Receiver Overview and DDC

DSP lab Spring 2013

4/30/2013
Jacobs University Bremen
Grishma Raj Pandeya, Joshan Chaudhary

PRELAB

1. What are the two main components that you must implement to create the digital receiver?
   Carrier Recovery, Symbol Timing Recovery.

2. What is the basic function of a digital down converter?
   To shift the signal to its baseband frequency and separate the in-phase and quadrature components.

3. What do I and Q stand for?
   I stands for in-phase (real part) and Q stands for Quadrature (imaginary part) of the symbol

4. How are I and Q related to a complex modulation signal?
   I gives the real part and Q gives the imaginary part of the complex baseband of the signal

5. What is the bandwidth and center frequency of our transmission signal?
   BW = 16 KHz, Center frequency = 24 KHz

6. What kind of modulation does our transmitter use?
   Complex modulation given by the 4PSK constellation

7. What is the bitrate of our transmission system?
   16 kbit/s

8. What is the purpose of the 8 kHz tone in the transmission?
   For synchronization which helps in carrier estimation

9. What is the PLL used for in the DDC?
   The PLL locks to the pilot tone and outputs the accumulator, which is used to generate sine and cosine of the carrier at three times the frequency .

10. What are the multiply and LPF operations for in the DDC?
    Multiply shifts the signal to complex baseband, and the LPF operation removes the doubled frequency components generated from the multiply as well as the pilot tone

11. The function system_param stores the parameters for system operation in what type of MATLAB variable?
    In Matlab struct

12. Modular operational blocks in MATLAB can be accomplished by passing the ___________________ as an input and output structure variable.
    State

13. True or False: Different state variables should be used for different instances of a given processing block?
    false

14. Why do we have to compensate for the delay of the bandpass filter used to isolate the 8 kHz tone?
    Because we will be off by a constant phase after the down conversion of the input if we do not compensate.

15. When testing the DDC, why do we initially have to allow several blocks to process before getting correct output?
    Because the PLL takes some time to synchronize properly.

2) A print out of your finnal MATLAB implementation of the DDC.
We were asked to make changes in the file ddc.m, pll.m

ddc.m

```
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
[pll_out accum_out state.pll_state] = pll(t1, state.pll_state);
% Compensate for delay of BPF to get aligned carrier.
accum_out = accum_out + state.del;

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

```
% Do down-conversion (multiply by carrier)
x_q = x .* (-1).* sin1;
x_i = x .* cos1;
% Matched filtering
 [ q state.lpf1_state]  = fir ( x_q, state.lpf1_state);
 [ i state.lpf2_state]  = fir ( x_i, state.lpf2_state);

% Now you should output Ib and Qb which are the baseband I and Q samples
 Qb = q;
 Ib = i;
% Return debug information if desired
if (nargout >= 4),
  d.accum = accum1;
  d.cos1 = cos1;
  d.sin1 = sin1;
  d.ref = x;
  d.t1 = t1;
end

state_out = state;
```

pll.m

```
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;

%% Put your PLL code here !!!
% Phase compare
y(1) = state.ref_in_prev/amp_est*state.ref_out_prev;

% Loop filter
z(1) = a(1)*state.z_prev + b(1)*y(1) + b(2)*state.y_prev;

% VCO
accum = accum + f0 - K*z(1)/(2*pi);
accum = accum - floor(accum);

accum_out(1) = accum;
ref_out(1) = sin(2*pi*accum);

for n=2:N,
  % Multiply (phase compare)
  y((n)) = ref_in(n-1).*ref_out(n-1)/amp_est;

  % Loop filter
  z((n)) = a(1)*z(n-1) + b(1)*y((n)) + b(2)*y(n-1);

  % VCO
  accum = accum + f0 - K*z(n)/(2*pi);
  accum = accum - floor(accum);

  accum_out((n)) = accum;
  ref_out((n)) = sin(2*pi*accum);
end

% Store state for next call
state.ref_in_prev = ref_in((n));
state.ref_out_prev = ref_out((n));
state.y_prev = y((n));
state.z_prev = z((n));
state.accum = accum;
% Don't forget to use state variables properly!

state_out = state;
```

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.

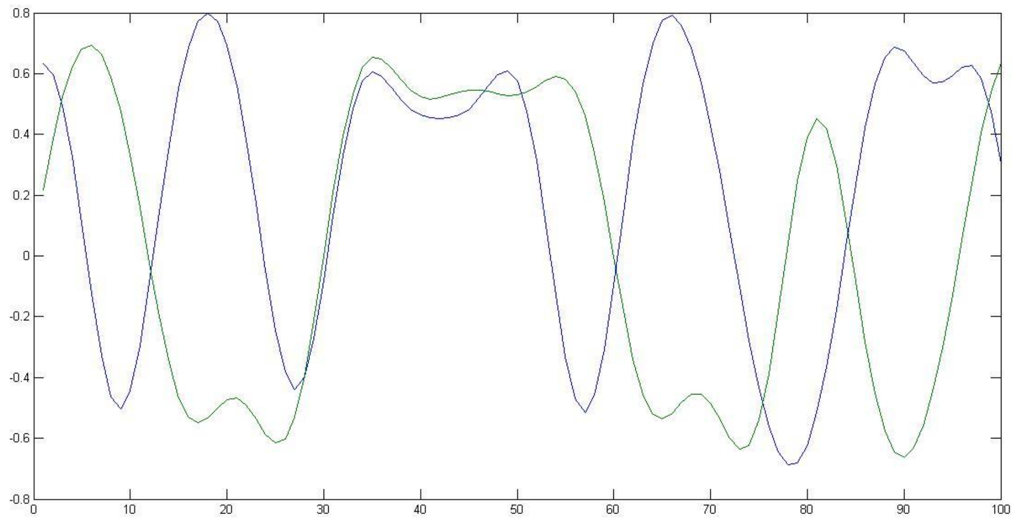

fig: plot showing the simulated performance of ddc.

The script generated a number of frames with the repeated symbol pattern
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];
From the plot it is easier to see that the first four sequence of 0000 because both I and Q are high and they lie in the first quadrant which corresponds to a zero. The fifth one the imaginary part is high and the real part is low which corresponds to “01” and is equivalent to 1. We can explain in the similar fashion for rest of the symbol pattern.

4) An explanation of any problems you ran into implementing the DDC and how you found and fixed them.
The first plot that we got after running the test_ddc.m file was not what we expected. Upon asking one of the TAs we came to know PLL requires some time to synchronize properly. Later we got the exact plot that we were expecting. And we had trouble in the script system_param.m because we were lacking the script win_method.m. It got us thinking for a while. Later we asked one of the TAs and the problem was resolved.