# LABORATORY PART

In the laboratory part we transmit a QPSK/RRC signal through a radio channel simulator and sample the simulator output using AD cards. The sampled data is transferred to MATLAB. You should design and implement a receiver that estimates the channel impulse response, removes intersymbol interference (ISI) from the signal, and demodulates the data.

## Step 1: Create the transmitted signal and download it to AWG

Assistant will assign you the message bits to be transmitted over the ISI channel. Write down the number of bits, since you will need it later. Map the bits to 4-ary symbols by using the function `bits2sym`. Add the training symbols to the beginning of the data block (Figure 1) and create the transmitted signal in MATLAB by using QPSK modulation and RRC signaling. Use eight samples per symbol. Save the real and imaginary part of the transmitted signal vector to separately under different names onto a floppy disk in ASCII form. Export both signals to AWG. Time interval between sampling points should be set to 1 $\mu s$. "Preserve time" when exporting.

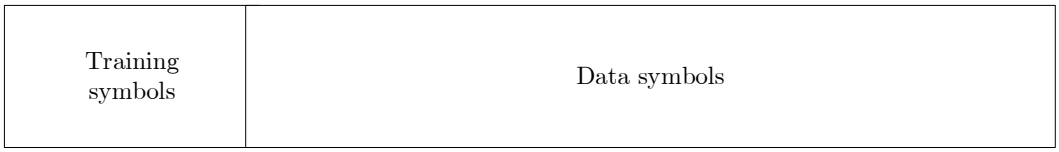


*Figure 1. The data format.*

You can follow the instructions in the file Lab2_L1.m

## Step 2: Acquiring the received signal to MATLAB

Connect the measurement setup.

Load data to the arbitrary waveform generator (AWG). For that you can follow the instructions given in the file Lab2_L2.doc.

The measurement set-up is depicted in Figure 2. In AWG, select the real part of the signal to be transmitted from channel A, and the imaginary part from channel B. Set the triggering to external with one burst transmitted on each trigger instant. Set the triggering signal frequency to 10 Hz or lower.

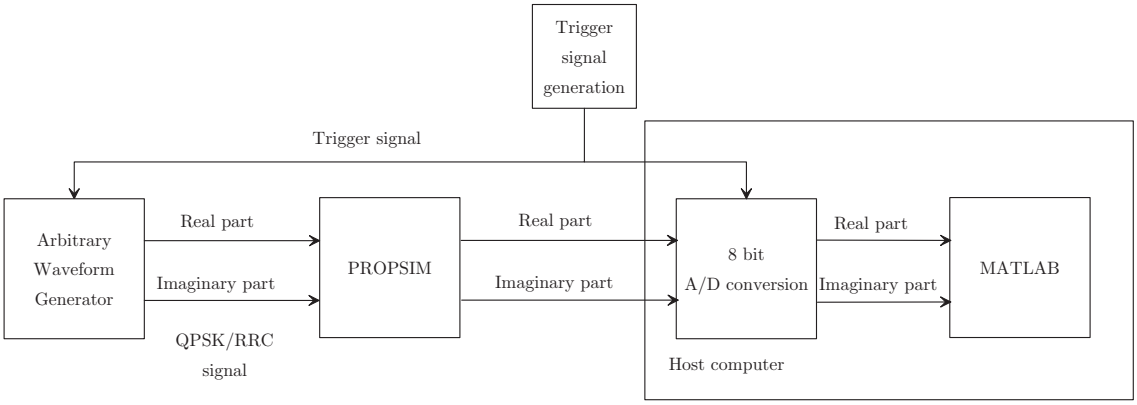


*Figure 2. Measurement set-up.*

Sampling cards are controlled with the script `lab2.m`. Running the script captures the complex channel output into two vectors that you should save for further processing.

Choose in the simulator channel models given below; What is the absolute delay spread of the channels? In theory, how many taps do you need in your channel estimate?

Once you have obtained the received signal, you may proceed to the post-laboratory part. If you have time you can start implementing the receiver in the student laboratory, and ask questions from the assistant.

Run the measurements with the following channels:

Lab2-1tap-const

Lab2-2tap-const

Lab2-3tap-const

Lab2-5tap-const

Lab2-3tap-const-rndph

Lab2-4tapconstfract

For the following channels make the measurements for two different user speed 3 and 10 km/h.

Lab2-2taprayl

Lab2-3taprayl

Lab2-5taprayl

## Step 3: Implement the receiver and detect the transmitted symbols

Implement the receiver shown in Figure 3 and detect the transmitted symbols. You may expand the script `t_ser.m` you implemented in the preliminary problems.

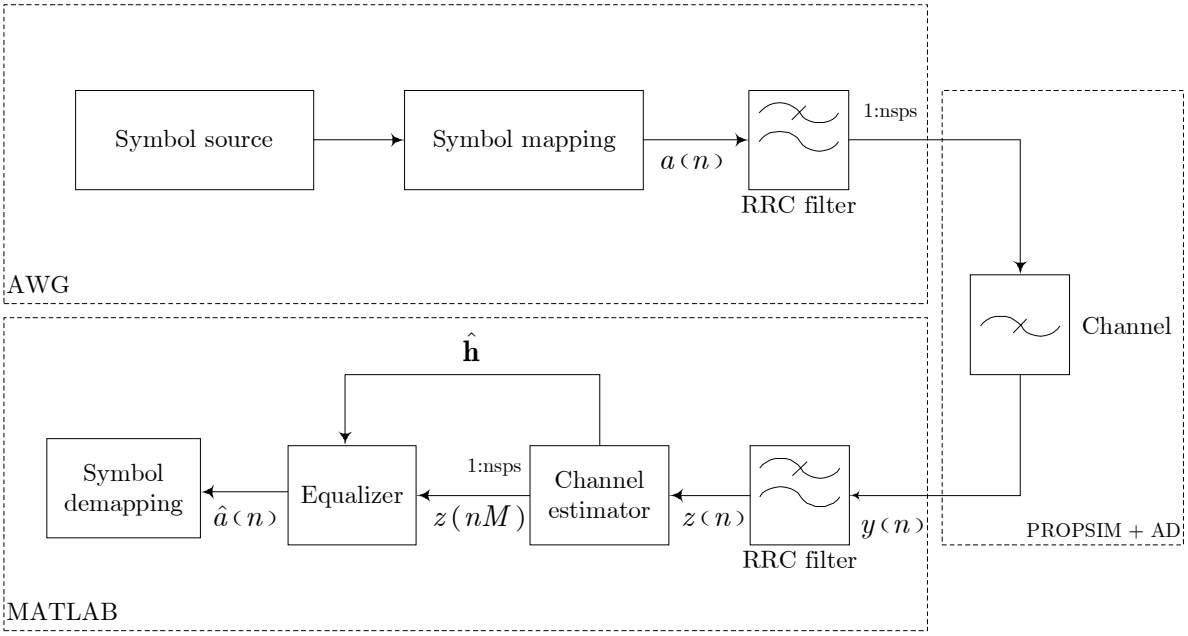


*Figure 3. Block diagram of the communication chain.*

Notes:

- RRC filter: Nothing strange here.
- Channel estimator: This block has two inter-related purposes: it has to supply a channel estimate $\hat{\mathbf{h}}$ to the equalizer, and it has to reduce the sampling rate of $z(n)$ to a single sample per symbol, since the equalizer works on that sampling rate. A heuristic procedure for accomplishing this is suggested below. You may figure out a better way of doing this.

Assume that nsps $= M$. Demultiplex the signal $z(n)$ to $M$ sequences with nsps=1 in each (sampling at symbol rate in each). You can now use the estimator function you implemented in the preliminary part to each of the $M$ subsequences separately. You will end up with $M$ estimates $\hat{\mathbf{h}}_i, i = 1,2,\ldots,M$.

- Symbol demapping: You can map the symbols back to bits with `bits2ascPrint.m`. At some point in the receiver chain you will have to remove the training sequence from the signal. You may do this after or before symbol demapping.

## Questions:

Analyze results

Make the following analysis for each measured channel for the first sampling sequence $z(n), n = 1$.

1. Estimate a channel response for all measured channels. Set the channel estimator length to 5.

2. Calculate the ISI for all the measured channels for the channel estimator length 5 and equalizer length 7.

3. Decode all the data sequence for all the channels. The sequence lab2_1tap_const will contain no errors. You can use this sequence as a correct reference sequence. Make the table describing the errors in different channels. Calculate the error for the binary bits. (This is the received sequence before converting it to ASCII text.)

4. Convert the detected data bits to ASCII text by using the script `bits2ascPrint.m`. Include the text in the appendix of your report.

5. Comment how different channels impact the equalization.

   - Impact of the channel taps.

   - Impact of the fading as a function of the user speed.

Make the following analyze only for the measurements with the channels: lab2_4tap_constfract, lab2_5tap_const.

In the following two questions use only the first sampled sequence $z(n), n = 1$.

6. Plot the system impulse response, i.e. the convolution of the channel response and the equalizer response. Repeat for a few different equalizer lengths and two different channel estimate lengths. (For example, estimator lengths 5,7,9 and equalizer lengths 5 9 21).

7. Compute the ISI. What is the optimal combination of channel estimate length and equalizer length? (Select all combinations of following parameters. Estimator length: 3, 5, 7; equalizer length: 3, 7, 9, 21)

Make the following analysis for each sampling instance $z(n), n = 1 \ldots M$.

8. Compute the ISI energy for each sampling instant using the optimum length equalizer and channel estimate.

9. Plot the frequency response of the channel, equalizer and the system for the optimum equalizer length, channel estimate length and sampling instant.

## Instructions for the lab report

Do not forget to include your email addresses to your lab report (for sending boomerang info). You should write the laboratory report with a word processor. There are no strict requirements on the format of the report, except that it must be readable and understandable. Anyone reading your report should be able to follow your line of reasoning and, if necessary, be able to repeat your actions and verify the results. To make the report more readable you should embed figures, tables etc to your report (not everything to appendices). Long MATLAB scripts and functions should be put into appendices.

For plotting figures, excellent black&white printing quality is obtained by using the command `print picname -deps2 -tiff`. The EPS picture file can be inserted into the document and resized according to one's needs without degradation in quality. Use `subplot` when applicable.

The report does not have to be long but you should include at least:

- A summary of your receiver implementation, including a short description of the MATLAB files.
- Answers to the questions presented above along with the appropriate figures.