Table of Contents

Simple FM radio stereo receiver

```
%RTL_BANK
%
% Stereo FM receiver, no de-emphasis filter
% Frequency domain filtering using a filter bank
% By R.W.

clear all, close all

% Sampling rate and decimation for the stereo reception
% The sampling rate should be multiple of the two-sided bandwidth 30 KHz of
% one FM channel
NDEC=8;
FESR=30e3*NDEC;

% Script to define some essential parameters and functions you need to
% figure yourself. The script is included as a p-code file.
% Contains the design of the low-pass filter FLOW and the filter bank
% function rw_bank()
FLOW = rw_ini_bank();
```

Radio parameters

```
% YLE 1
%expFreq = 87.9e6;
% YLE Puhe
expFreq = 103.7e6;
% YLE Radio Suomi
%expFreq = 94e6;
% Radio Dei: 89e6, radio Helsinki 89.7
%expFreq = 89e6;


% nSample = #Samples to read at one round. Without overlap-save or overlap-
add,
% the FFT size is nSample as well so nSample should not be too large.
nSample = 2048*NDEC;

% The filter can be displayed by
% fvtool(FLOW,1,'Fs',FESR)

% Number of vectors/frames to read
```

```
nFrame = 2e2;

% Your dongle's frequency offset correction here
% Parts per million = frequency_offset/carrier_frequency * million
% Must be integer
PPM = 38;

hSDRrRx = comm.SDRRTLReceiver(...
    'RadioAddress', '0',...
    'CenterFrequency',     expFreq, ...
    'EnableTunerAGC',      true, ...
    'SampleRate',          FESR, ...
    'SamplesPerFrame',     nSample, ...
    'FrequencyCorrection', PPM, ...
    'OutputDataType',      'double')
fprintf('\n')

hSpectrumAnalyzer = dsp.SpectrumAnalyzer(...
    'Name',               'FM signal',...
    'Title',              'FM signal', ...
    'SpectrumType',       'Power density',...
    'FrequencySpan',      'Full', ...
    'SampleRate',         FESR, ...
    'YLimits',            [-50,0],...
    'SpectralAverages',   10, ...
    'FrequencySpan',      'Start and stop frequencies', ...
    'StartFrequency',     -60e3, ...
    'StopFrequency',      60e3,...
    'Position',           figposition([50 30 30 40]));



% Audio object to listen to the radio
% Max. sampling frequency depends on the hardware
%hAudio = audioDeviceWriter(FESR/NDEC,'BufferSize',ceil(nSample*2/NDEC));
hAudio = audioDeviceWriter(FESR/NDEC);
% List available audio outputs
%getAudioDevices(hAudio);
```

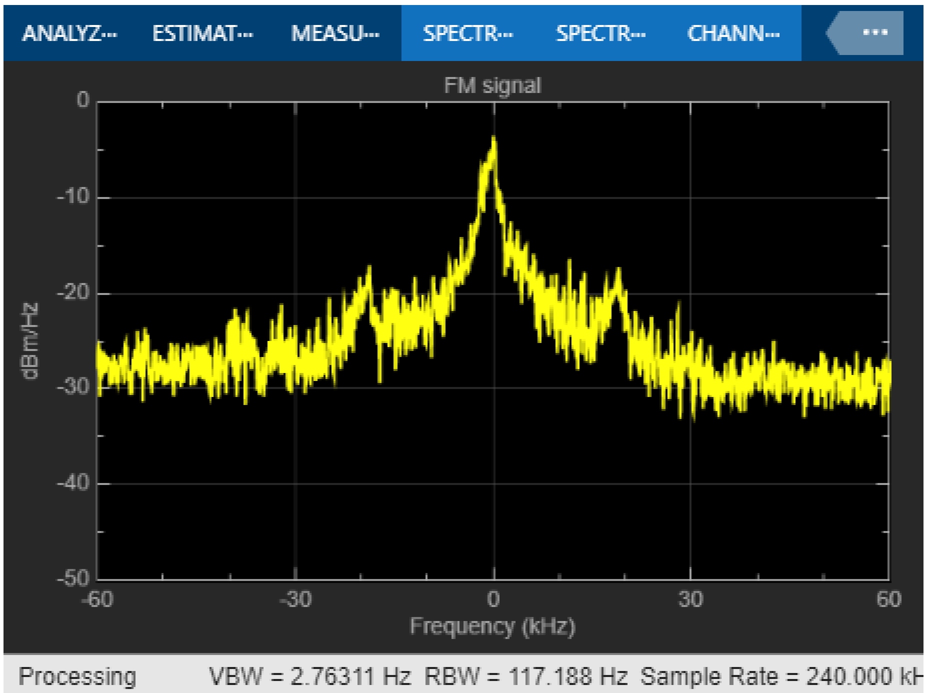
```
hSDRrRx =

  comm.SDRRTLReceiver with properties:

           RadioAddress: '0'
        CenterFrequency: 103700000
         EnableTunerAGC: true
             SampleRate: 240000
         OutputDataType: 'double'
        SamplesPerFrame: 16384
    FrequencyCorrection: 38
        EnableBurstMode: false
```

Stream Processing

```
if isempty(sdrinfo(hSDRrRx.RadioAddress))
    error(message('SDR:sysobjdemos:MainLoop'))
end

fprintf('Receive time %f [s]   \n', nSample/FESR*nFrame)

% Record the wall clock time of the loop
tic;
% Run as real time as possible. Variables needn't declared bu don't
% change the size of the array within the loop
for iFrame = 1:nFrame
    rxSig = step(hSDRrRx);
    rxSig = rxSig - mean(rxSig);  % Remove DC component

    % Display received frequency spectrum
    hSpectrumAnalyzer(rxSig);

    % Demodulate the FM modulated signal. The output must be the same size
    % as the input. The same receiver as in Exercise 3.
    fmSig = rw_fmrx(rxSig);

    % The steps in rw_bank():
    %
    % Calculate FFT of fmSig and FLOW. Zero-pad the both sequencies. Lenght
    % of the zero+pad should be a factor of NDEC.
    % Make sure fmSig and FLOW are both column vectors.
    %
    % Make the filterbank: the FFT of FLOW + its circular shift in frequency
 domain.
    % The circularly shifted part filters the L-R channel. Use circshit()
    %
    % Multiply the FFT of fmSig with the filter bank
    %
    % Extract the two channels. This makes decimation in frequency domain
    % The two-sded bandwidths of the channels are FESR/NDEC=30KHz.
    % Note that without fftshift() in the first channel half of the samples
 are
    % in the beginning of the vector and the other half in the end
    %
    % Take ifft() of the L+R and L-R channels and make the output real() to
 take care
    % of the numerical errors
    %
    % Return the two FM channels in time domain. The sampling rate in each
    % channel is 30 KHz. Column vectors.
    [lpSig1,lpSig2] = rw_bank(fmSig, FLOW, FESR);

    % Extract the stero channels, two column vectors into a matrix
    aSig=[lpSig2+0.5*lpSig1,0.5*lpSig1-lpSig2];
    % Mono only
    %aSig = lpSig1;
```

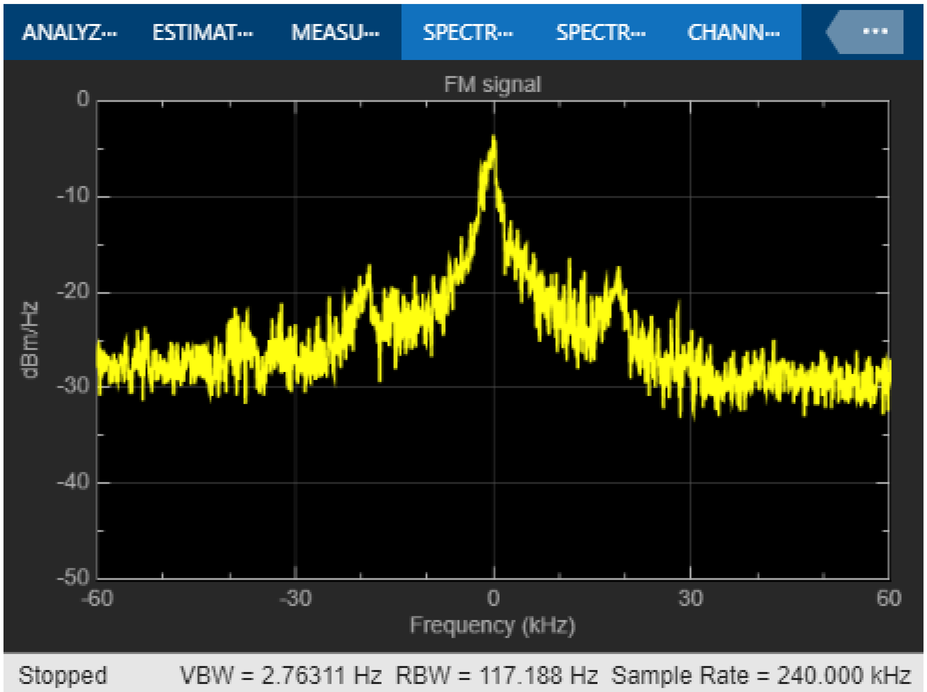
```
    % Underrun may occure in the loop
    % Arbitrary scaling of the signal amplitude
    nUnderrun = hAudio(0.5*aSig);

    if nUnderrun > 0
        fprintf('Audio player queue underrun by %d samples.\n',nUnderrun);
    end
end

fprintf('Clock receive time %f [s]\n', toc)
```

Receive time 13.653333 [s]
Clock receive time 18.944525 [s]



Release all System objects

```
release(hSDRrRx);
clear hSDRrRx
release(hAudio); release(hSpectrumAnalyzer)
```



Published with MATLAB® R2023a