Single-Sideband-Modulation

Table of Contents

Test Signal

```
fs = 48e3; % Abtastfrequenz in Hz
fc = 1000; % carrier frequency in Hz

t = 0:1/fs:.1-1/fs; % chirp
%t = 0:1/fs:.8008-1/fs; % maybe next time
%t = 0:1/fs:4.5938-1/fs; % longer signal

input = audioread("maybe-next-time.wav");
x_test = round((2^15-1)*input); %maybe next time
% x_test = round((2^15-1)*chirp(t,100,.1-1/fs,1000));  %input signal
%x_test = fi(chirp(t,100,.1-1/fs,1000),1,16,15);  %input signal
%x_test = round((2^15-1)*input_example); %longer example from emil

% LUTs for sine and cosine
LUT_cos=round((2^7-1)*cos(2*pi*(fc/fs)*(0:48-1)));
LUT_sin=round((2^7-1)*sin(2*pi*(fc/fs)*(0:48-1)));
```

Filter-Design of Hilbert filter

```
N = 128; % filter order
Hd = designfilt('hilbertfir', 'FilterOrder', N, ...,
                          'TransitionWidth', 0.05, ...
                          'DesignMethod', 'equiripple');

% frequency response of Hilbert filter
%freqz(Hd);

% filter coefficients
b_k_test = round((2^15-1)*Hd.Coefficients);
```

initialize needed arrays

```
x_zwsp=zeros(1,filtord(Hd));
G = filtord(Hd)/2; % Filter delay
x_tilde_test=zeros(1,length(x_test));
x_delayed_zwsp = zeros(1,length(x_test)-G);
```

```
x_delayed_test = zeros(1,length(x_test));
y=zeros(1,length(x_test));
index_LUT = uint8(1);
```

For loop simulation of sample by sample

```
for n=1:length(x_test)
% shift register for Filter
    for k=length(x_zwsp):-1:2
        x_zwsp(k)=x_zwsp(k-1);
    end
    x_zwsp(1)=x_test(n);
% calculate filtered values with difference equation
    test_zwsp=0;
    for k=1:length(x_zwsp)
        temp=b_k_test(k)*x_zwsp(k);

        test_zwsp = test_zwsp + round(temp/2^15);
        %test_zwsp = round(test_zwsp/2^8);
    end
    x_tilde_test(n)=test_zwsp;
% delay line
    x_delayed_zwsp(n) = x_test(n);
    if n > G
        x_delayed_test(n) = x_delayed_zwsp(n-G);
    end
% multiply with sine/cosine and add signals
    y(n) = x_delayed_test(n)*LUT_cos(index_LUT) -
 x_tilde_test(n)*LUT_sin(index_LUT);
% calculate index for LUT
    index_LUT = index_LUT+1;
    if index_LUT > 48
        index_LUT = 1;
    end
end
```

Output

```
%round y
y=round(y);
y=y./(2^23);

% Power Spectrum
figure;
periodogram(y, [], 4096, fs, 'power', 'centered');
title('Periodogramm des modulierten Signals (mit Cosinus-Array)');
ylim([-200 0]);
xlim([-10 10]);

% Spectrogramm
figure;
spectrogram(y, hamming(512), [], [], fs, 'yaxis');
title('Spektrogramm des modulierten Signals (mit Cosinus-Array)');
```

```
ax = gca;
ax.YLim = [0 3];
```

Published with MATLAB® R2021b