Flexible GFDM PHY Tutorial

December 22, 2017

Author Ana B. Martinez
ana-belen.martinez@ifn.et.tu-dresden.de

Contributors
Martin Danneberg
Shahab Ehsanfar
Zhitao Lin
Maximilian Matthe
Ahmad Nimr
Dan Zhang

Table of Contents

1 Introduction

This document provides basic information about how to get started with the *Flexible-GFDM-PHY* project. The user is expected to have previous knowlegde about the LabVIEW environment and programming.

2 System Requirements

The *Flexible-GFDM-PHY* project has been implemented on a National Instruments (NI) Software Defined Radio (SDR) platform. For the development, NI USRP-RIOs 2953R with 40 MHz bandwidth and LabVIEW Communications Design Suite 2.0 have been used [1] [2]. The connection between the USRP-RIO devices and the control desktop computers has been done via an NI PCIe-8371 card. The project has been tested via cable as well as over the air with antennas VERT2450 [3].
Ensure that you are in compliance with all local laws before running this project with antennas to avoid a possible violation of local laws.
Compatibility with software and hardware versions different from the ones named in this document cannot be guaranteed.

3 URSP-RIO Setup

The project can run on one single USRP-RIO 2953R with 40 MHz or 120 MHz bandwidth or on two different USRP-RIO devices. In both cases the transmission can be realized either via a cable or over the air with antennas.

3.1 Setup with one USRP-RIO

The communication always takes place between one TX port and one RX port of the USRP-RIO. In this setup, we will choose transmitter and receiver on different RF channels of one USRP-RIO. Therefore, the two possible configurations are:

1. from RF0/TX1 to RF1/RX2

2. from RF1/TX1 to RF0/RX2

For the communication via cable, an additional 30 dB attenuator has to be connected between one end of the cable and one of the ports. In case of transmission over the air, antennas have to be located at the corresponding ports.

3.2 Setup with two USRP-RIOs

In this setup, each USRP-RIO is controlled by a separate desktop computer. Multiple configurations are possible, considering that the communication can only occur between one TX port of one USRP-RIO and one RX port of the other USRP-RIO.

4 Overview

The compressed file *Flexibe-GFDM-PHY-Master.zip* with the LabVIEW implementation of the GFDM transceiver can be downloaded from the website: "https://github.com/ewine-project/Flexible-GFDM-PHY".
Once this file is copied and extracted to the hard disk drive, it can be opened with NI Labview Communications Design Suite 2.0.

4.1 Folder structure

Figure 1 shows the folder structure of the **GFDM PHY.lvproject**.

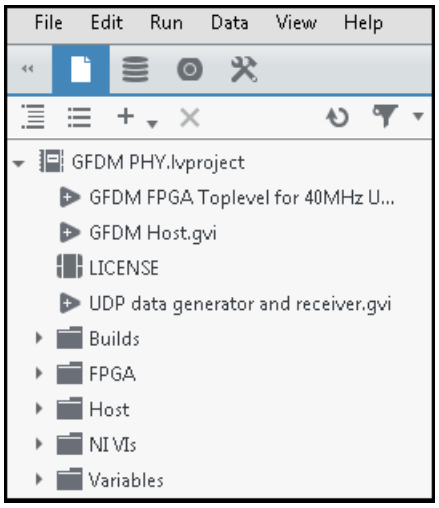


Figure 1: Folder structure of the **GFDM PHY.lvproject**.

The project contains different VIs:

1. **GFDM Host.gvi** - This host VI allows the configuration of the transceiver and interfaces with the bitfile **GFDM FPGA TopLevel for 40MHz USRP.lvbitx**.
2. **GFDM FPGA Toplevel for 40MHz USRP.gvi** - Top-level FPGA VI.
3. **UDP data generator and receiver.gvi** - Implements the communication via UDP.

and folders:

1. **Builds** - Contains precompiled bitfiles.
2. **FPGA** - Is organized into subfolders with FPGA VIs, addressing different functionalities of the transceiver.
3. **Host** - Comprises Host VIs and subfolders for different functions.
4. **NI VIs** - Contains VIs developed by NI.
5. **Variables** - Includes resources and variables.

5 Running Flexible GFDM PHY Project

1. Lunch LabVIEW Communications System Design Suite.
2. Open the project **GFDM PHY.lvproject**.
3. In the Files pane, open the host VI **GFDM Host.gvi**.
4. Configure the source data as well as the parameters for the proper operation of the transceiver. This part will be explained in the following sections.
5. Runk the LabVIEW host VI by clicking Run (the green button).

6 Configuration of parameters in GFDM Host.gvi

Figure 2 shows the transceiver user interface. The tabs on the right side (TX, RX and Data) are dedicated to the configuration of the parameters. Results corresponding to different modules of the transceiver are displayed on the tabs of the left side. Although not shown here, outside the user interface window, this VI displays important parameters with the configuration of the FPGA, which add information about the internal operation of the transceiver.

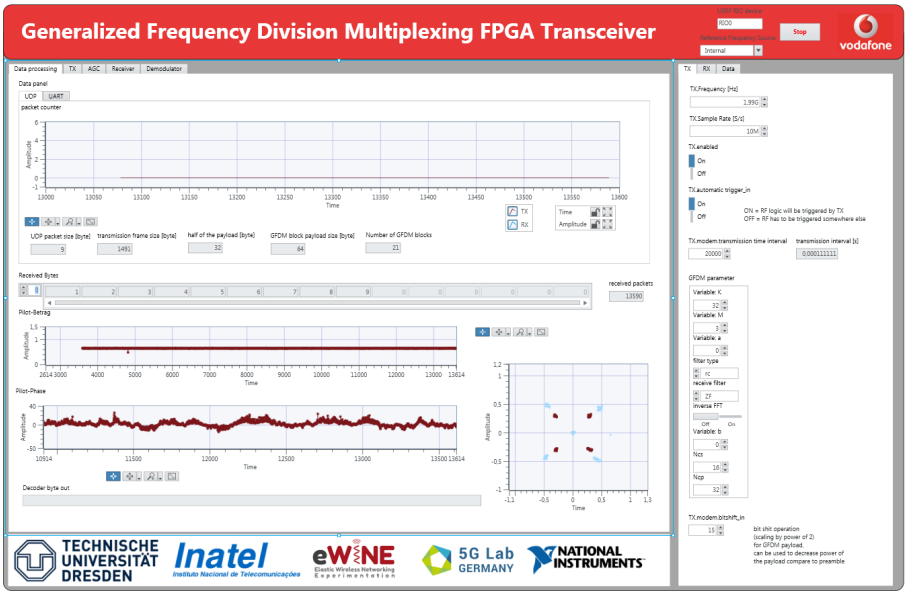


Figure 2: GFDM Host transceiver user interface. Data processing results (left) and parameter configuration (right).

The first step to take is the configuration of the **USRP RIO device** field in the upper right part of the screen. The NI Measurement & Automation Explorer (MAX) utility can be used to obtain the name of the USRP-RIO device.

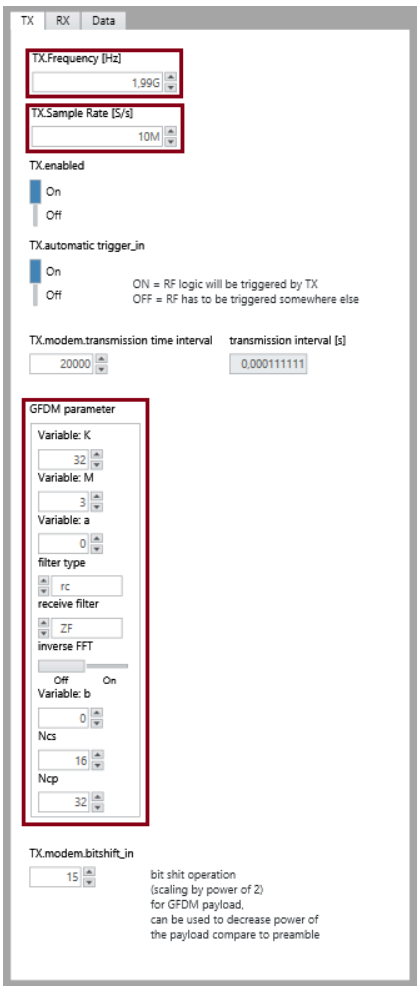
The USRP-RIO needs a reference frequency of 10 MHz to derive its clocks. The **Reference Frequency Source** has to be selected among:

1. **Internal** - Uses the internal reference clock.
2. **REF IN** - The reference is taken from the REF IN port.
3. **GPS** - The GPS module provides the clock reference.

The following subsections describe the parameters of the configuration tabs. Fields with white backgroud are configuration fields, while fields with grey background are indicators, i.e., they value cannot be changed. Some of the parameters of these tabs are used for internal operations and can be left to their default values. This document focuses on those parameters that need to be adapted to the specific application.

6.1 TX

The most important parameters to be configured on the TX tab are highlighted in Figure 3 and briefly explained on the right side.



- **TX.Frequency [Hz]** - Carrier frequency in Hz at the transmitter.
- **TX.Sample Rate [S/s]** - Sampling frequency in samples per second.
- **GFDM parameter**
  - **Variable: K** - Number of subcarriers.
  - **Variable: M** - Number of subsymbols.
  - **Variable: a** - Roll-off factor of the prototype filter (between 0 and 1).
  - **filter type** - The prototype filter can be chosen between raised cosine (rc) or rectangular in time domain (rect).
  - **receive filter** - Zero Forcing (ZF) or Matched Filter (MF).
  - **inverse FFT** - Set to default value Off for common operation (FFT is performed in modulator and inverse FFT in detector).
  - **Variable: b** - Roll-off factor of the time domain window.
  - **Ncs** - Length of cyclic suffix in samples.
  - **Ncp** - Length of cyclic prefix in samples.

Figure 3: TX configuration tab and description of highlighted parameters.

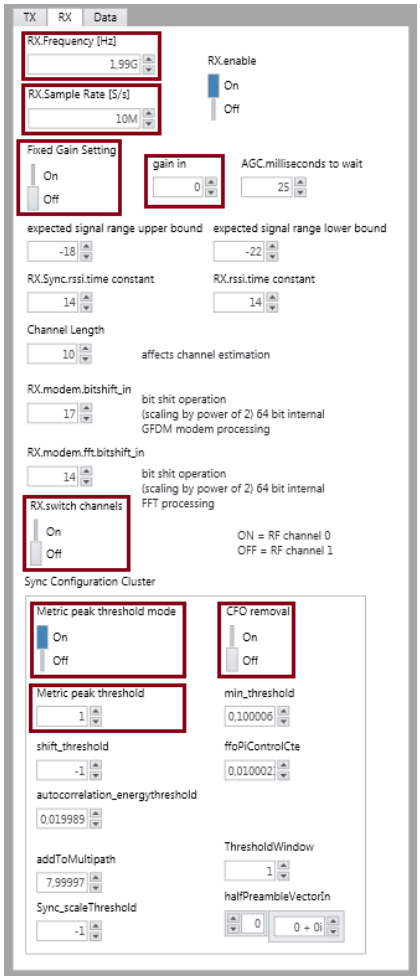
The following parameters can be left to their default values:

- **TX.enabled** - Enables/disables the transmitter. Default value On, transmitter enabled.
- **TX.automatic trigger_in** - If it is set to On, the transmitter triggers the RF logic. Otherwise another event has to activate it. Default value: On.
- **TX.modem.tranmission time interval** - Sets the number of cycles the transmitter has to wait until it can start generating a new packet. Default value: 20000.

- **tranmission time interval [s]** - Previous value expressed in seconds.
- **TX.modem.bitshift_in** - Shift (in bits) to scale the GFDM payload. This option can be useful, e.g., in case the preamble is sent with higher power than the payload. Default value: 15.

6.2 RX

Figure 4 displays the configuration parameters for the receiver.



- **RX.Frequency [Hz]** - Carrier frequency in Hz at the receiver.
- **RX.Sample Rate [S/s]** - Sampling frequency in samples per second.
- **Fixed Gain Setting** - The receiver can work with fixed gain (On) or adaptive (Off) one. Default value: Off.
- **gain in** - Fixed value of gain to work with in case **Fixed Gain Setting** is set to On. Default value: 0.
- **RX.switch channels** - Indicates which RF port is used for the reception. Select On to use the RX port on RF0 and Off for RF1. Default value: Off.
- **Metric peak threshold mode** - Activates adaptive (On) calculation of threshold or uses a fixed (Off) one to determine if synchronization is found. Default value: On.
- **Metric peak threshold** - Threshold to use for detection of synchronization peaks if the previous parameter is set to Off. Default value: 1.
- **CFO removal** - Indicates if CFO has to be compensated. Default value: On.

Figure 4: RX configuration tab and description of highlighted parameters.

A short description of the rest of parameters of the **RX** tab follows below:

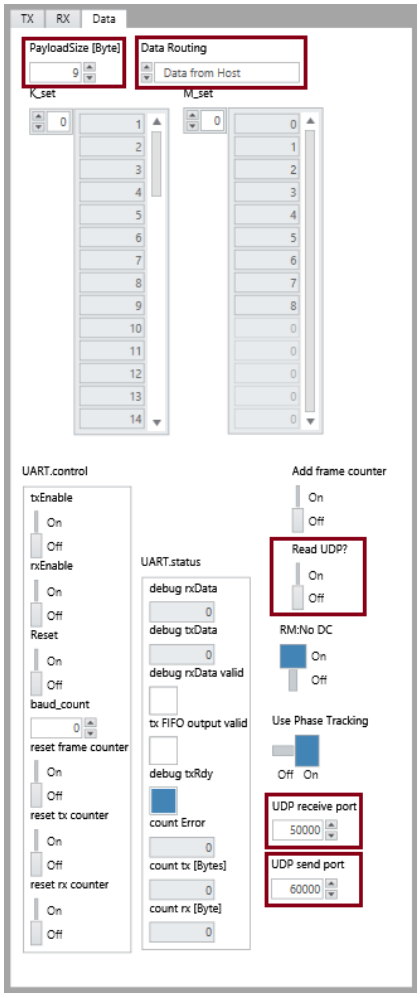
- **RX.enabled** - Enables/disables the receiver. Default value: On. In case the project runs on one single USRP-RIO this parameter and its counterpart on the **TX** tab must be enabled. Their values have to be adapted if different USRP-RIOs are used as transmitter and receiver.

- When **Fixed Gain Setting** is set to Off, different parameters are needed for internal calculations and suitable adjustment of the gain. These parameters, which can be left to their values, are listed below:
  - **AGC.milliseconds to wait** - Time to wait (in milliseconds) to adapt the gain.
  - **expected signal range upper bound** - Parameter needed to adjust the gain depending on the previous value of gain and the measured power. Default value: -18.
  - **expected signal range lower bound** - Parameter needed to adjust the gain depending on the previous value of gain and the measured power. Default value: -22.
  - **RX.Sync.rssi.time constant** - Time constant used during internal power calculation (after synchronization). Default value: 14.
  - **RX.rssi.time constant** - Time constant used during internal power calculation (before synchronization). Default value: 14.
- **Channel Length** - Allows to adjust the length of the estimated Channel Impulse Response (CIR), i.e. the number of coefficients that will be used to calculate the channel frequency response with $N = K \cdot M$ samples. Default value: 10.
- **RX.modem.bitshift_in** - This parameter indicates the number of bits used internally to make a conversion from 64 to 32 bits in the core modem. Default value: 17.
- **RX.modem.fft.bitshift_in** - As in the previous case, a proper scaling is necessary to convert a 64-bit signal to a 32-bit signal at the output of the fft block. Default value: 14.
- The most important parameters of the **Sync Configuration Cluster** are described next to Figure 4. The rest of parameters related to the synchronization are listed below:
  - **min_threshold** - Minimum threshold used to detect synchronization peaks. A low value higher than zero has to be chosen. Default value: 0,1.
  - **shift_threshold** - Scales the threshold calculated to detect synchronization peaks. Default value: -1.
  - **ffoPiControlCte** - Constant to control the speed of fractional frequency offset compensation. Default value: 0,01.
  - **autocorrelation_energythreshold** - Avoids division by zero when calculating the normalized autocorrelation.This parameter has to be a low value higher than zero. Default value: 0,02.
  - **addToMultipath** - Internal parameter. Default value: 8.
  - **ThresholdWindow** - Internal parameter. Default value: 1.

– **Sync_scale Threshold** - Internal parameter. Default value: -1.

– **halfPreambleVectorIn** - Periodic part of the preamble used for synchronization and channel estimation. The value of this parameter is currently read from a file.

6.3 Data



- **PayloadSize [Byte]** - Payload size in bytes. Default value: 9.
- **Data Routing** - Determines the source of input data among:
  - **Data from Host**
  - **Data from UART**
  - **UART Loopback**

  Default value: Data from Host. More details about this parameter are given below.
- **Read UDP?** - If it is set On, UDP data is used. Otherwise a ramp signal with **PayloadSize** length is generated. Default value: Off.
- **UDP receive port** - Number of UDP port used to receive data from the video stream player. Default value: 50000.
- **UDP send port** - Number of UDP port used to send data to the video stream player. Default value: 60000.

Figure 5: Data configuration tab and description of highlighted parameters.

The parameter **Data Routing** indicates where the source data used by the transmitter comes from.

- In case **Data from Host** is selected, the input data is sent from the host to the FPGA. As indicated before, depending on the parameter **Read UDP?**, the source data is generated as a ramp function (Off) or obtained via UDP (On). This latter case is specially convenient if an external video streaming application is used. The transmitter receives the source data

as UDP packets from the video streaming server and the receiver sends the output data via UDP to the player to be displayed. In order to allow UDP data communication, transmitter and receiver have to be configured properly.

- Configuration of the transmitter:
  - Start cmd.exe and change to the VLC installation directory.
  - Identify the path and name of the video file that will be used for streaming "video_file".
  - Start the VLC application as a streaming server with the following command:
    vlc "video_file"
    :sout=#std{access=udp{ttl=1},mux=ts,dst=127.0.0.1:"vlc_tx_port"}
    where "vlc_tx_port" corresponds to **UDP receive port** in the **Data** configuration tab.
- Configuration of the receiver:
  - From the command line window, start the VLC application as a streaming client with the command:
    vlc udp://@:"vlc_rx_port"
    where "vlc_rx_port" corresponds to **UDP send port** in the **Data** configuration tab.

The **GFDM PHY.lvproject** has been succesfully tested with VLC and the default **GFDM parameter** values, changing the roll-off factors of the prototype filter **a** and the time domain window **b**.

- The data can also be obtained through one of the pins of the AUX I/O port of the USRP-RIO if **Data from UART** is selected. Once the received data has been decoded, it will be sent to the host to be displayed as well as to one of the pins of the AUX I/O port, being in this way available for other purposes.

- In case **UART Loopback** is selected, the input data received from the AUX I/O is sent to the encoder, to the host to be displayed and to the corresponding output pin of the AUX I/O.
- In these two last cases where the UART communication is involved, the **UART.control** parameters have to be configured, which, by default, are deactivated.
- **baud_count** indicates the number of clock cycles between baud ticks used to set a known transmit/receive baud rate. The clock frequency (in Hz) divided by the expected baud rate gives the baud count. For a clock frequency of 40 MHz, this parameter should be set to 200 in order to obtain a baud rate of 200000.

Some additional parameters in this tab are:

- **K_set** - Displays the set of allocated subcarriers, which corresponds to $[0, 1, ..., K-1]$, where K is the number of subcarriers selected in the **TX** configuration tab.

- **M_set** - Displays the set of allocated subsymbols: $[0, 1, ..., M-1]$, where M is the number of subsymbols selected in the **TX** configuration tab.
- **Add frame counter** - Activates the frame counter which is shown in the **Data processing** tab. Default value: Off.
- **RM:No DC** - If set to On, leaves the subcarrier corresponding to DC unused. Default value: On.
- **Use Phase Tracking** - Enables Common Phase Error estimation and compensation if it is set to On. Default value: On.

7 GFDM FPGA TopLevel for 40MHz USRP

This section describes briefly the functionality of the Clock Driven Logic (CDL) loops of the top-level FPGA VI and adds references to relevant literature for further information.

7.1 Data Routing

Different FIFOs are configured in this CDL depending on the type of data chosen in the field **Data Routing**.

7.2 Encoder

Performs convolutional encoding and QPSK mapping.

7.3 Resource Mapper - Demapper

The resource mapper mapps data and control signals before the modulation, whereas the demapper demapps the data coming from the detector and applies QPSK demapping prior to the decoding.

7.4 Decoder

The decoder takes as input the already demapped data from the previous CDL and applies Viterbi decoding.

7.5 Detector

This CDL contains the channel equalizer and the core modem acting as demodulator.

7.6 Modulator

Performs Generalized Frequency Division Multiplexing (GFDM) modulation [4] using the core modem. Afterwards Cyclic Prefix (CP) and Cyclic Suffix (CS) insertion as well as windowing take place. Finally, a preamble, which is used for synchronization and channel estimation purposes, is prepended to the modulated data.

7.7 Synchronizer

Receives the data from the Analog Digital Converter (ADC) and performs synchronization according to the algorithm proposed in [5].

7.8 Register Bus and Required Controls/Indicators

In this CDL, registers needed for the appropriate operation of the transceiver are defined and initialized.

7.9 Streaming Transceiver Engine

Contains original code from NI with the definition of the streaming engine used as a baseline for the transceiver. This CDL receives from the modulator the signal to be transmitted and provides as output the received signal after the ADC operation.

References

[1] National Instruments. *NI USRP RIO. Data Sheet*. URL: http://http://www.ni.com/datasheet/pdf/en/ds-538.

[2] National Instruments. *Getting Started Guide. USRP 2950/2952/2953/2954/2955*. URL: http://www.ni.com/pdf/manuals/376355c.pdf.

[3] A National Instruments Company. Ettus Research. *VERT2450 Antenna*. URL: https://www.ettus.com/product/details/VERT2450.

[4] Martin Danneberg et al. "Flexible GFDM Implementation in FPGA with Support to Run-Time Reconfiguration". In: *VTC Fall*. IEEE, 2015.

[5] Ivan S. Gaspar et al. "A synchronization technique for generalized frequency division multiplexing". In: *EURASIP Journal on Advances in Signal Processing*. 2014.