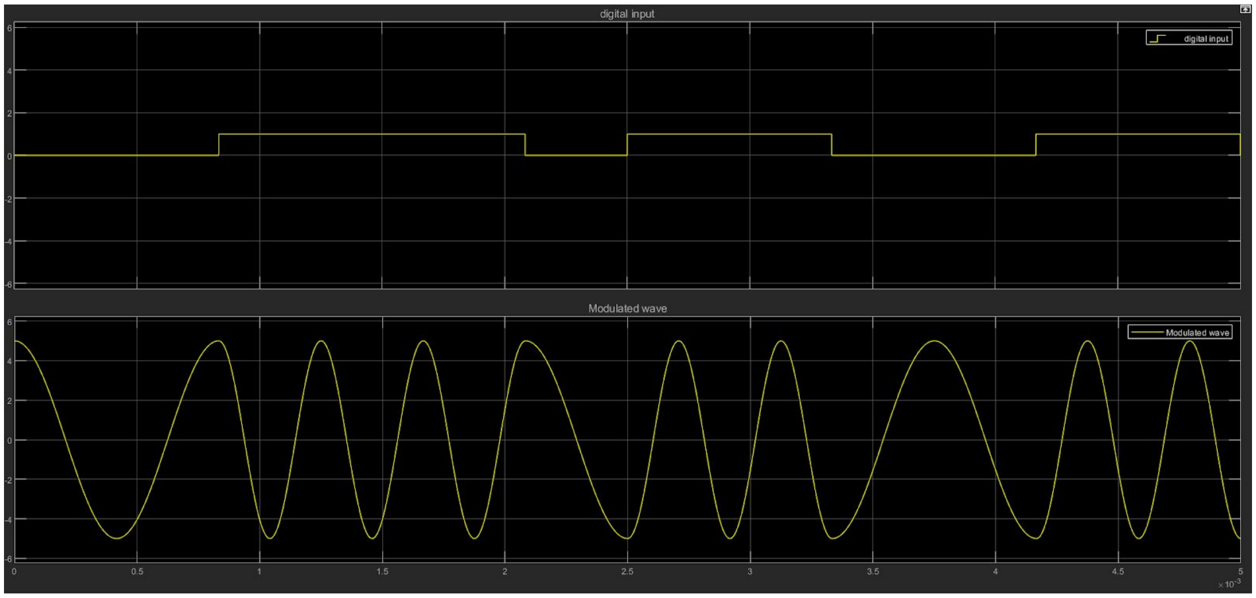
The output of the MATLAB code was as follows:



The Simulink model is as follows:



The output of the Simulink model is as follows:



Demodulator

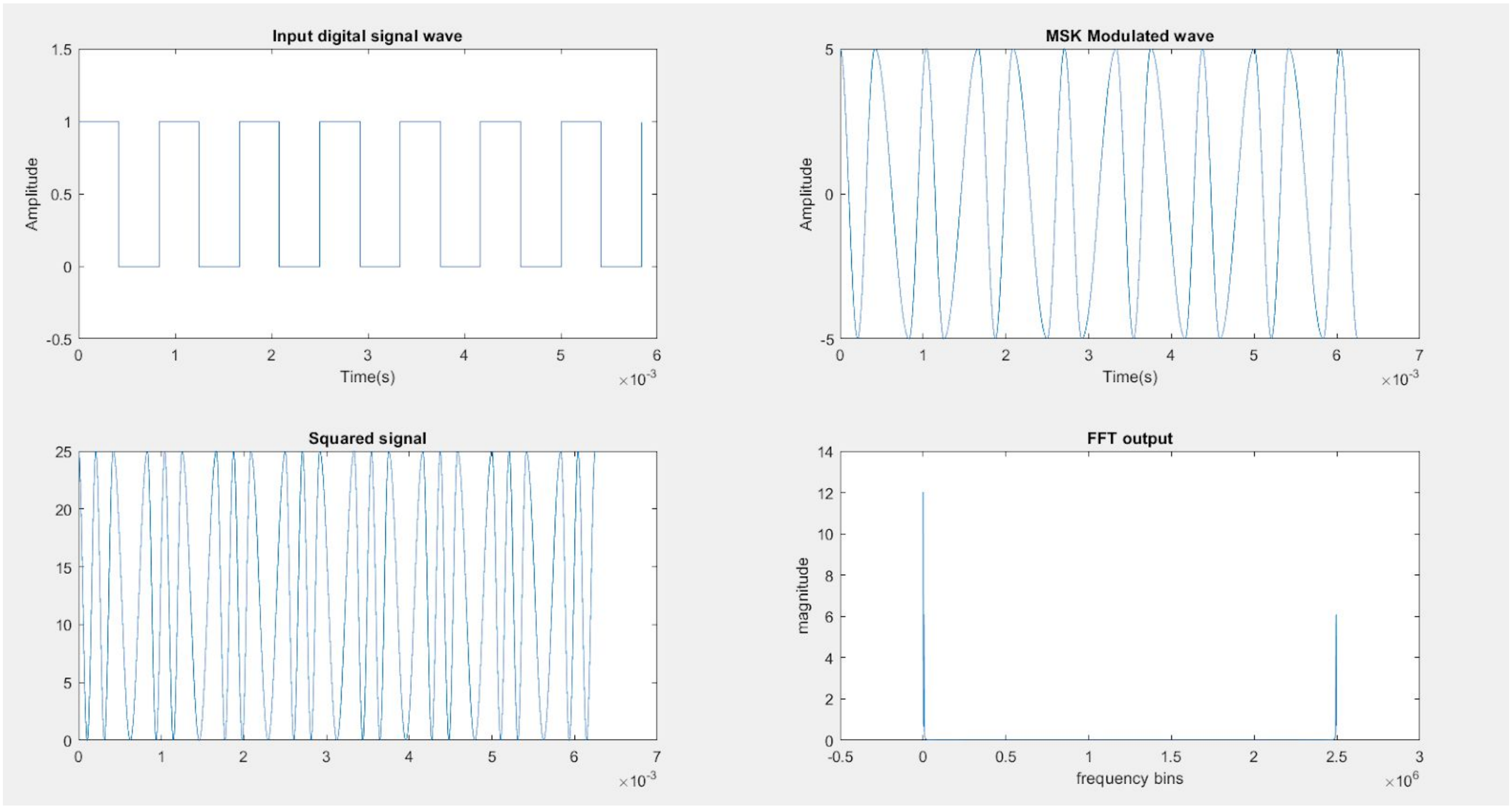
The input is the modulated wave corresponding to an alternating bitstream of size 15.

The following approaches were considered while designing the demodulator:

1. FFT on frames

The MSK modulated wave was decomposed into frames and the Fast Fourier transform was computed for each frame. Since the frequency tones were separated by 1200Hz and 2400Hz(for a squared input) it would be fairly easy to locate the spikes in the transform. There were a few issues where it was noticed that for the FFT to be accurate, a larger frame would be required which spans more than Tb of the MSK wave. Hence realizing that there were not enough data points, this approach was discarded.

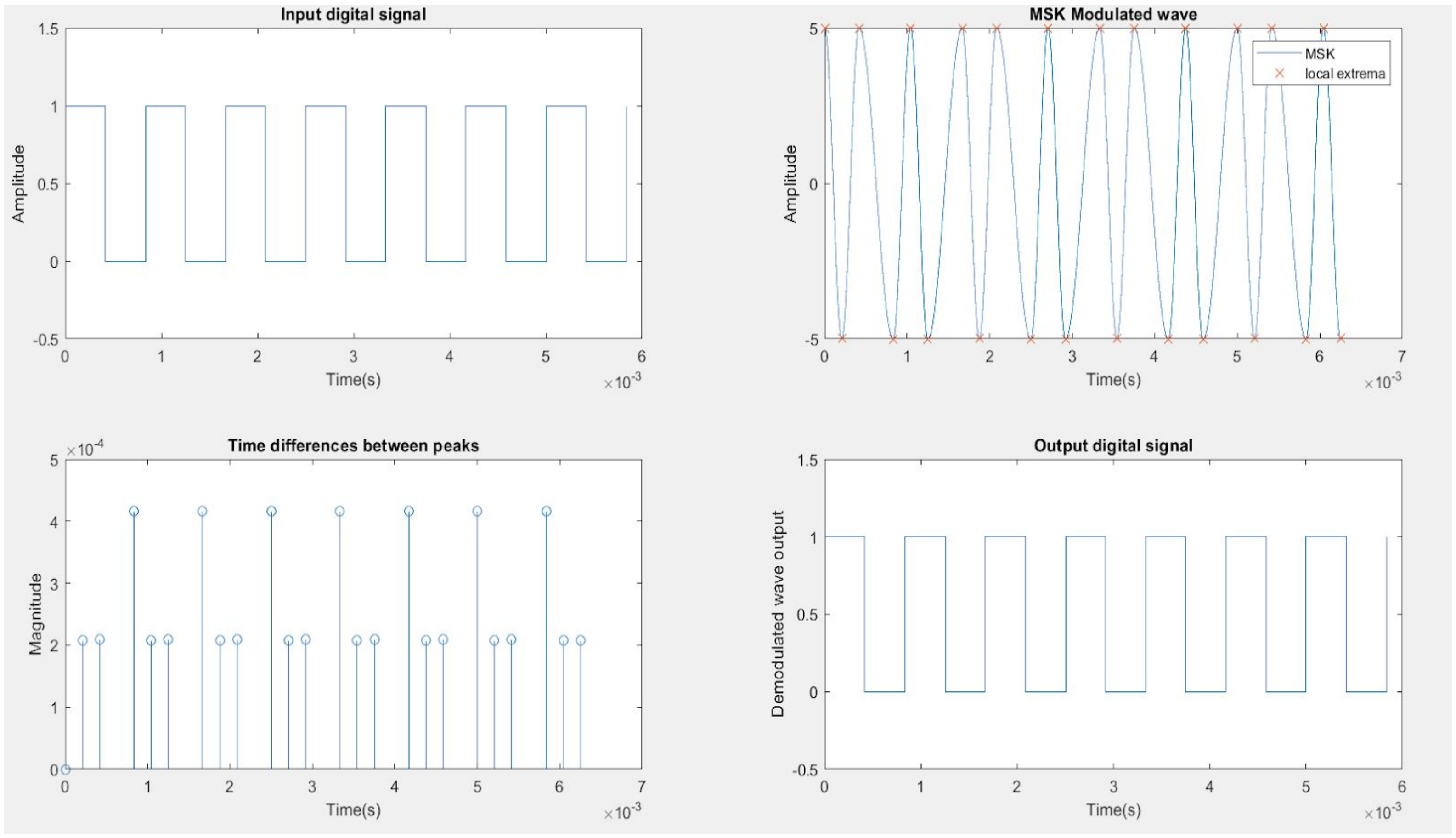
MATLAB script for FFT on a single frame can be found in annexure (b).



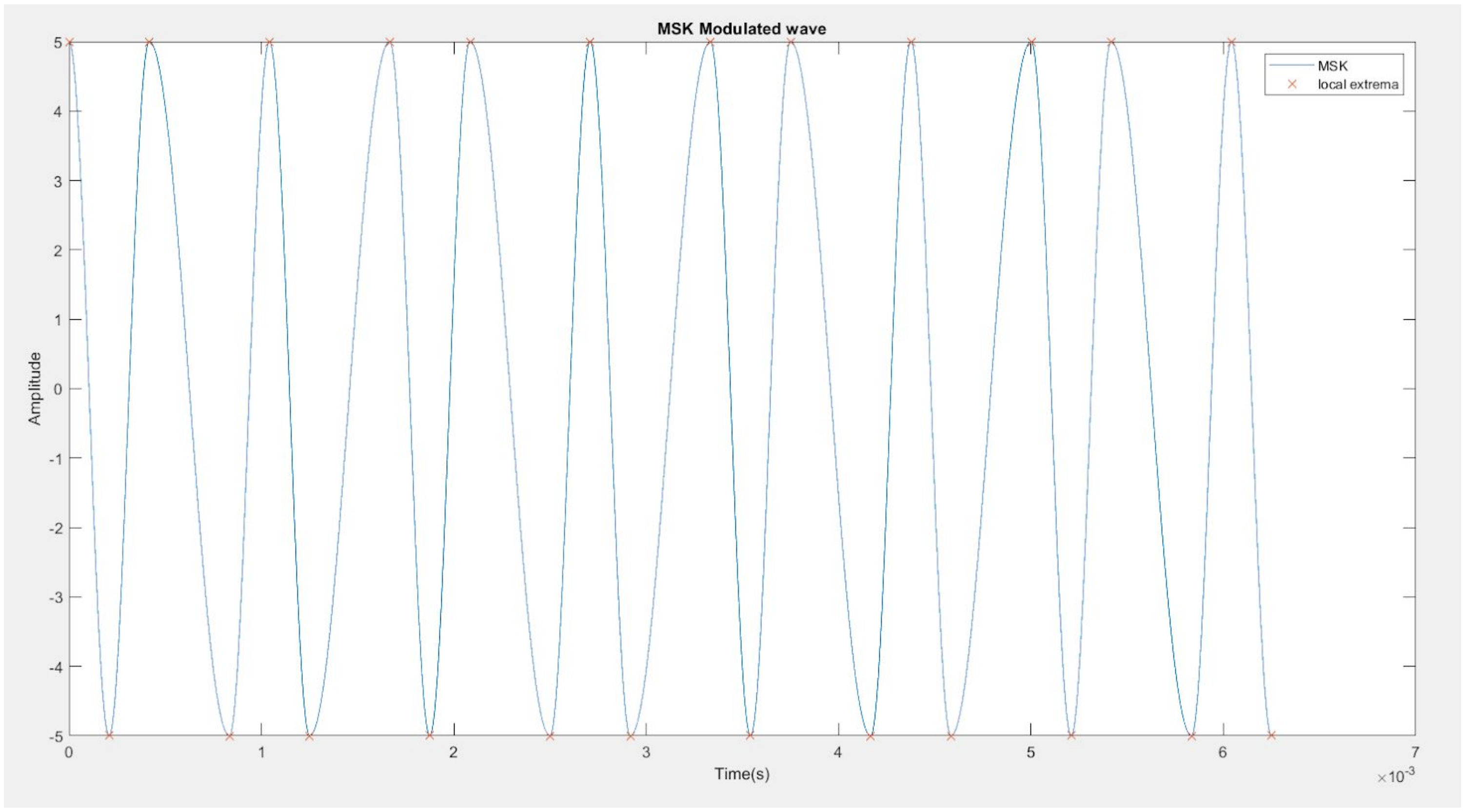
2. Time differences between successive peaks and valleys

Using a simple peak finding algorithm, the local maxima and minima were found and the time differences between those time indices were computed. Differences were found to be large wherever there was a lower frequency component while small where a higher frequency component was present. These differences were then measured and classified accordingly into bit values.

The MATLAB script can be found in annexure (c).



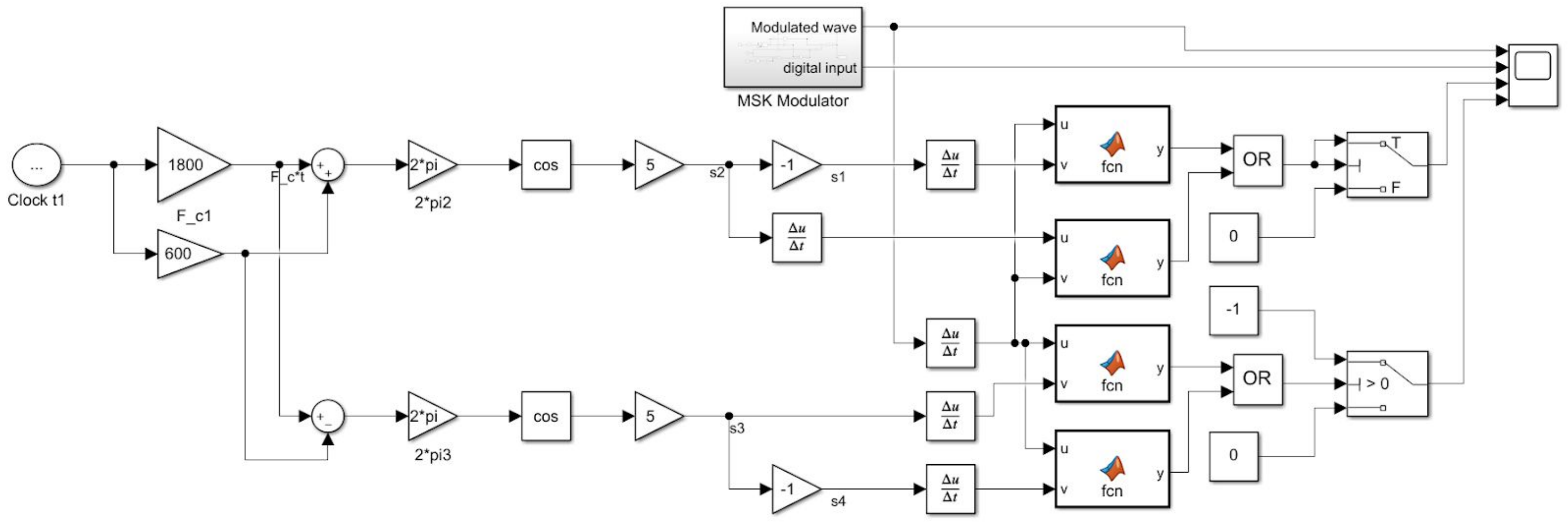
The below graph is the second subplot displaying the local extrema in greater detail (peaks and valleys). The crosses indicate these points.

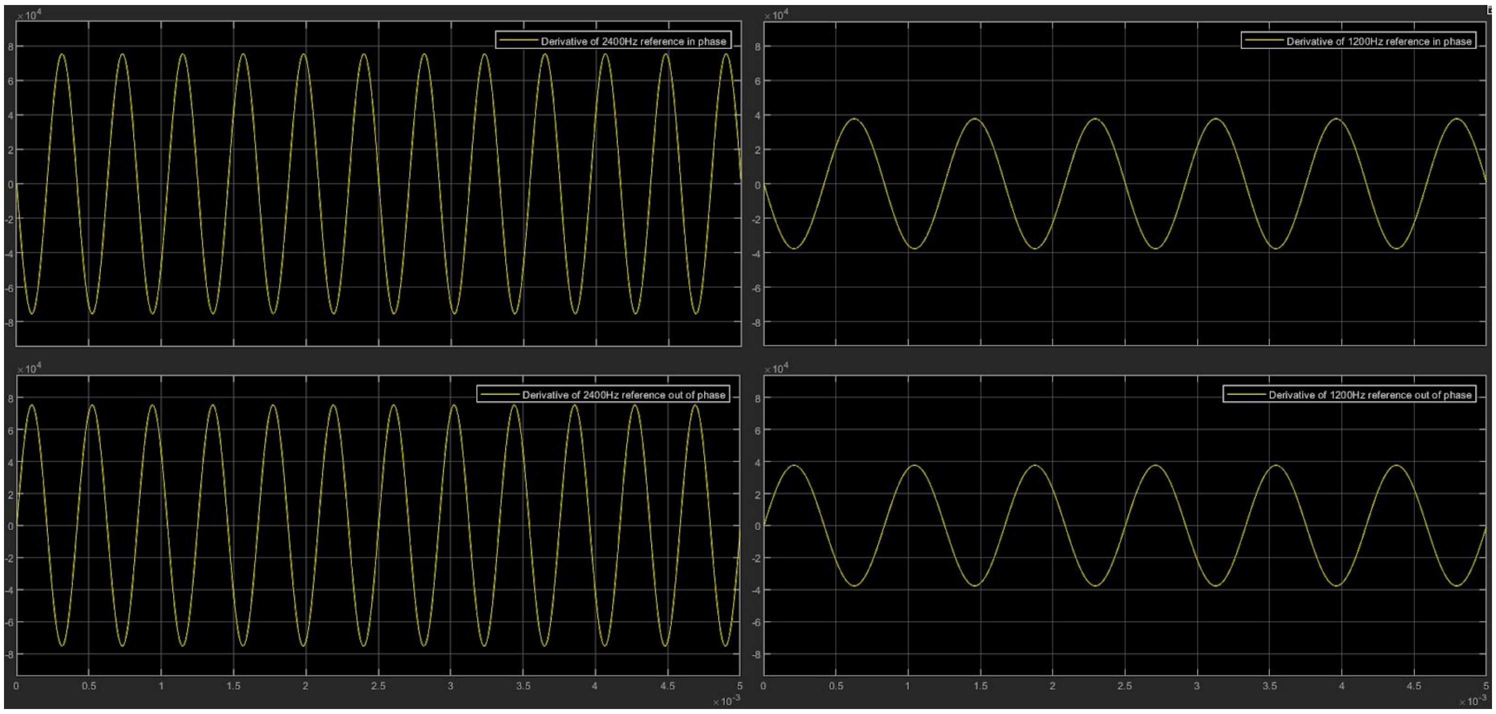


Limitations - The algorithm to find local extrema is not robust for double datatype which is encountered in analog signals.

3. Continuously comparing the derivative

Four reference signals were generated, two each corresponding to a single frequency tone. Each of these two signals is separated by a phase shift of 90 degrees. The continuous-time derivatives for the MSK wave as well as for each of these reference signals were then computed. Each of the derivatives for the references was continuously compared with that of the MSK's derivative. Since at the end of every interval of length T_b the MSK wave either changes in frequency or stays the same, the derivative could be

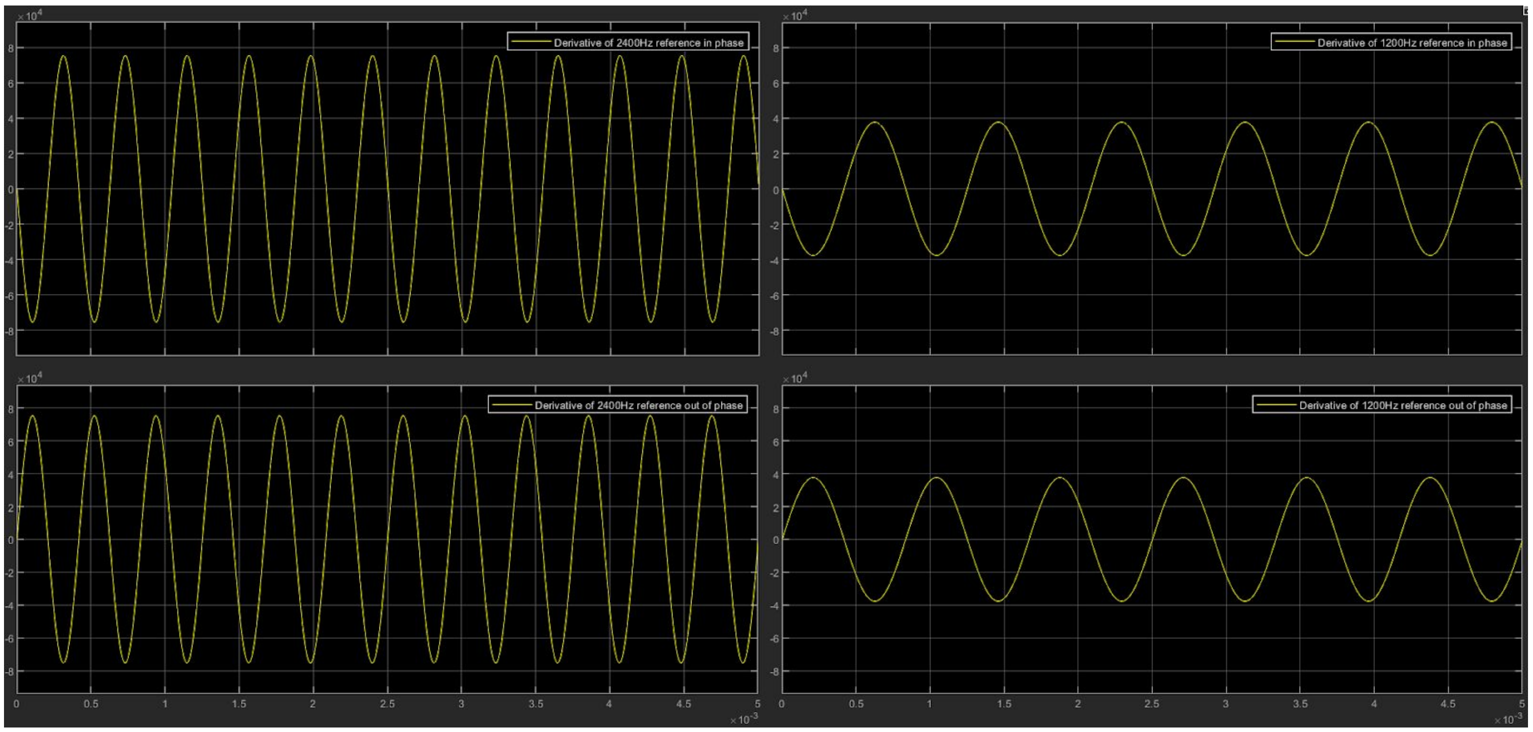




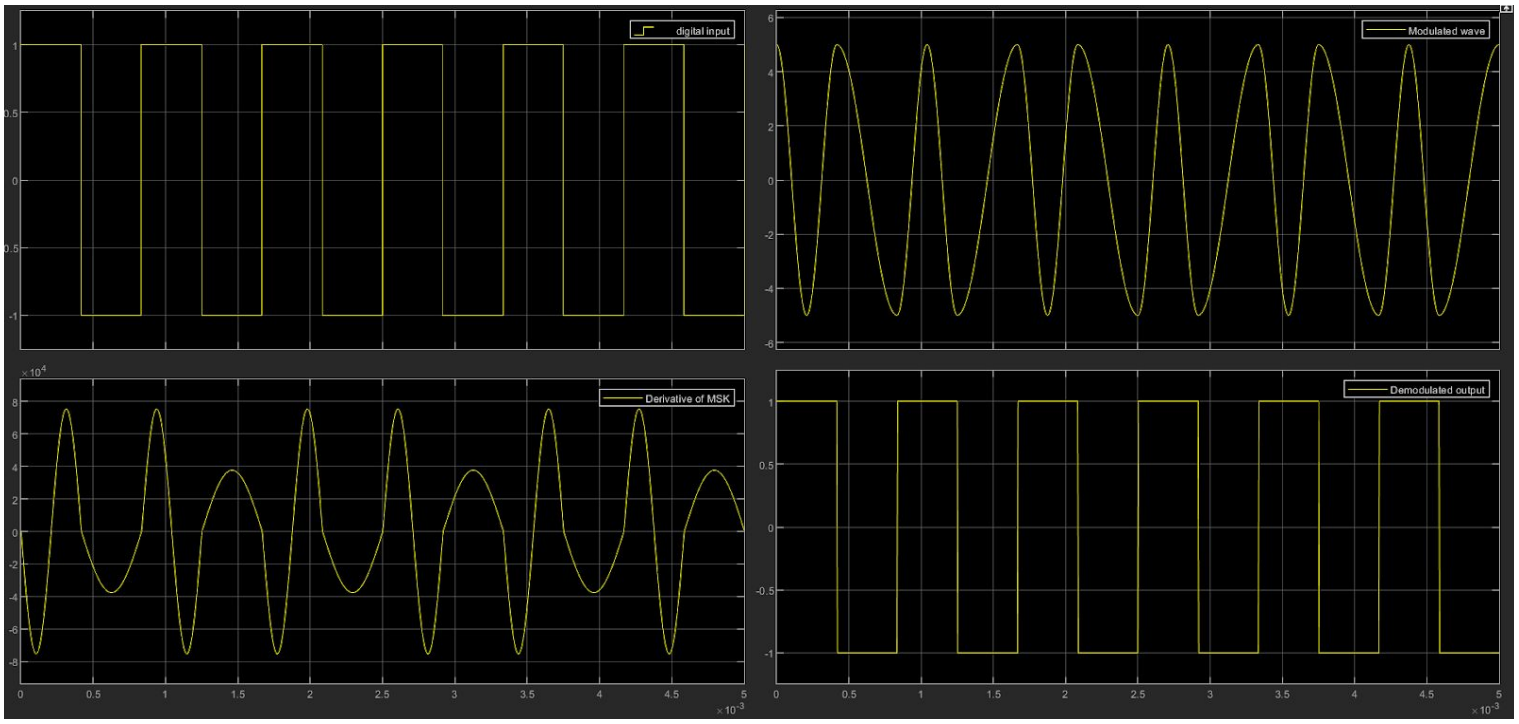
checked for equality among that of the four reference signals. A larger absolute derivative corresponds to a higher frequency component and vice versa.

The output is as follows:

The first figure shows the derivatives of the four reference signals generated to compare with



The second displays the digital input, modulated wave, derivative of modulated wave and demodulated output.



Limitations - A phase-locked loop needs to be established before this approach can be used.

4. Passing the MSK modulated wave through a low-pass filter

The MSK modulated wave is passed through a suitably designed low-pass filter in order to filter out and increase the strength of low-frequency components while simultaneously attenuating that of the higher frequency components. Butterworth, Chebyshev and Bessel filters were used with varying degrees of success.

a) Butterworth filter:

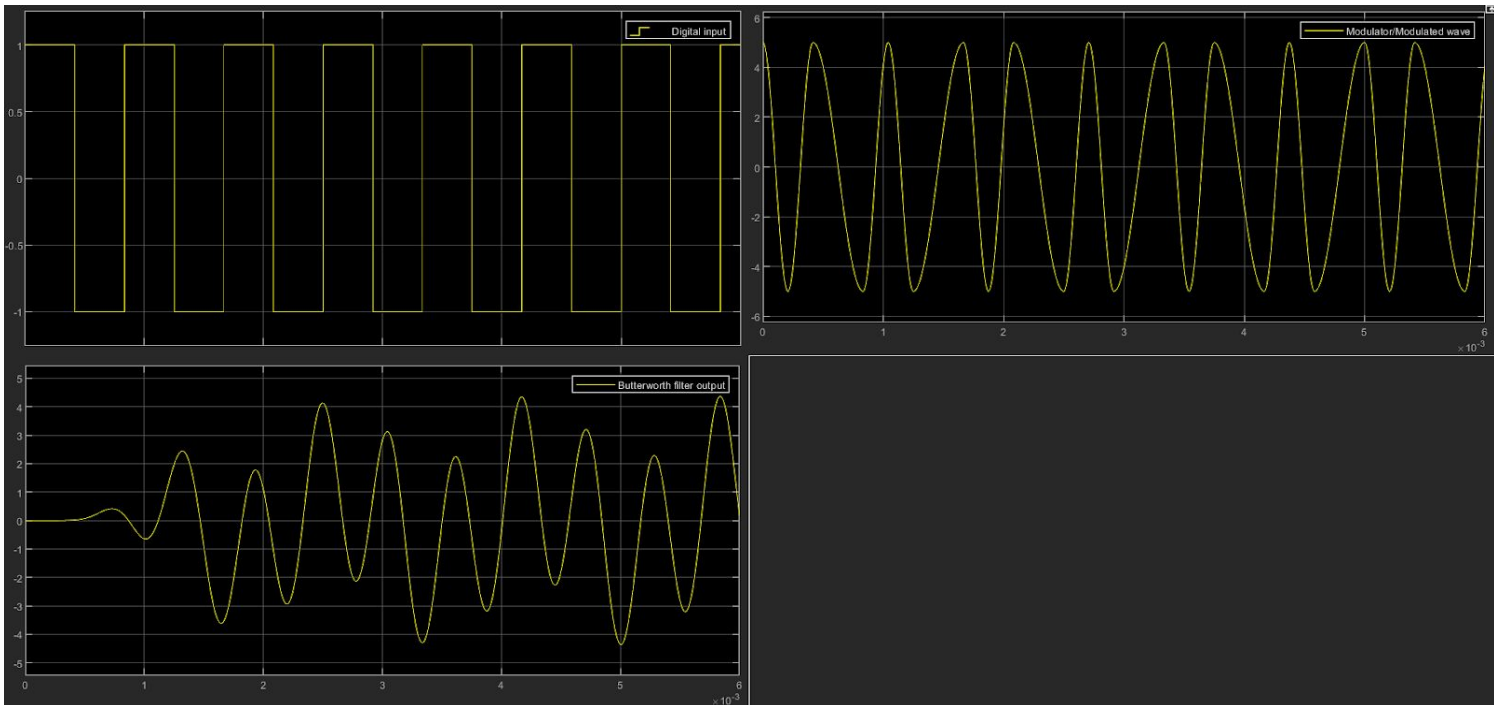
The frequency response of a Butterworth filter is maximally flat in the passband and rolls off towards zero in the stopband. Given that this type of filter has a slow roll-off in the transition band when compared to other types of filters, this filter was not suitable for the current application.

Butterworth specifications:

Type - Low-pass

Filter order - 13

Passband edge frequency (Hz) - 1800Hz



b) Chebyshev Type I/II filter:

Chebyshev filters are analog/digital filters having a steeper roll-off than Butterworth filters and have a passband ripple(type I) or stopband ripple(type II). Given the inherent nature of this filter which includes ripples either in the passband or stopband, irregularities were observed in the filtered MSK modulated wave.

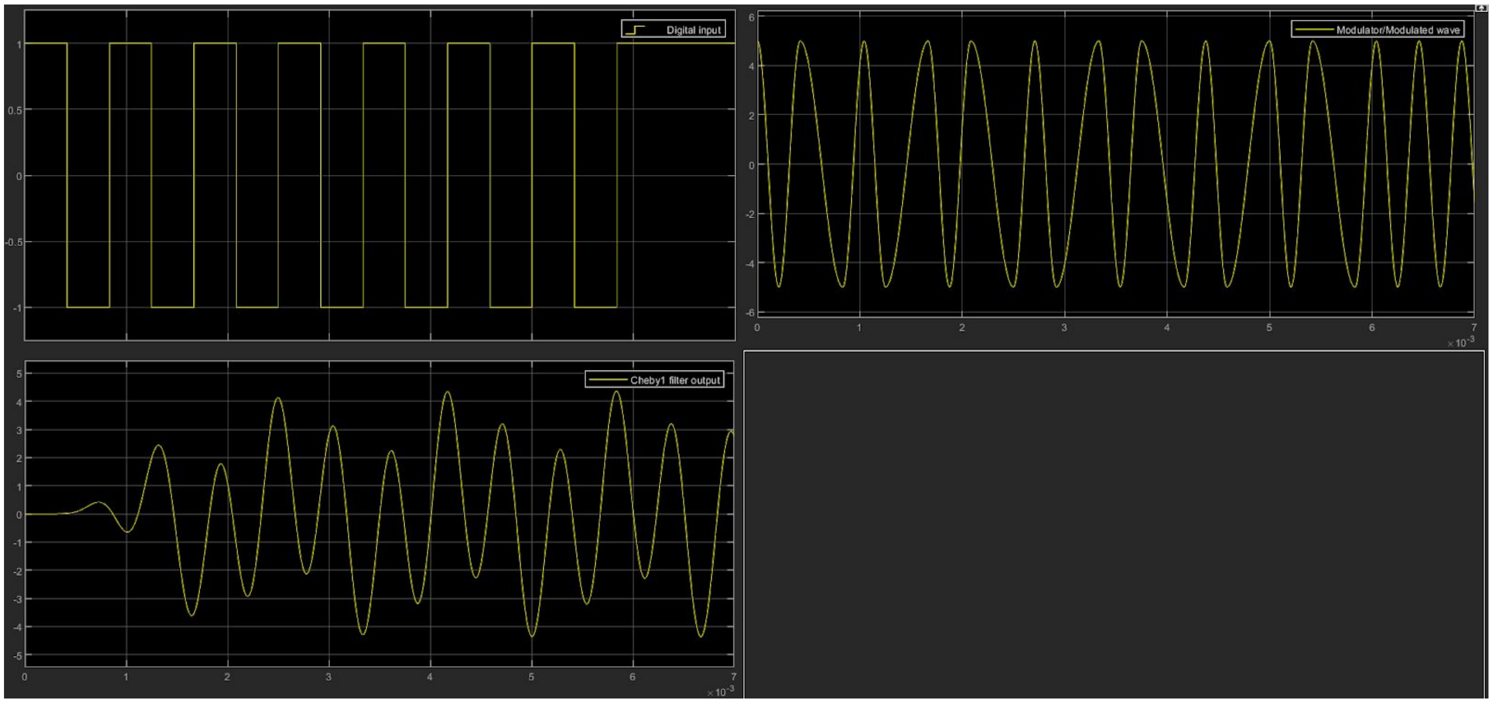
<u>Chebyshev Type 1 specifications:</u>

Type - Low-pass

Filter order - 15

Passband edge frequency (Hz) - 1800Hz

Passband ripple (dB) - 6dB



c) Bessel filter:

Bessel filter is a type of analog linear filter with a maximally flat group/phase delay, which preserves the wave shape of filtered signals in the passband. This allows the preservation of the shape of a multi-frequency signal such as in data transmission. Hence this filter was the most appropriate to choose for demodulation.

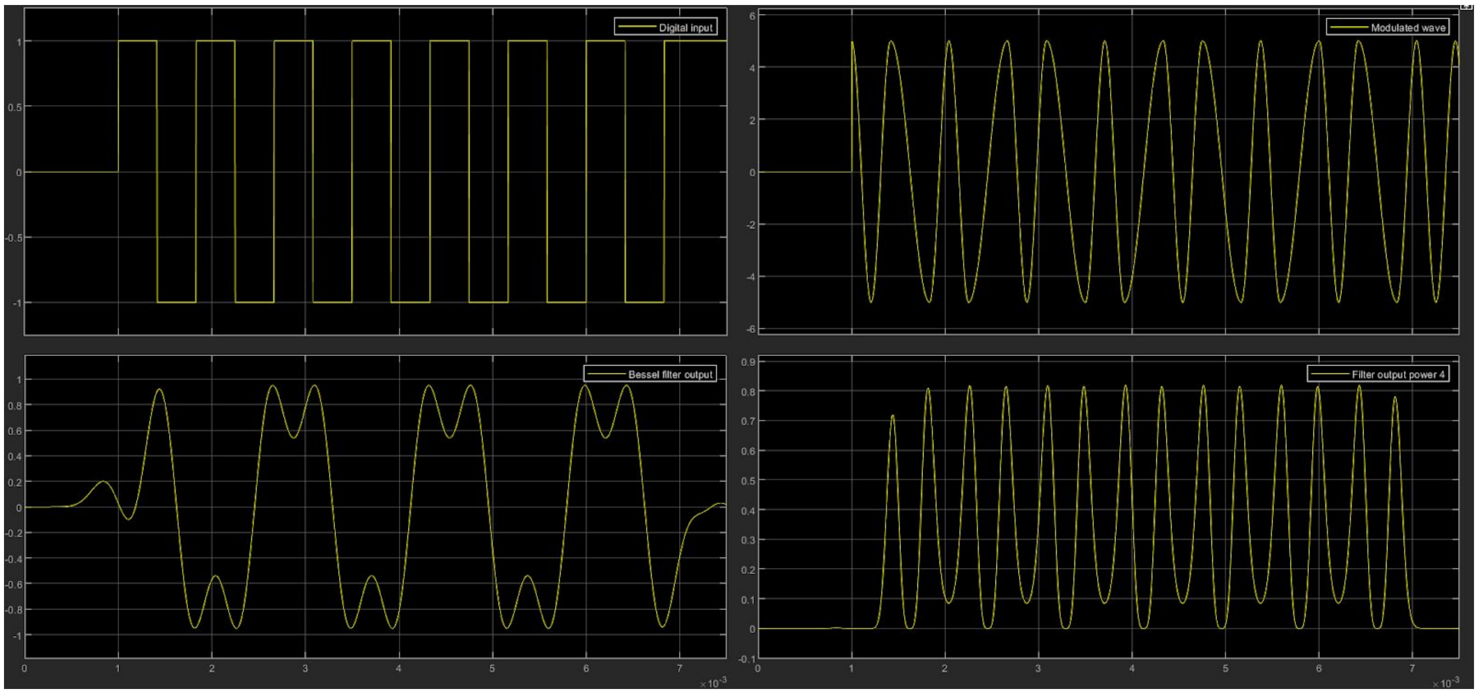
<u>Bessel specifications:</u>

Type - Low-pass

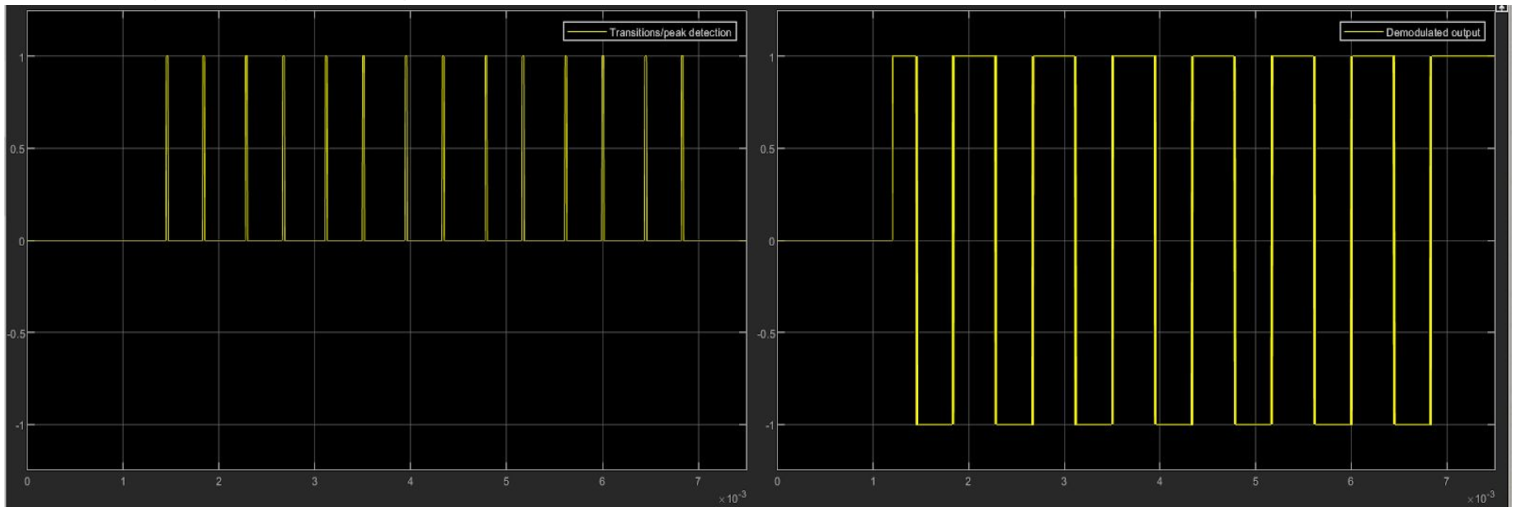
Filter order - 15

Passband edge frequency (Hz) - 1800Hz

Following this, the filter's output was raised to the 4th power in order to suppress the magnitudes of the attenuated frequency components less than 1 and to amplify that for lower frequency components.



A MATLAB function block was added to apply the peak finding algorithm restricted to local maxima less than 10 so as to ignore the amplified peaks of the lower frequencies. It was observed that these lower magnitude peaks corresponded to a shift in frequency. Hence, what remained was to identify the first bit in the transmission correctly. This first bit could then be complemented at the occurrence of the above-mentioned peaks thereby generating the output bitstream.



The above graph highlights the points at which the small peaks were detected

The code of the MATLAB function block used in the demodulator can be found in the annexure (d).

As can be seen from the graph, a delay of 0.975ms exists in the output of the demodulator. Slight instabilities can also be observed at the transition points in the output of the demodulator. The delay occurs because the filter requires a certain amount of data before it can begin processing it. Instabilities around the transition points occur due to the nature of double datatype when comparing values extremely close to each other.

Result:

MSK modulation is achieved where the input data is generated using a pseudo-random number generator.

The MSK wave generated is used as an input to the demodulator to retrieve back the original data stream.

Annexure:

a) MATLAB function file for modulator:

Filename - modulateMSK.m

```
function [signal, T, a, sampled] = modulateMSK(Tb, Rb, fc, Fs)
    a = randi([0, 1], 1, 15);
    a(a == 0) = -1;
    n = 0 : 1 : length(a)-1;
    signal = [];
    T = [];
    theta = 0;
    sampled = [];

    for i=n
    t = i*Tb:1e-6:(i+1)*Tb;
    %disp(length(t));
    k = s(t,Rb,Tb,fc,i,a(i+1),theta);
    signal = [signal k];
    T = [T t];
    Ts = 1/Fs;
    Time = i*Tb:Ts:(i+1)*Tb;
    sampled = [sampled, s(Time,Rb,Tb,fc,i,a(i+1),theta)];
    theta = theta + a(i+1)*pi/2;
    end
end

function y = s(t,Rb,Tb,fc,n,a,theta)
    y = 5*cos(2*pi*fc*t + 2*pi*a*(Rb/4)*(t-n*Tb) + theta);
end
```

b) MATLAB script for FFT on frames (approach #1):

Filename - demodApproach1.m

```
clc; clear; close;
Rb = 2400;
Tb = 1/Rb;
fc = 1800;
Fs = 2500000;

[signal, T, a, sampled] = modulateMSK(Tb, Rb, fc, Fs);
a(a == -1) = 0;

square = signal.*signal;
square1 = sampled.*sampled;
disp(length(sampled));
time = (0:length(sampled)-1)*(1/Fs);


part_one = square1(1:1000);
time_1 = time(1:1000);
time_1 = Fs*(0:1000-1)/1000;
fourier = abs(fft(part_one)/1000);

subplot(2,2,4);
plot(time_1,fourier);
xlabel('frequency bins');
ylabel('magnitude');
title('FFT output');
xlim([-.5e6 3e6]);

subplot(2,2,1);
time = (0:length(a)-1)*Tb;
stairs(time,a);
ylim([-.5 1.5]);
xlabel('Time(s)'); ylabel('Amplitude'); title('Input digital signal wave');

subplot(2,2,3);
plot(T,square);
title('Squared signal');

subplot(2,2,2);
```

```
plot(T,signal);
xlabel('Time(s)'); ylabel('Amplitude'); title('MSK Modulated wave');
```

c) MATLAB script for time differences between peaks and valleys (approach #2):

Filename - demodApproach2.m

```
clc; clear; close;
Rb = 2400;
Tb = 1/Rb;
fc = 1800;
Fs = 2500000;

[signal, T, a, sampled] = modulateMSK(Tb,Rb,fc,Fs);
a(a == -1) = 0;

[pospeaks, pospindices] = findpeaks([0, signal, 0]);
[negpeaks, negpindices] = findpeaks([0, signal.*(-1), 0]);
negpeaks = negpeaks.*(-1);

if pospindices(end) > length(T)
    pospindices(end) = length(T);
end
if negpindices(end) > length(T)
    negpindices(end) = length(T);
end

if length(negpeaks) < length(pospeaks)
    indices = [pospindices; negpindices 0];
    peaks = [pospeaks; negpeaks 0];
    indices = indices(:)';
    indices = indices(1:end-1);
    peaks = peaks(:)';
    peaks = peaks(1:end-1);
else
    indices = [pospindices; negpindices];
    peaks = [pospeaks; negpeaks];
    indices = indices(:)';
    peaks = peaks(:)';
end

tdiffs = diff(T(indices));
tdiffs = round(tdiffs.*1e3, 1);

tstamps = zeros(1, 15);
output = zeros(1, 15);

k = 1;
j = 1;
N = length(tdiffs);
while j <= N
    curdiff = tdiffs(j);
    if j < N
        nextdiff = tdiffs(j + 1);
    end
    if curdiff == 0.2 && nextdiff == 0.2
        output(k) = 1;
        tstamps(k) = T(indices(j));
        j = j + 2;
    elseif curdiff == 0.4
        output(k) = 0;
        tstamps(k) = T(indices(j));
        j = j + 1;
    end
    k = k + 1;
end

square = signal.*signal;

subplot(2,2,1);
time = (0:length(a)-1)*Tb;
stairs(time,a);
ylim([-.5 1.5]);
xlabel('Time(s)'); ylabel('Amplitude'); title('Input digital signal');

subplot(2,2,2);
plot(T,signal);
xlabel('Time(s)'); ylabel('Amplitude'); title('MSK Modulated wave');
hold on;
subplot(2, 2, 2);
plot(T(indices), peaks, 'x');
```

```
legend('MSK', 'local extrema');
hold off;

disp(time);
disp(tstamps);
subplot(2, 2, 3);
stairs(tstamps, output);
ylim([-.5 1.5]);
xlabel('Time(s)');
ylabel('Demodulated wave output');
title('Output digital signal');
```

d) MATLAB script for the function block 1 used in demodulator approach #4:

```
function [y, shift]  = fcn(time, prevbit, t, u, v)

%for finding local maximas
fdiff = t - u;
sdiff = u - v;

%for finding the first bit
if (prevbit == 0)
    if (u > 0.4 && time < 1.2e-3)
        k = -1;
    elseif (time > 1.2e-3)
        k = 1;
    else
        k = prevbit;
    end
    shift = 0;

%last clause in if is to handle edge cases where there might be extremely
%small local maximas less than 10e-6
elseif (fdiff < 0 && sdiff > 0 && u < 2 && max(abs(fdiff), abs(sdiff)) >
1e-5 && time > 1.3e-3)
    k = transition(prevbit);
    shift = 1;
```

```
else
    k = prevbit;
    shift = 0;
end

y = k;

function nextbit = transition(prevbit)
    if (prevbit == 0)
        nextbit = prevbit;
    elseif (prevbit < 0)
        nextbit = 1;
    else
        nextbit = -1;
    end
```