QPSK with carrier recovery

Bernd Porr*

1 Introduction

In this lab we are going to do the first step towards realistic processing in the receiver namely that we need to recover the carrier with the help of a PLL. Also in a realistic scenario we don’t know when the data stream is actually starting so that we need to introduce a start symbol which actually kicks off the sampling of the symbols.

2 QPSK rewrite

Rewrite the QPSK program in a way that it processes data truly sample by samples in the receiver. Have a look on moodle. There is an example which is also the solution of last week's lab. Use this code or modify your code. Here are some important changes:

- We have one large loop which retrieves the signal sample by sample and is then processed:

```matlab
for step = 1:ttotal
    v = psk_signal(step);
    [do some processing]
    [...]  
end
```

This has an impact on the filtering operation because the filter needs to remember which values have been fed in it previously. This is stored in a vector which needs to be initialised before the main loop starts:

```matlab
b = fir1(100,0.1);
stateLowpassInph = zeros(numel(b)-1,1);
stateLowpassQuad = zeros(numel(b)-1,1);
```

These are state memories for the filters (which are the values of its delays). When you run the filter operation the state is stored in these arrays:

```matlab
[lpinph(step),stateLPi] = filter(b,1,mixer_i(step),stateLPi);
[lpquad(step),stateLPq] = filter(b,1,mixer_q(step),stateLPq);
```

*© 2012 Bernd Porr under the Creative Commons license (attribution,non commerical,derivatives allowed) – mail@berndporr.me.uk – http://www.berndporr.me.uk
• Set up a counter which is used to sample the data at symbol intervals. The initial counter can be set to a different value that the first data sample is delayed. The standard approach is to have the counter count do zero, then a symbol is sampled and then the counter is reset to the symbol duration.

3 PLL

Write a PLL which can lock on the carrier signal:

• The first building block is a quadrature oscillator which you can download from moodle. At the core we have two interconnected difference equations which create an oscillation at frequency \( w_0 \):

\[
\text{for } i=1:1000 \\
\%"voltage" controlled oscillator with f=0.1 +/- v input \\
w_0 = 2\pi*(0.1+v*0.005); \\
c\_delay = c; \\
s\_delay = s; \\
c = c\_delay * \cos(w_0) - s\_delay * \sin(w_0); \\
s = s\_delay * \cos(w_0) + c\_delay * \sin(w_0); \\
sine = [sine s]; \\
cosine = [cosine c]; \\
\text{end}
\]

You get a signal in phase and one 90° degrees out of phase. \( v \) here is the voltage which can change the frequency. Run the oscillator and test that you can change the frequency.

• Now generate a test signal which later will become your carrier:

\[
\% and we have some random phase angle. \\
y = \sin(2\pi*0.099*t+\pi/2.345543); \\
\% and compare its phase to that of the VCO:
\]

\[
\% phase detector \\
p = y*c; \\
\% filtering out the 2f term \\
[v,zf] = filter(b,1,p,zf); 
\]

where \( v \) then feeds back to the VCO and hopefully changes the frequency so that our test signal \( y \) and the VCO output are in phase after a while.

• Check that the PLL locks on the test signal by plotting the VCO output and the test signal. They should be in phase. Check the VCO signal and see how long it takes until it stabilises. Experiment with the VCO gain and also the cutoff frequency of the PLL lowpass.
4 QPSK with PLL lock

- Modify the transmitter that it transmits 10 symbols at zero phase so that the PLL can lock on it. Remember zero phase is not used in the real transmission.

- Add the PLL to the QPSK program and replace the fixed sine/cosine terms with the PLL’s output. Check that the PLL locks on the dummy symbols at the beginning.

- Change the code so that symbol reception only starts after these 20 dummy symbols and that you see the pacman again.

- Add a start symbol to the transmitter, for example 11.

- Rewrite the code in a way that the length of the dummy sequence for PLL locking needs not to be known. Add a threshold function in the receiver which switches from PLL sync to data reception as soon as the 11 symbol is received (i.e. phase change from zero degrees to non-zero).

- Test the receiver by changing the dummy length to different values.

At the end you should have a receiver which is able to lock onto any phase and slightly frequency shifted signals which is important when you try out your receiver on signals which have travelled through a real physical medium.