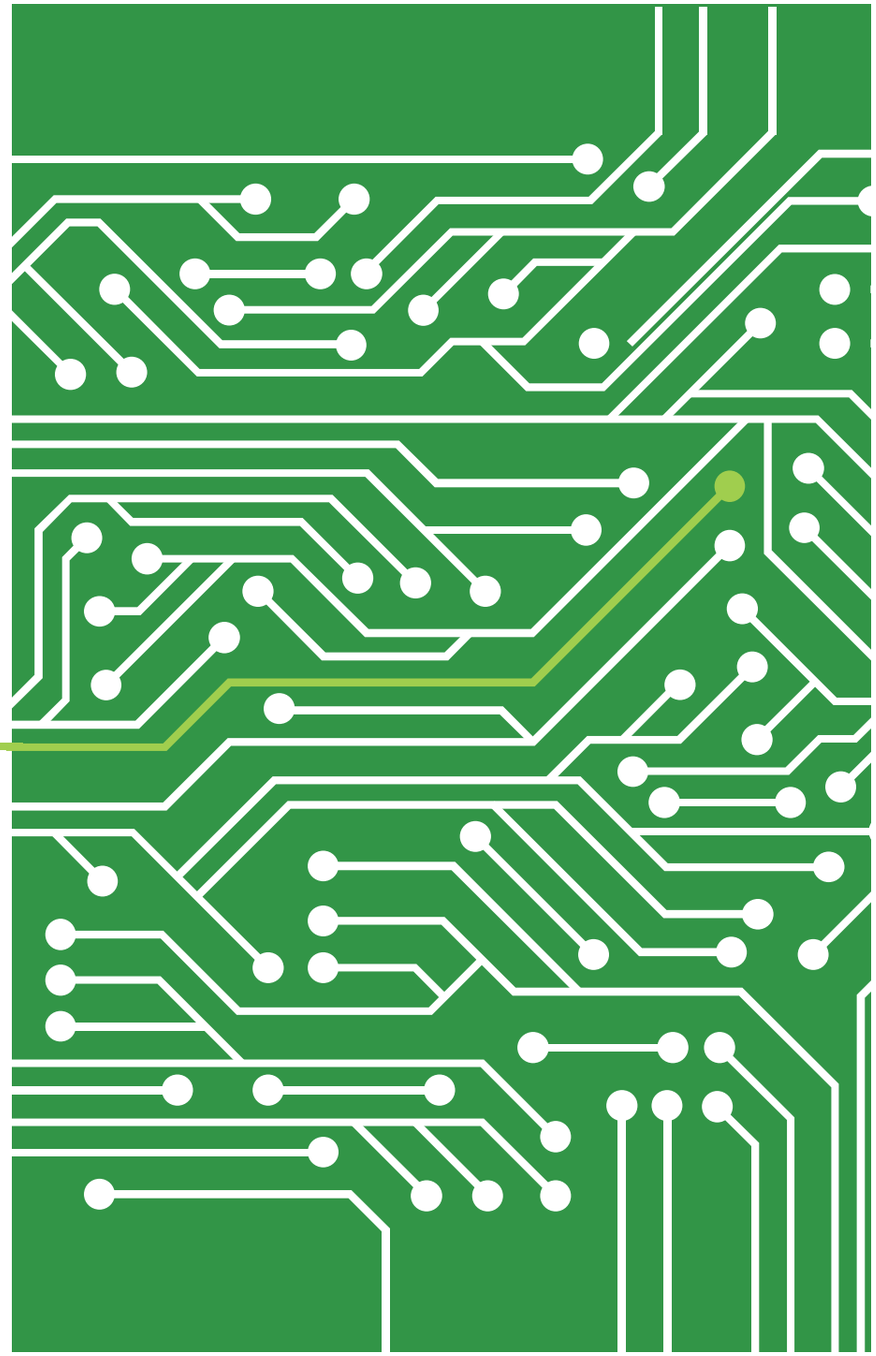
Lime microsystems

Vodafone CrowdCell Course:

LimeSuite APIs

Lime Microsystems| FPRF company
Guildford, Surrey, United Kingdom

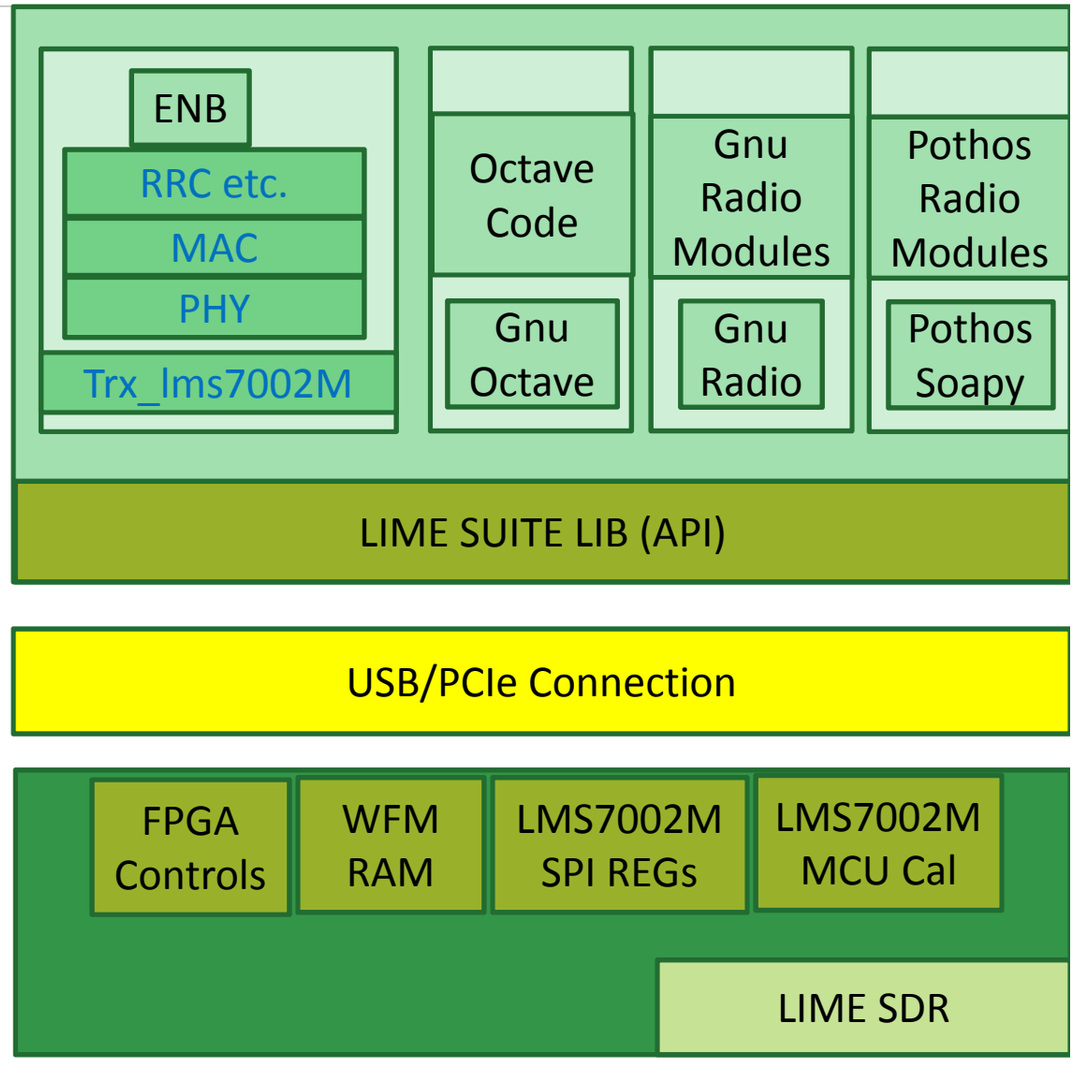


LimeSuite API

LimeSuite Library

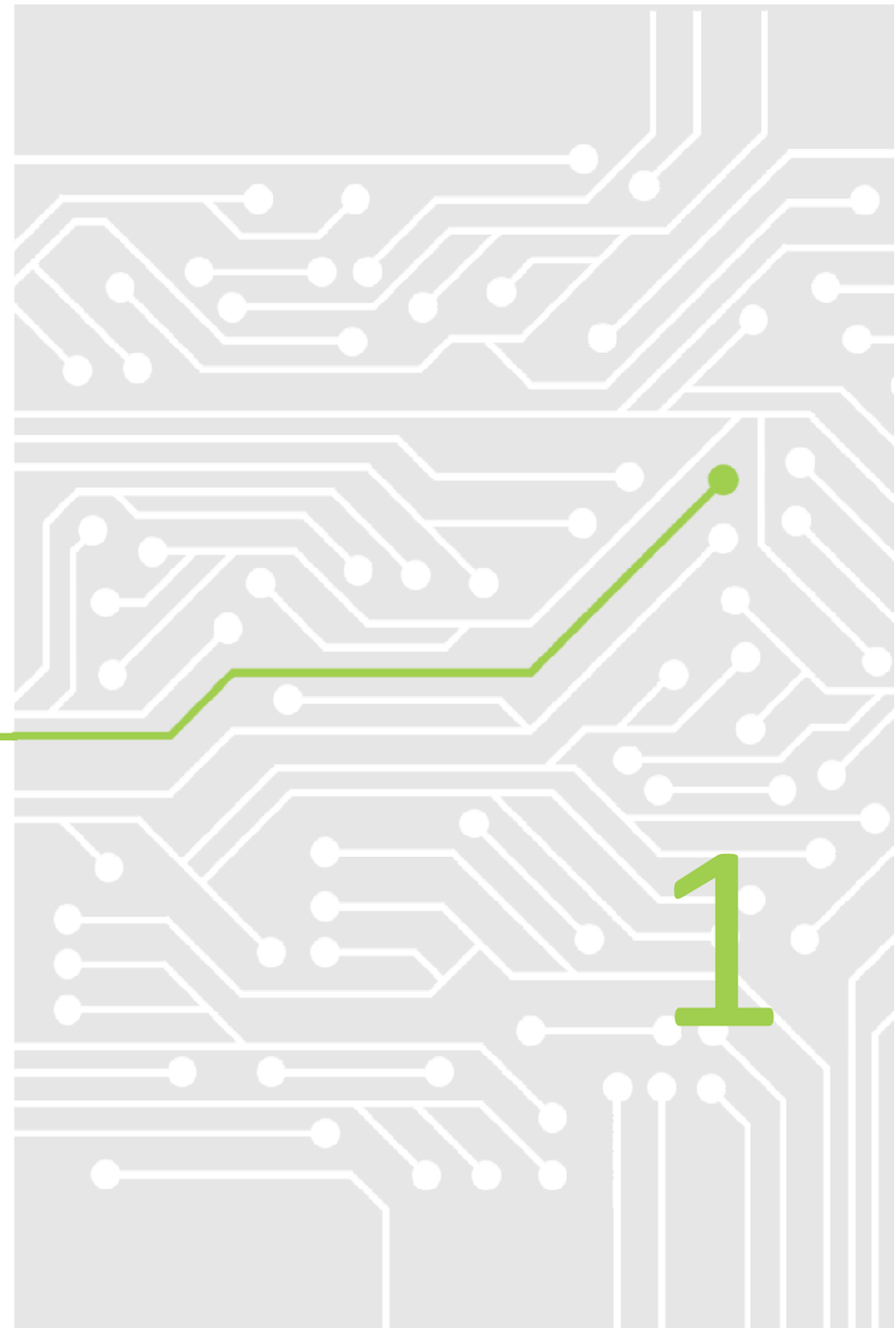
- C/C++ Application Interface
- Joins software defined radio (e.g. 4G ENB, GnuRadio etc) with the physical hardware of the LimeSDR

1. **LimeSuite APIs C/C++**
2. **Octave link to LimeSuite C/C++**
3. **GNU Radio link to LimeSuite**
4. **Pothos link to LimeSuite (via SoapyRF)**



Lime microsystems

C/C++



LimeSuite APIs – Header Files

Library build instructions

- https://wiki.myriadrf.org/Lime_Suite

Library's are stored

- /opt/local/lib

~/LimeSuite/src/lime/LimeSuite.h

contains

- header functions
- data types
- Documentation

~/LimeSuite/src/examples

- Contains simple examples

Typical minimial headers for LimeSuite

- #include <stdio.h>
- #include <stdlib.h>
- #include <string.h>
- #include <math.h>
- #include <complex.h>
- #include "lime/LimeSuite.h"

LimeSuite APIs – Connecting to an SDR

Read in a list of devices

- lms_device_t* device=NULL; // SDR device
- …
- int n; //Number of devices
- if((n=LMS_GetDeviceList(NULL))<0) error(); // Pass NULL to obtain devices
- lms_info_str_t* list = new lms_info_str_t[n]; // List of devices
- if(LMS_GetDeviceList(list)<0) error();

Select first device from list (usually only one LimeSDR present

- if(LMS_Open(&device,list[0],NULL)) error(); //Open the first device
- delete [] list;
- …

LimeSuite – Configuring the SDR

Can set up device in 5 ways

- Reset Device
- Use default settings
- Load a .ini file (from LimeSuite)
- Editing SPI registers
- Using helper functions

Reset (LMS7002M only)

- if(LMS_Reset(device)) error();

Default settings (LimeSDR and LMS7002M)

- LMS_Init(device);

Using .ini (e.g. from LimSuiteGUI)

- LMS_LoadConfig(device, “./config/file.ini");

LimeSuite – Register edit

// 0x0113,bits 1,0 = 3 => RFE:TIA Gain=2

// https://github.com/myriadrf/LMS7002M-docs/blob/master/LMS7002M_Programming_and_Calibration_Guide_v31r05.pdf

- LMS_WriteParam(lms_device_t *device, struct LMS7Parameter param, uint16_t val); struct LMS7Parameter
- {
- uint16_t address;
- uint8_t msb;
- uint8_t lsb;
- uint16_t defaultValue;
- const char* name;
- const char* tooltip;
- };

LimeSuite – Configuring with helper functions

Enable device

- Ch=0; // is Channel A in LimeSuite
- if(LMS_EnableChannel(device,LMS_CH_RX,Ch,true)!=0) error();
- if(LMS_EnableChannel(device,LMS_CH_TX,Ch,true)!=0) error();

Set LO frequencies

- double frq_cen=860e6; // Hz
- if(LMS_SetLOFrequency(device,LMS_CH_RX,Ch,frq_cen)!=0) error();
- if(LMS_SetLOFrequency(device,LMS_CH_TX,Ch,frq_cen)!=0) error();

Set Rx Antenna

- char LNAnum=3; // LNAW=3, LNAL=2, LNAH=1, 0=default
- lms_name_t antenna_list[10];
- if((n = LMS_GetAntennaList(device, LMS_CH_RX, 0, antenna_list)) < 0) error();
- if(LMS_SetAntenna(device, LMS_CH_RX, Ch, LMS_PATH_LNAH) != 0) error();

LimeSuite – Configuring with helper functions

Set sample rate

- unsigned char OSR=16; // oversample ratio DAC/ADC to USB/PCIe
- float_type rate=frq_Samp; // USB/PCIe samples/second
- double frq_Samp=1.0e6; // USB/PCIe Samples/second
- if(LMS_SetSampleRate(device,frq_Samp,OSR) != 0) error();

Set LPF frequency

- double frq_LPF=2.0e6;
- if(LMS_SetLPFBW(device, LMS_CH_RX,Ch,frq_LPF)!=0) error();

LimeSuite – Repeated Waveform Playback

LimeSDR-USB, LimeSDR-PCIe, LimeSDR-QPCIe only

Create Storage for waveform

- complex16_t **wfmBuffers = new complex16_t*[chCount];
- for(int i=0; i<chCount; ++i)
- wfmBuffers[i] = new complex16_t[samplesCount];

// load your data into wfmBuffers e.g. from wfm files

Load data into SDR RAM LMS_UploadWFM(device, (const void)wfmBuffers,ch, size_t sample_count, int format);**

Enable waveform playback

- LMS_EnableTxWFM(device,Ch,true);

LimeSuite – Starting the Tx stream

To send IQ samples to Tx, a stream has to be set up for each channel.

- lms_stream_t TXstreamId; // SDR stream
- lms_stream_meta_t tx_metadata;
- TXstreamId.channel=Ch; //channel number
- TXstreamId.fifoSize=4*pts; //fifo size in samples
- TXstreamId.throughputVsLatency=0.5; //optimize 0:1
- TXstreamId.isTx=true; //TX channel
- TXstreamId.dataFmt=lms_stream_t::LMS_FMT_I16;
- if(LMS_SetupStream(device,&TXstreamId)!=0) error();
- LMS_StartStream(&TXstreamId);

LimeSuite – Starting the Rx stream

To receive IQ samples from Rx, a stream has to be set up for each channel.

- lms_stream_t RXstreamId; // SDR stream
- lms_stream_meta_t rx_metadata;
- RXstreamId.channel=Ch; //channel number
- RXstreamId.fifoSize=4*pts; //fifo size in samples
- RXstreamId.throughputVsLatency=0.5; //optimize 0:1
- RXstreamId.isTx=false; //RX channel
- RXstreamId.dataFmt=lms_stream_t::LMS_FMT_I16;
- if(LMS_SetupStream(device,&RXstreamId)!=0) error();
- LMS_StartStream(&RXstreamId);

LimeSuite – Finishing up nicely

Disable Tx

- if(LMS_EnableChannel(device,LMS_CH_TX,Ch,false)!=0) error();
- if(LMS_SetGaindB(device,LMS_CH_TX,Ch,0)!= 0) error(); // switch off Tx

Close streams

- LMS_StopStream(&RXstreamId); // start again with LMS_StartStream()
- LMS_StopStream(&TXstreamId); // start again with LMS_StartStream()
- Streams so far are only paused,
- to release memory and permanently stop
- LMS_DestroyStream(device, &RXstreamId);//stream can no longer be used
- LMS_DestroyStream(device, &TXstreamId);//stream can no longer be used

Close device

- LMS_Close(device);

LimeSuite – Calibrating the SDR

Use LMS7002M MCU scripts to calibrate LMS7002M

Each MIMO channel is done separately

If stream is running, stop stream first, restart after calibration

- float_type rate=1.0e6; // USB/PCIe samples/second
- …
- if(LMS_Calibrate(device,LMS_CH_RX,Ch,rate,0)!=0) error();
- if(LMS_Calibrate(device,LMS_CH_TX,Ch,rate,0)!=0) error();

LimeSuite – Transmitting and receiving

To receive IQ samples from Rx

- samplesRead=LMS_RecvStream(&RXstreamId,RXbuffer,pts,&rx_metadata,1000);

To send IQ samples to Tx

Delay needed as Tx will be transmitted in the future!

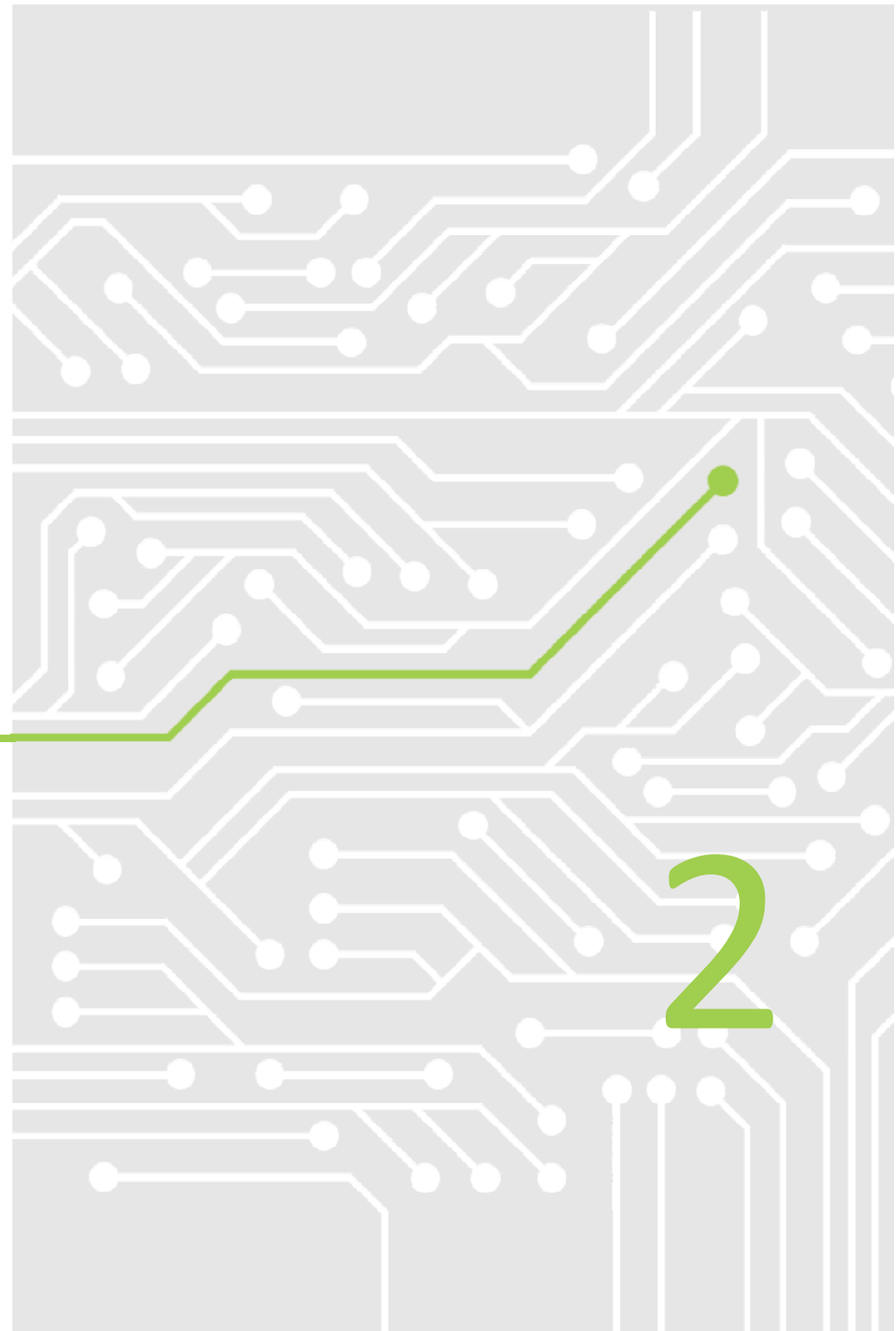
- int delay=1024*32; // delay between RX data start time and Tx data start time
- tx_metadata.timestamp=rx_metadata.timestamp+delay;
- LMS_SendStream(&TXstreamId,TXbuffer,samplesRead,&tx_metadata,1000);

LimeSuite – Detecting problems

Checking for lost packets in USB/PCIe connection

- float fifosizelog;
- uint32_t droplog;
- uint64_t timeStamp; // do not convert to long if R’Pi
- err=LMS_GetStreamStatus(&streamId,&status);
- timeStamp[cp]=status.timestamp;
- droplog=status.droppedPackets;
- fifosizelog=(100.0*status.fifoFilledCount)/status.fifoSize;
- bad=status.droppedPackets;

Octave



Installing libraries for Octave

Notes:

- Windows uses a DLL and a path must be set to this.

Linux

- Copy files to ~/octave

Use package installer

- octave
- cd ~/octave
- pkg install limesdr_1.0.2.tar.gz
- pkg install limesp_1.0.1.tar.gz
- pkg list

Octave

Load libraries

- pkg load limesdr
- pkg load limesp

Load LimeSuite library

- LoadLimeSuite
- LimeInitialize();

Read settings file

- LimeLoadConfig(‘./test.ini');

Generate SSB test signal

- phase = pi/8;
- Sig = 0.7*complex(sin(0:phase:1000*pi-phase), cos(0:phase:1000*pi-phase));

Set up streams

- LimeStartStreaming(length(sig), ['tx0'; 'rx0']);

For repeated waveform playback on LimeSDR-USB

- LimeLoopWFMStart(sig);

Else for LimeSDR-mini, LimeNet micro

- iqDataRx = LimeTransceiveSamples(sig, 3, 0);

Receive samples

- samples = LimeReceiveSamples(1360);

Pause Waveform Playback and Streaming

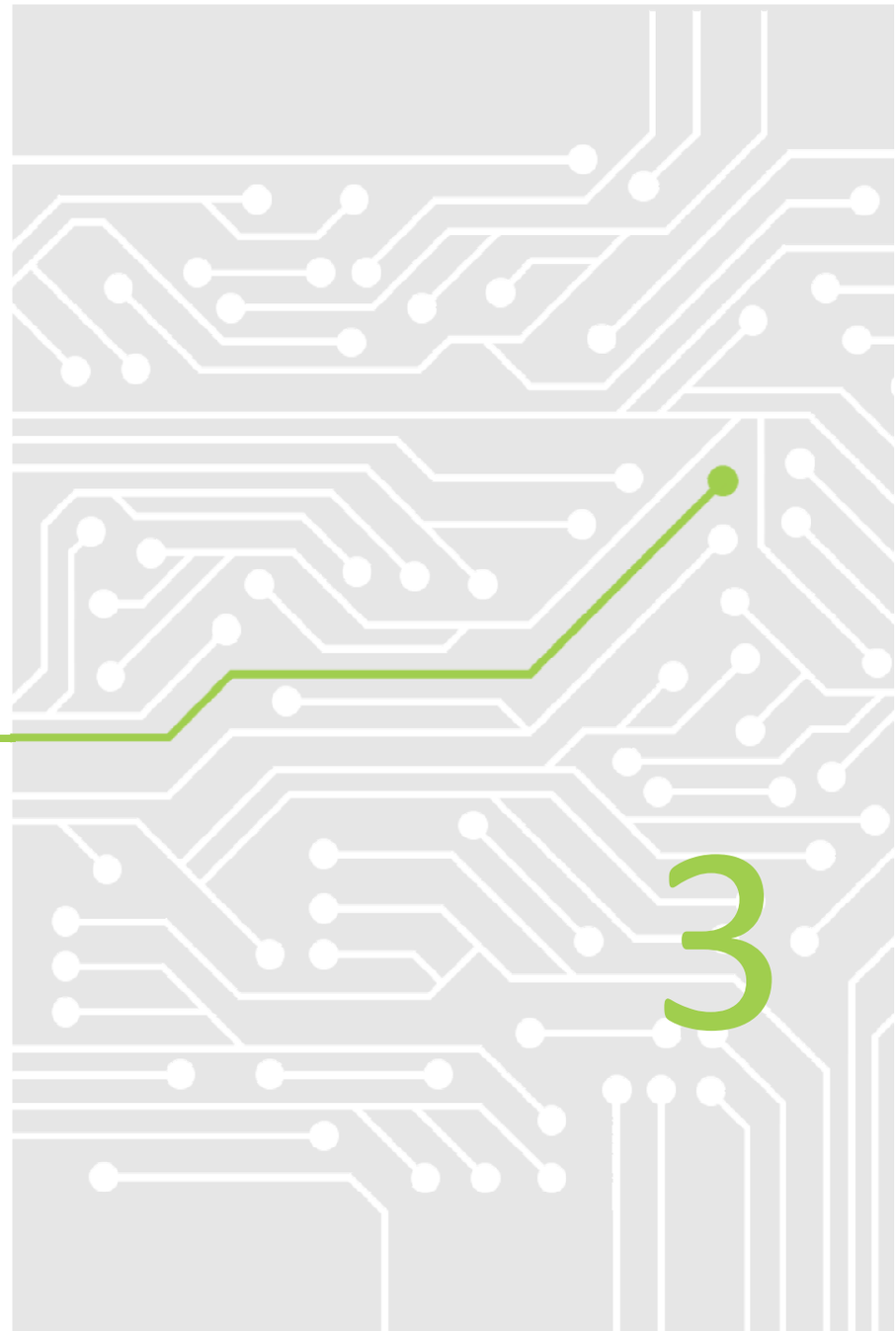
- LimeLoopWFMStop();
- LimeStopStreaming();

Close SDR

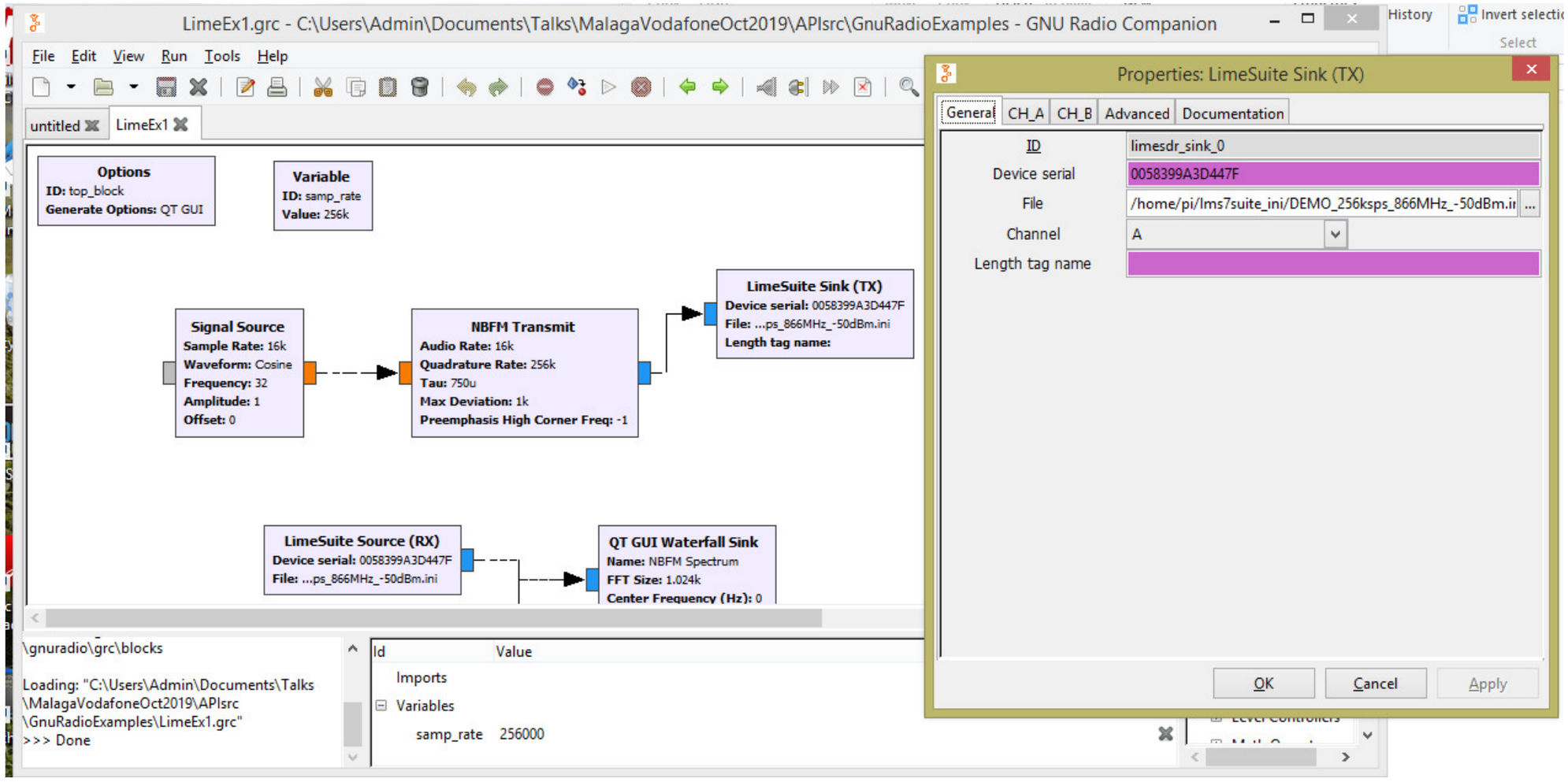
- LimeDestroy();

Lime microsystems

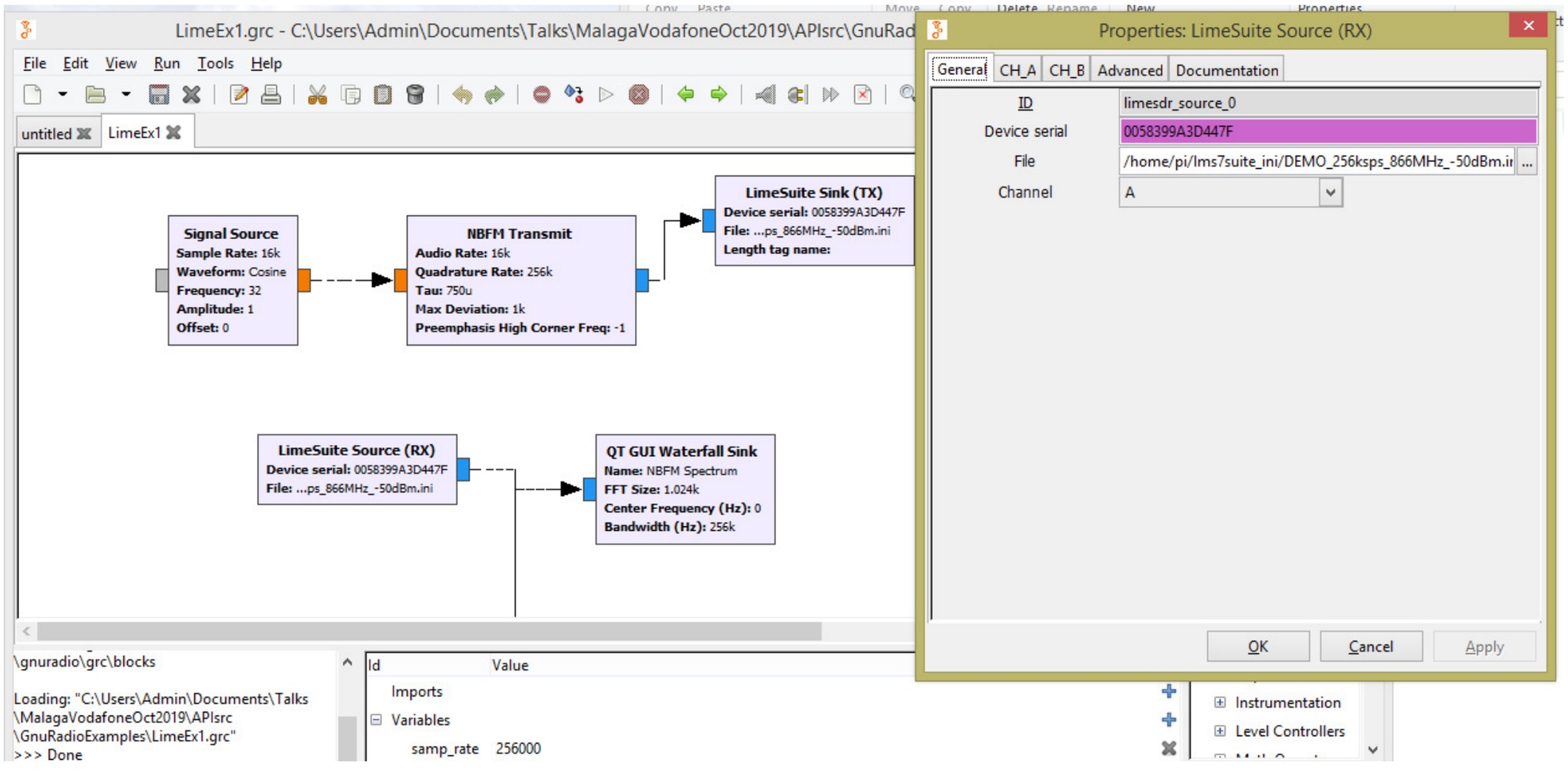
GNU Radio



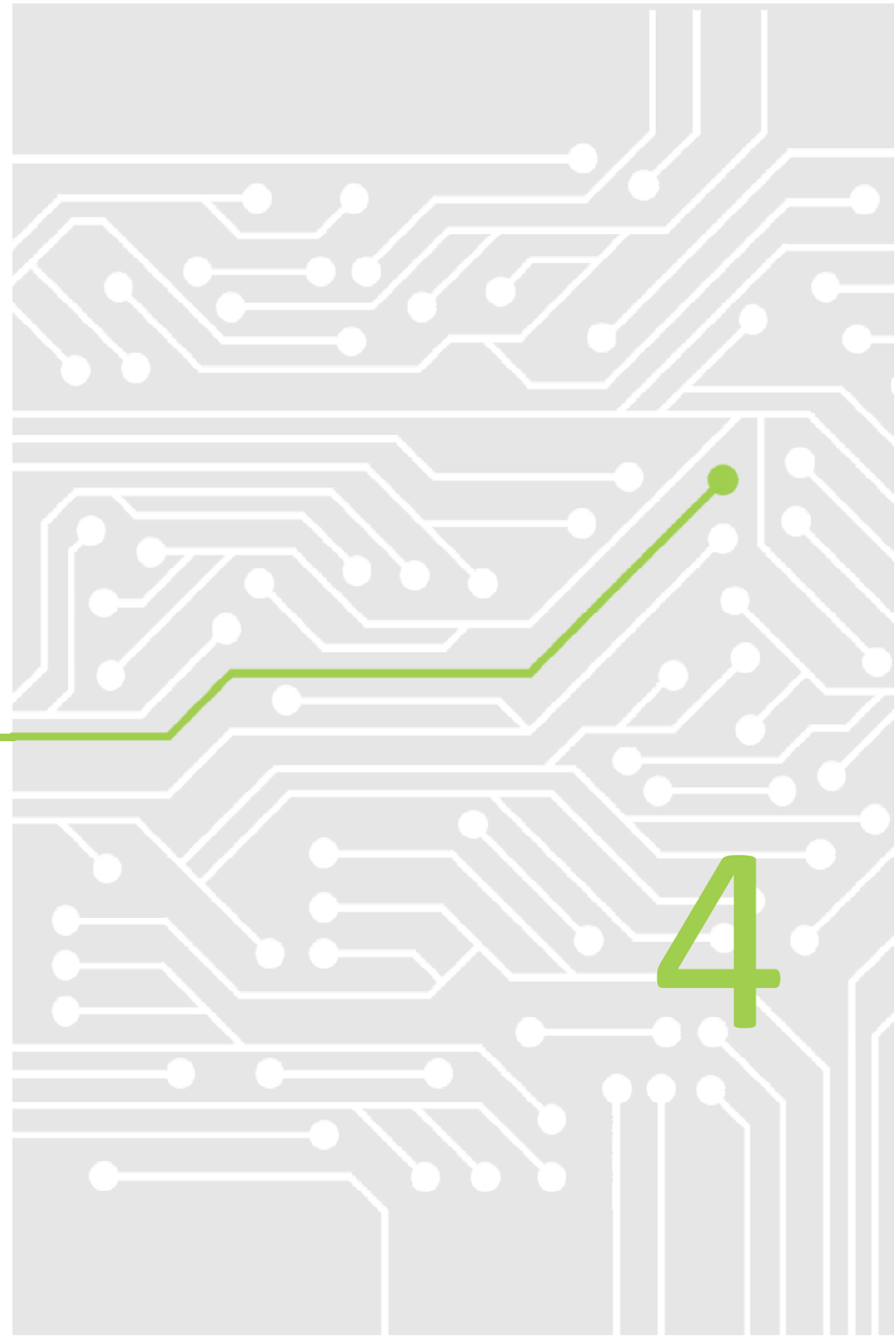
GNU Radio LimeSDR Sink



GNU Radio LimeSDR Source



Pothos



Pothos LimeSDR Source