1 Introduction

Receiver

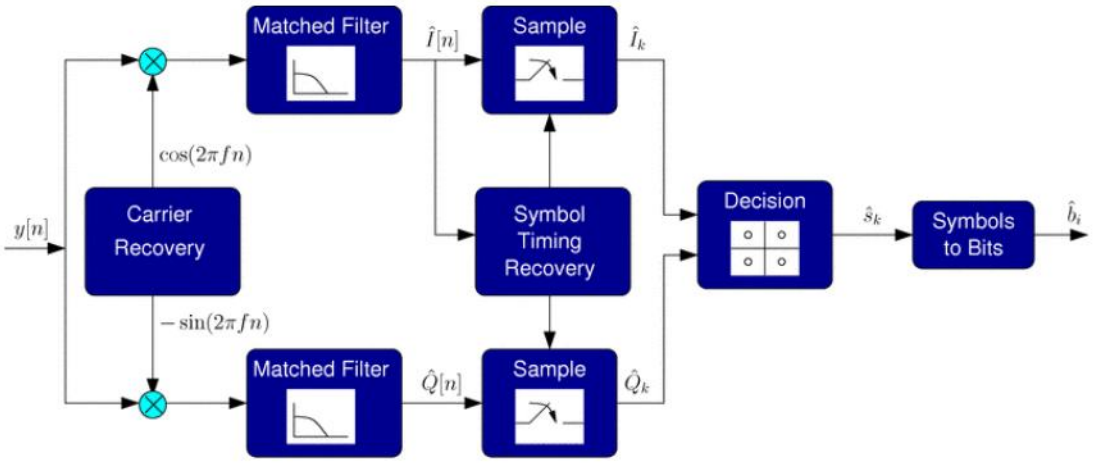

Figure 1 Receiver chain

The basic operation of receiver is shown in figure 1. The receiver follows the steps:
1. Multiplication by the conjugate of carrier to shift the signal to baseband. (Zero center frequency.)
2. The signal at double frequency is removed by matched filter.
3. Sampled at optimal points and estimates of I/Q symbol weights are obtained.
4. The symbol decision is made.
5. Conversion of symbols to bits

Transmission Signal

4PSK modulation is used by multiplying a complex carrier as seen in figure 2. These symbols is Gray coded: two adjacent symbols values differ in only one bit (binary digit). Gray coding is used because it gives small errors.

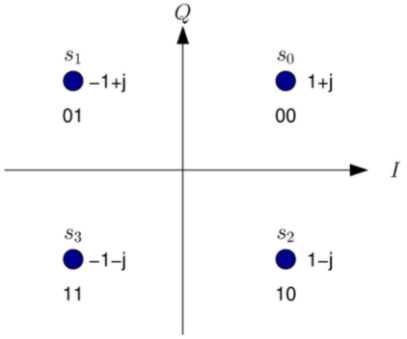

Figure 2 4PSK symbol

Digital Down Converter (DDC)

Digital down converter converts a digitized, band limited signal to a lower frequency signal at a lower sampling rate to simplify the subsequent radio stages. In this lab, the DDC is done by converting intermediate frequency (IF) down to a complex baseband signal (I/Q samples). For converting, the following steps are used:

1. Estimation of carrier signal by pilot tone
For this step, first the signal is passed through a bandpass filter near 8kHz. Second, the PLL locks the pilot tone at 8kHz and outputs the accumulator generating sin and cos components at 24 kHz.
2. Multiplication of the signal by the estimated carrier
This process shifts the signal to complex baseband.
3. Matched Filtering
By passing through a low pass filter, the double frequency components and the pilot tone is removed giving the Q/I symbols.

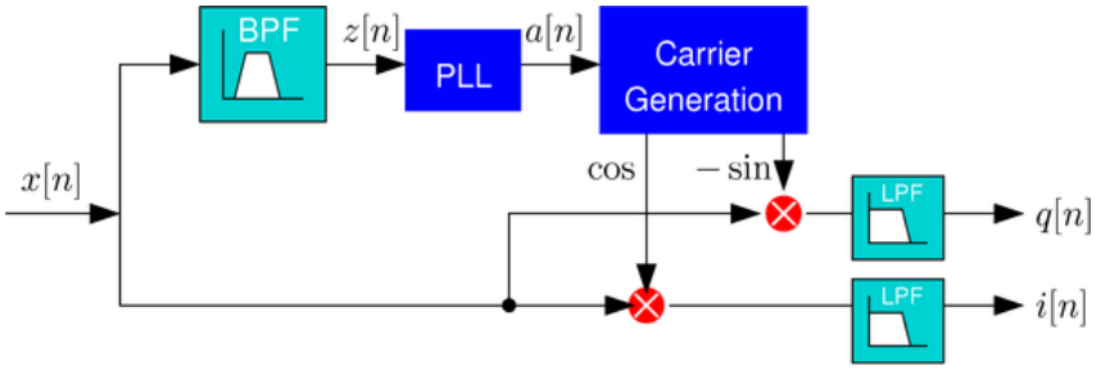

Figure 3 Diagram of DDC

2 Execution / Evaluation

Check – off

Demonstrate to the TA or instructor that your DDC is working correctly. Be prepared to answer any questions about the operation of your code and how to tell if the output is correct

Shown in Lab write up

Lab Write up

(1) Answers to the pre-lab questions

1) What are the two main components that you must implement to create the digital receiver?

Demodulator, Decoder

2) What is the basic function of a digital down converter?

Digital down converter converts a digitized, band limited signal to a lower frequency signal at a lower sampling rate to simplify the subsequent radio stages. In this lab, the DDC is done by converting intermediate frequency (IF) down to a complex baseband signal (I/Q samples).

3) What do I and Q stand for?

I: In phase

Q: Quadrature phase

4) How are I and Q related to a complex modulation signal?

I: Real part of the complex modulation signal

Q: Imaginary part of the complex modulation signal

5) What is the bandwidth and center frequency of our transmission signal?

Bandwidth: 16kHz

Center frequency: 24kHz

6) What kind of modulation does our transmitter use?

I/Q Complex modulation, 4PKS(QPSK) constellation.

7) What is the bitrate of our transmission system?

16kbit/s

8) What is the purpose of the 8 kHz tone in the transmission?

For synchronization, the pilot tone locked to the carrier is generated at 8 kHz.

9) What is the PLL used for in the DDC?

The pilot tone outputs the accumulator and generate sin cos components of the carrier at three times the frequency. (24kHz)

10) What are the multiply and LPF operations for in the DDC?
The recovered carrier is multiplied to the signal which shifts the signal to the complex baseband. The LPS removes the double frequency components from the multiply and the pilot tone, giving Q/I symbols.

11) The function system_param stores the parameters for system operation in what type of MATLAB variable?
Global structure variable

12) Modular operational blocks in MATLAB can be accomplished by passing the ***_state variable_*** *as an input and output structure variable.*
13) True or False: Different state variables should be used for different instances of a given processing block?
True. Because it resets the values to the initial function for the next buffer to continue and keep tracking.

14) Why do we have to compensate for the delay of the bandpass filter used to isolate the 8 kHz tone?
Because after the down conversion, there is a delay caused by the bandpass filter.

15) When testing the DDC, why do we initially have to allow several blocks to process before getting correct output?
This is to allow the PLL to track the phase properly.

(2) A printout of your final MATLAB implementation of the DDC

PLL function

```
function [ref_out, accum_out, state_out] = pll(ref_in, state_in);

% [ref_out] = pll(ref_in, state_in);
% Does PLL tracking of the input waveform.  Operates on complete waveform.
% Inputs:
%   ref_in      Input reference
%   state_in    State and parameters
% Outputs:
%   ref_out     Output reference
%   accum_out   Output accumulator

% Get parameters
state = state_in;

f0 = state.f0;
K = state.K;
a = state.a;
b = state.b;

N = length(ref_in);

ref_out = zeros(N, 1);
accum_out = zeros(N, 1);
%% Estimate amplitude of block
amp_est = mean(abs(ref_in))*(pi/2);
%% Get accumulator
accum = state.accum;
%% PLL
for n=1:N,
  % Multiply
  z(n) = state.ref_in_prev * state.ref_out_prev / amp_est;
  % Loop filter
  v(n) = state.a(1)*state.v_prev + state.b(1)*z(n) + state.b(2)*state.z_prev;

  state.z_prev = z(n);
  state.v_prev = v(n);
  % VCO
  state.accum = state.accum + state.f0 - state.K*v(n)/(2*pi);
  state.accum = state.accum - floor(state.accum);

  accum_out(n) = state.accum;
  ref_out(n) = sin(2*pi*state.accum);
  state.ref_out_prev = ref_out(n);
  state.ref_in_prev = ref_in(n);
end
state_out = state;
```

ddc fucntion

```
function [Ib, Qb, state_out, d] = ddc(x, state_in);
% [Ib, Qb, state_out, d] = ddc(x, state_in);
% Operates on a single block of data to generate I and Q outputs.
% Inputs:
%   x   Data
% Outputs:
%   Ib, Qb  Down-converted outputs
%   d   Debug information

state = state_in;

% BPF to isolate tone
[t1 state.bpf_state] = fir(x, state.bpf_state);

% Send tone to PLL to get reference
[ref accum1 state.pll_state] = pll(t1, state.pll_state);

% Compensate for delay of BPF to get aligned carrier.
accum1 = accum1 + state.del;

% Generate sine and cosine waveforms.  Take multiple of frequency.
accum1 = accum1 * 3;
cos1 = cos(2*pi*accum1);
sin1 = sin(2*pi*accum1);

% Do down-conversion (multiply by carrier)
i1 = cos1.*x;
q1 = -sin1.*x;

% Matched filtering
[Ib state.lpf1_state] = fir(i1, state.lpf1_state);
[Qb state.lpf2_state] = fir(q1, state.lpf2_state);

% Return debug information if desired
if (nargout >= 4),
  d.accum = accum1;
  d.cos1 = cos1;
  d.sin1 = sin1;
  d.ref = ref;
  d.t1 = t1;
end
state_out = state;
```

Test_ddc code

```
% test_ddc.m
%   Tests the digital down-converter using a known test signal.
%% Block parameters
Ns = 100;   % Samples per block
n = [1:Ns];
%% System parameters
param = system_param;
%% Signal parameters

fs = param.fs; % Sample rate
f0 = param.f0; % Nominal Carrier frequency
ft = param.ft; % Tone frequency
fc = param.fc; % Symbol rate
cps = param.cps; % Cycles per symbol

% Amplitude of signal and tone
as = 1;
at = 1;

% Do repeated pattern over and over
Nframe = 100;
symb1 = [0 0 0 0 1 2 2 3 1 2 2 0 1 2 2 3 1 2 2 0 1 2 2 3 1 2 2 0 1 2 2 3];
% Replicate this frame over and Nframe times
symb = kron(ones(1, Nframe), symb1);

% Generate signal from symbols
[s s_debug] = make_signal_4psk(fs, f0, ft, cps, param.h_ps, as, at, symb);

% Add noise (start without noise)
%SNR = 20;
%vs = sqrt(mean(abs(s).^2));
%s = s + vs.*randn(length(s), 1)/10^(SNR/20);

% Compute number of blocks we have
Nb = floor(length(s)/Ns);

% Make signal into blocks
sb = reshape(s(1:Ns*Nb), Ns, Nb);

% DEBUG: Get components into blocks for debug.
% Tone signal alone
t_debug = reshape(s_debug.t(1:Ns*Nb), Ns, Nb);
% Carrier signal alone
c_debug = reshape(s_debug.c(1:Ns*Nb), Ns, Nb);
% Modulation signal alone
m_debug = reshape(s_debug.mod(1:Ns*Nb), Ns, Nb);

% DEBUG: Plot modulated signal
```

```
%for ii=1:Nb,
%  plot([1:Ns], real(m_debug(:, ii)), [1:Ns], imag(m_debug(:, ii)));
%  pause;
%end

% DEBUG: Reference design (ideal non-block operations)
%bb_ideal = conv(s.*conj(s_debug.c), param.ddc.h_lp);
%I_ideal = real(bb_ideal); I_ideal_b = reshape(I_ideal(1:Ns*Nb), Ns, Nb);
%Q_ideal = imag(bb_ideal); Q_ideal_b = reshape(Q_ideal(1:Ns*Nb), Ns, Nb);

% Process blocks
ddc_state = ddc_init(param.ddc);

for ii=1:Nb,
  % Get a single block
  x = sb(:, ii);

  % Digital down converter
  [Ib Qb ddc_state ddc_debug] = ddc(x, ddc_state);

  % DEBUG. Plot recovered carriers
  %plot(n, ddc_debug.cos1, n, real(s_debug.c((ii-1)*Ns+[1:Ns])));
  %plot(n, ddc_debug.sin1, n, imag(s_debug.c((ii-1)*Ns+[1:Ns])));

  % DEBUG. Plot I/Q Output vs. ideal conv operations
  %plot(n, Ib, n, I_ideal_b(:,ii));
  %plot(n, Qb, n, Q_ideal_b(:,ii));

  % DEBUG. Plot I/Q outputs
  plot(n, Ib, n, Qb);
  pause;
end
```

(3) A plot showing the simulated performance of the DDC, showing that the correct output is obtained. Please write a few sentences explaining how you can tell it is correct.

In the graph, 0, 1, 2, 3 symbol is shown.
The symbols can be read using the 4PSK constellation, which is indicated as:
0 symbol: positive I, positive Q
1 symbol: negative I, positive Q
2 symbol: positive I, negative Q
3 symbol: negative I, negative Q
Symbols 0 0 0 0 1 2 2 3 1 2 2 0 1 can be read according to the graph. The front part of the given symbol - symb1 = [0 0 0 0 1 2 2 3 1 2 2 0 1 2 2 3 1 2 2 0 1 2 2 3 1 2 2 0 1 2 2 3] – has the same symbol as the one in the graph. Showing that the code is working well.

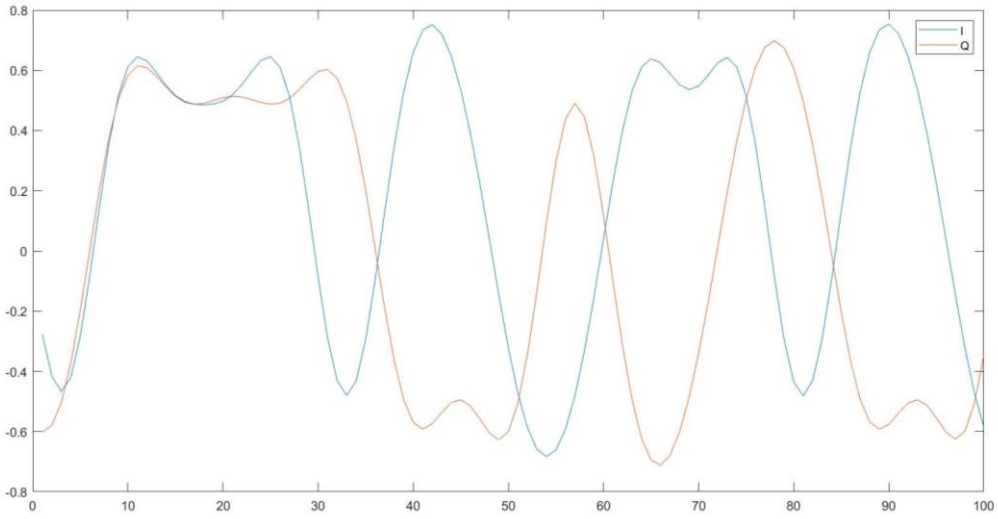

Figure 4 Plotted DDC

DDC design with noise
DDC graphed with noise 20 (SNR = 20) is shown in the figure below.

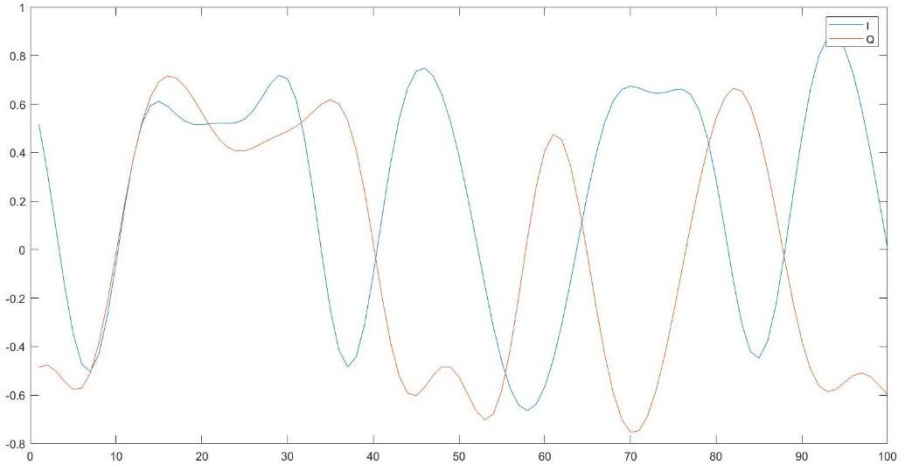

Figure 5 DDC with SNR 20

Even with noise, tracking is visible.

(4) An explanation of any problems you ran into implementing the DDC and how you found and fixed them.

There was a minor errors in the PLL regarding the ‘for’ loop. The ‘for’ loop index was mixed with ‘i’ and ‘k’ giving wrong graphs. This minor mistake was fixed quickly. However there were some troubles with the graph, and the no errors were found. This was fixed by moving the code to another folder. After fixing some errors, the code showed a proper graph.

3 **Conclusion**

The Digital down convertor (DDC) converts a signal with intermediate frequency down to a complex baseband signal. This DDC was coded using matlab. The code gave a graph showing the Q, I signal. The symbols can be read by analysing the graph using the 4PSK conversion table. As shown the lab write up section, the code showed a proper graph giving the proper symbols. Some errors were encountered because of mixed use of index in ‘for’ loop. This was fixed by checking the code over. Also, there were some errors regarding the computer. This was fixed by moving the whole code to another folder. Afterwards, proper graph was managed to be drawn.

4 **References**

[1] http://dsp-fhu.user.jacobs-university.de/?page_id=234
[2] “Digital down Converter.” Wikipedia, Wikimedia Foundation, 21 Jan. 2018, en.wikipedia.org/wiki/Digital_down_converter.
[3] “Gray Code.” Wikipedia, Wikimedia Foundation, 20 Nov. 2018, en.wikipedia.org/wiki/Gray_code.