RF & SDR Appendix — Full Project Code (LimeSDR Mini v2 / No-SDR)

All listings are complete and runnable. Default SDR: LimeSDR (SoapySDR driver=lime).

Windows paths use C:\code\..., Linux paths use /home/wofl/...

Project 1 — Wideband Wi■Fi/BT Sweep (LimeSDR)

Sweeps 2.400–2.500 GHz in 5 MHz steps and prints an ASCII PSD bar per tune.

Windows file: C:\code\sdr\projects\spectrum\wifi_sweep_24xx_lime.py
Linux file: /home/wofl/code/sdr/projects/spectrum/wifi_sweep_24xx_lime.py

```
#!/usr/bin/env python3
# wifi_sweep_24xx_lime.py — Sweep 2.400–2.500 GHz using LimeSDR (SoapySDR).
# Creates an ASCII spectrum per step. Rx-only.
# Requires: pip install SoapySDR numpy
import numpy as np, time, sys
import SoapySDR
from SoapySDR import *

CENTER_START = 2.400e9
CENTER_STOP  = 2.500e9
SAMP_RATE    = 2.5e6
FFT_N        = 4096
GAIN_DB      = 30.0
STEP_HZ      = 5e6
DEVICE       = "driver=lime"

def open_sdr():
    sdr = SoapySDR.Device(dict([kv.split("=") for kv in DEVICE.split(",")]))
    sdr.setSampleRate(SOAPY_SDR_RX, 0, SAMP_RATE)
    sdr.setGain(SOAPY_SDR_RX, 0, GAIN_DB)
    sdr.setAntenna(SOAPY_SDR_RX, 0, "LNAL")
    return sdr

def psd_at(sdr, f0):
    sdr.setFrequency(SOAPY_SDR_RX, 0, f0)
    rx = sdr.setupStream(SOAPY_SDR_RX, SOAPY_SDR_CF32)
    sdr.activateStream(rx)
    N = FFT_N * 8
    buff = np.empty(N, np.complex64)
    got = 0
    # collect a little burst
    while got < N:
        st = sdr.readStream(rx, [buff[got:]], N - got)
        if st.ret > 0:
            got += st.ret
        else:
            break
    sdr.deactivateStream(rx); sdr.closeStream(rx)
    if got < N: return None
    segs = buff.reshape(-1, FFT_N)
    win = np.hanning(FFT_N).astype(np.float32)
    psd = np.mean(np.abs(np.fft.fftshift(np.fft.fft(segs*win)))**2, axis=0)
    psd_db = 10*np.log10(psd + 1e-12)
    return psd_db

def bar(psd_db, cols=100):
    lo, hi = np.percentile(psd_db, 5), np.percentile(psd_db, 95)
    rng = max(hi - lo, 5.0)
    blocks = " ■■■■■■■■"
    idx = ((psd_db - lo)/rng*(len(blocks)-1)).clip(0,len(blocks)-1).astype(int)
    return "".join(blocks[i] for i in idx[:cols])

def main():
    sdr = open_sdr()
    f = CENTER_START
    print("center(MHz)  | spectrum")
    print("-"*112)
    while f <= CENTER_STOP:
        psd = psd_at(sdr, f)
        if psd is None:
            line = "<no data>"
        else:
            line = bar(psd, cols=100)
        print(f"{f/1e6:9.3f}  | {line}")
        f += STEP_HZ
```

```
if __name__ == "__main__":
    main()
```

Run:

```
py -m pip install SoapySDR numpy`npython C:\code\sdr\projects\spectrum\wifi_sweep_24xx_lime.py
pip install --user SoapySDR numpy && python3 /home/wofl/code/sdr/projects/spectrum/wifi_sweep_24xx_lime.py
```

Project 2 — WBFM Broadcast Receiver (LimeSDR)

Receives a broadcast FM station and plays audio at 48 kHz. UK de■emphasis 50 µs.

Windows file: C:\code\sdr\projects\receivers\wbfm_play_lime.py
Linux file: /home/wofl/code/sdr/projects/receivers/wbfm_play_lime.py

```
#!/usr/bin/env python3
# wbfm_play_lime.py — Play a WBFM broadcast via LimeSDR → PC speakers.
# Requires: pip install SoapySDR numpy sounddevice
import numpy as np, sounddevice as sd, math
import SoapySDR
from SoapySDR import *

CENTER = 99.9e6     # Change to local station
SAMP   = 1.92e6     # Convenient for decimation to 48k
GAIN   = 30
DEVICE = "driver=lime"

def fm_demod(iq):
    ph = np.unwrap(np.angle(iq))
    d  = np.diff(ph, prepend=ph[:1])
    return d

def deemph(x, fs, tau=50e-6):
    y = np.zeros_like(x, dtype=np.float32)
    a = math.exp(-1.0/(fs*tau)); b = 1.0 - a
    acc = 0.0
    for i,v in enumerate(x):
        acc = b*v + a*acc
        y[i] = acc
    return y

def main():
    sdr = SoapySDR.Device(dict([kv.split("=") for kv in DEVICE.split(",")]))
    sdr.setSampleRate(SOAPY_SDR_RX,0,SAMP)
    sdr.setGain(SOAPY_SDR_RX,0,GAIN)
    sdr.setAntenna(SOAPY_SDR_RX,0,"LNAL")
    sdr.setFrequency(SOAPY_SDR_RX,0,CENTER)

    rx = sdr.setupStream(SOAPY_SDR_RX, SOAPY_SDR_CF32)
    sdr.activateStream(rx)
    sd.default.samplerate = 48000
    sd.default.channels = 1
    decim = int(SAMP//240000) or 1
    audio_decim = int((SAMP//decim)//48000)

    with sd.OutputStream():
        while True:
            N = 256*1024
            buf = np.empty(N, np.complex64)
            st = sdr.readStream(rx, [buf], N)
            if st.ret <= 0: continue
            iq = buf[:st.ret]
            # Simple low-pass + decimate via FFT chunking could be added; for demo, crude slice
            dem = fm_demod(iq)
            # downsample directly to 48k (rough but works for a demo)
            step = int(SAMP//48000)
            audio = dem[::step]
            audio = deemph(audio, 48000, 50e-6)
            audio = np.tanh(audio*2.0).astype(np.float32)
            sd.play(audio, 48000, blocking=True)

    sdr.deactivateStream(rx); sdr.closeStream(rx)

if __name__ == "__main__":
    main()
```

Run:

```
py -m pip install SoapySDR numpy sounddevice`npython C:\code\sdr\projects\receivers\wbfm_play_lime.py
```

```
pip install --user SoapySDR numpy sounddevice && python3 /home/wofl/code/sdr/projects/receivers/wbfm_play_lim
```

Project 3 — Wi■Fi Channel Map (LimeSDR)

Measures average PSD around each 2.4 GHz channel center and prints a bar chart.

Windows file: C:\code\sdr\projects\wifi\channel_map_24_lime.py
Linux file: /home/wofl/code/sdr/projects/wifi/channel_map_24_lime.py

```
#!/usr/bin/env python3
# channel_map_24_lime.py — Estimate Wi■Fi channel occupancy (2.4 GHz) per channel.
import numpy as np
import SoapySDR
from SoapySDR import *

SAMP_RATE = 2.5e6
GAIN_DB   = 30
DEVICE    = "driver=lime"
CENTER    = 2.437e9
FFT_N     = 4096

CH_CENTERS = {ch: 2.412e9 + 5e6*(ch-1) for ch in range(1,14)}

def open_sdr():
    d = SoapySDR.Device(dict([kv.split("=") for kv in DEVICE.split(",")]))
    d.setSampleRate(SOAPY_SDR_RX,0,SAMP_RATE)
    d.setGain(SOAPY_SDR_RX,0,GAIN_DB)
    d.setAntenna(SOAPY_SDR_RX,0,"LNAL")
    return d

def measure(d, f0):
    d.setFrequency(SOAPY_SDR_RX,0,f0)
    rx = d.setupStream(SOAPY_SDR_RX, SOAPY_SDR_CF32)
    d.activateStream(rx)
    N = FFT_N*8
    buf = np.empty(N, np.complex64); got=0
    while got<N:
        st=d.readStream(rx,[buf[got:]],N-got)
        if st.ret>0: got+=st.ret
        else: break
    d.deactivateStream(rx); d.closeStream(rx)
    if got<N: return None
    segs = buf.reshape(-1,FFT_N)
    win = np.hanning(FFT_N).astype(np.float32)
    psd = np.mean(np.abs(np.fft.fftshift(np.fft.fft(segs*win)))**2,axis=0)
    freqs = np.linspace(f0 - SAMP_RATE/2, f0 + SAMP_RATE/2, FFT_N)
    return freqs, 10*np.log10(psd+1e-12)

def main():
    d = open_sdr()
    freqs, psd_db = measure(d, CENTER)
    results = []
    for ch,fc in CH_CENTERS.items():
        mask = (freqs>=fc-10e6)&(freqs<=fc+10e6)
        p = np.mean(psd_db[mask]) if np.any(mask) else -200
        results.append((ch, p))
    results.sort()
    lo = min(p for _,p in results); hi=max(p for _,p in results); rng=max(hi-lo,5)
    bars=" ■■■■■■■■"
    print("2.4GHz Wi■Fi channel occupancy (approx.):")
    for ch,p in results:
        level=int(np.clip((p-lo)/rng*(len(bars)-1),0,len(bars)-1))
        print(f"ch {ch:2d}: {bars[level]*40}  ({p:.1f} dB)")

if __name__=="__main__":
    main()
```

Run:

```
py -m pip install SoapySDR numpy`npython C:\code\sdr\projects\wifi\channel_map_24_lime.py
pip install --user SoapySDR numpy && python3 /home/wofl/code/sdr/projects/wifi/channel_map_24_lime.py
```

Project 4 — ADS■B 1090 MHz (Educational, LimeSDR)

Detects Mode■S preambles and prints raw bits/hex. For learning; not full CRC decode.

Windows file: C:\code\sdr\projects\adsb\adsb_1090_lime.py
Linux file: /home/wofl/code/sdr/projects/adsb/adsb_1090_lime.py

```
#!/usr/bin/env python3
# adsb_1090_lime.py — Educational ADS■B/Mode■S preamble detector + bit slicer.
# Not a full decoder, but will detect frames and print raw bits/hex.
# Requires: pip install SoapySDR numpy
import numpy as np, sys, binascii
import SoapySDR
from SoapySDR import *

FS   = 2_000_000      # 2 Msps
FC   = 1090_000_000   # 1090 MHz
GAIN = 30
DEVICE="driver=lime"

PRE_US = 8e-6
SYM_US = 1e-6
PRE_SAM = int(FS*PRE_US)
SYM_SAM = int(FS*SYM_US)

def open_sdr():
    s=SoapySDR.Device(dict([kv.split("=") for kv in DEVICE.split(",")]))
    s.setSampleRate(SOAPY_SDR_RX,0,FS)
    s.setGain(SOAPY_SDR_RX,0,GAIN)
    s.setAntenna(SOAPY_SDR_RX,0,"LNAL")
    s.setFrequency(SOAPY_SDR_RX,0,FC)
    rx=s.setupStream(SOAPY_SDR_RX,SOAPY_SDR_CF32); s.activateStream(rx)
    return s, rx

def magnitude(x): return (x.real*x.real + x.imag*x.imag)

def main():
    s,rx=open_sdr()
    try:
        N=FS//2
        while True:
            buff=np.empty(N,np.complex64)
            st=s.readStream(rx,[buff],N)
            if st.ret<=0: continue
            iq=buff[:st.ret]
            pwr=magnitude(iq).astype(np.float32)
            thr=np.mean(pwr)+3*np.std(pwr)
            # naive preamble: look for rising edge
            idx=np.where(pwr[1:]>thr)[0]
            for i0 in idx[:200]:  # limit search per block
                i=i0
                # grab window for bits after preamble
                start=i+PRE_SAM
                if start+112*SYM_SAM>=len(pwr):
                    continue
                bits=[]
                for b in range(112): # short frames
                    s0 = np.sum(pwr[start + b*SYM_SAM : start + b*SYM_SAM + SYM_SAM//2])
                    s1 = np.sum(pwr[start + b*SYM_SAM + SYM_SAM//2 : start + (b+1)*SYM_SAM])
                    bits.append(1 if s1>s0 else 0)
                # pack bits to hex for a peek
                by=bytearray()
                for j in range(0,len(bits),8):
                    acc=0
                    for k in range(8):
                        acc=(acc<<1)|(bits[j+k]&1)
                    by.append(acc)
                hx=binascii.hexlify(bytes(by)).decode()
                print("ADS-B bits:", "".join(str(b) for b in bits[:56]), "... hex:", hx[:28], "…")
    finally:
        s.deactivateStream(rx); s.closeStream(rx)
if __name__=="__main__":
```

```
    main()
```

Run:

```
py -m pip install SoapySDR numpy`npython C:\code\sdr\projects\adsb\adsb_1090_lime.py
pip install --user SoapySDR numpy && python3 /home/wofl/code/sdr/projects/adsb/adsb_1090_lime.py
```

Project 5 — NOAA APT (LimeSDR → WAV → Image)

Records 10 minutes around 137 MHz, FM-demods to 11025 Hz WAV. Process with noaa-apt.

Windows file: C:\code\sdr\projects\noaa\noaa_apt_record_lime.py
Linux file: /home/wofl/code/sdr/projects/noaa/noaa_apt_record_lime.py

```
#!/usr/bin/env python3
# noaa_apt_record_lime.py — Receive 137 MHz APT via LimeSDR, FM-demod, write 11025 Hz WAV.
# Then run: noaa-apt -i out.wav -o out.png
# Requires: pip install SoapySDR numpy soundfile
import numpy as np, soundfile as sf, math
import SoapySDR
from SoapySDR import *

CENTER = 137.1e6   # NOAA-19 example; change per pass
SAMP   = 240000    # baseband
GAIN   = 40
DEVICE = "driver=lime"

def fm_demod(iq):
    ph = np.unwrap(np.angle(iq))
    d  = np.diff(ph, prepend=ph[:1])
    return d

def deemph(x, fs, tau=50e-6):
    y=np.zeros_like(x,dtype=np.float32)
    a=math.exp(-1.0/(fs*tau)); b=1.0-a; acc=0.0
    for i,v in enumerate(x):
        acc=b*v+a*acc; y[i]=acc
    return y

def main():
    s=SoapySDR.Device(dict([kv.split("=") for kv in DEVICE.split(",")]))
    s.setSampleRate(SOAPY_SDR_RX,0,SAMP)
    s.setGain(SOAPY_SDR_RX,0,GAIN)
    s.setAntenna(SOAPY_SDR_RX,0,"LNAL")
    s.setFrequency(SOAPY_SDR_RX,0,CENTER)
    rx=s.setupStream(SOAPY_SDR_RX,SOAPY_SDR_CF32); s.activateStream(rx)

    total_secs=600  # 10 min capture window; adjust for pass
    out = []
    block = 262144
    got_samps = 0
    target = int(SAMP*total_secs)
    while got_samps < target:
        buf=np.empty(block,np.complex64)
        st=s.readStream(rx,[buf],block)
        if st.ret>0:
            iq=buf[:st.ret]
            fm=fm_demod(iq)
            # downsample to 11025
            step=int(SAMP//11025)
            audio=fm[::step]
            audio=deemph(audio,11025,50e-6).astype(np.float32)
            out.append(audio)
            got_samps += st.ret
        else:
            break
    s.deactivateStream(rx); s.closeStream(rx)
    if out:
        y=np.concatenate(out)
        sf.write("out.wav", y, 11025, subtype="PCM_16")
        print("WAV saved: out.wav — now run: noaa-apt -i out.wav -o out.png")

if __name__=="__main__":
    main()
```

Run:

```
py -m pip install SoapySDR numpy soundfile`npython C:\code\sdr\projects\noaa\noaa_apt_record_lime.py`nnoaa-ap
```

```
pip install --user SoapySDR numpy soundfile && python3 /home/wofl/code/sdr/projects/noaa/noaa_apt_record_lime
```