QPSK transmission via a real communication channel

Bernd Porr*

1 Introduction

In this lab we are going to send your QPSK data through an auditory communciation channel consisting of a loudspeaker, a plastic box and a microphone.

2 Preparing the transmission

- Increase the symbol length to 50ms or 20ms (otherwise the ISI is too strong and decoding won't be possible without channel equalisation.)
- save your QPSK signal as a wav file with the command

```
wavwrite (QPSK_signal,'QPSK_signal_tx.wav');.
```

3 Transmission and recording

We are using two computers for the transmission and recording and use two simple command line tools for wave playback and recording.

- Download your QPSK wav file onto the transmission computer.
- Type on the receiving computer

```
arecord -f S16_LE QPSK_signal_rx.wav.
```

- On the transmitting computer type shortly after:

```
aplay QPSK_signal50ms.wav.
```

  or use VLC.

- Load the wav file into octave and plot it to check that it has reasonable quality.

Note we have only one setup and you can work in the meantime with an example wav file from moodle.

4 MATLAB program for reception

Use the code from last week where you transmitted from MATLAB to MATLAB and modify it. There is one major difference to your matlab-matlab transmission: the beginning of the transmission is very hard to detect because the signal has a gradual onset and there is no clear point where it starts. In order to overcome this problem we can use the VCO of the PLL to detect the beginning. The PLL will lock onto the signal as soon as it is strong enough. So we just need to detect if the PLL has locked on the carrier and then we can wait for the start symbol. So, essentially you need to write a lock detector. Alternatively you could look at the signal amplitude and apply a threshold on this. However other noise might trigger the reception.

The PLL lock is detected by adding another phase detector to the PLL which uses the quadrature signal from the oscillator. Remember the VCO signal settles if the multiplier output is zero because VCO signal and carrier are 90 degrees out of phase in the lock condition. However, if we use a 90 degree out of phase signal from the VCO then we get maximum output in the lock condition. That can be used with a threshold to decide that we can now wait for the start symbol `11`.

Here are the most important changes you need to make:

- Load the received wav file into MATLAB/OCTAVE with `wavread`.
- Normalise the received file:

  ```
  QPSK_signal = QPSK_signal / max(QPSK_signal);
  ```

- PLL dock detection: multiply the carrier with the quadrature output from the VCO and lowpass-filter it:

  ```
      % detect the quatrature which is basically a lock detect
      q = sample*s;
      [ld,zfLock] = filter(bPLL,1,q,zfLock);
  ```

- Check that the output `ld` from the lowpass filter goes high when the PLL has locked on the carrier (see Fig. 1).
- Add a counter which counts down from 1000 (or any other value which works) as soon as there is a lock detect and switches to data reception once it's zero:

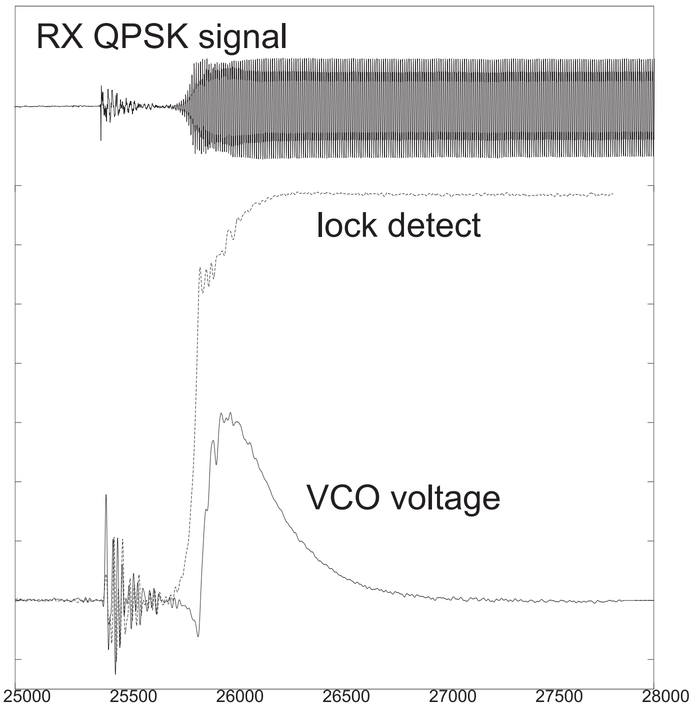

Figure 1: VCO voltage and lock detector output at the moment the PLL locks on the carrier.

```
if (ld > 0.1)
  lockCounter = lockCounter - 1;
else
  % spurious peak, no real lock, reset to max settling time
  lockCounter = PLLsettingTime;
end
```

The countdown allows the PLL to completely settle before it's switched into freeze. See Fig. 1 where the lock is detected at sample number 26000 but the PLL still needs to settle.

- Switch to data reception and PLL freeze as soon as the lock counter is zero.
- Check that your pacman is showing up again. If not tune your parameters.
- Plot the constellation diagram.