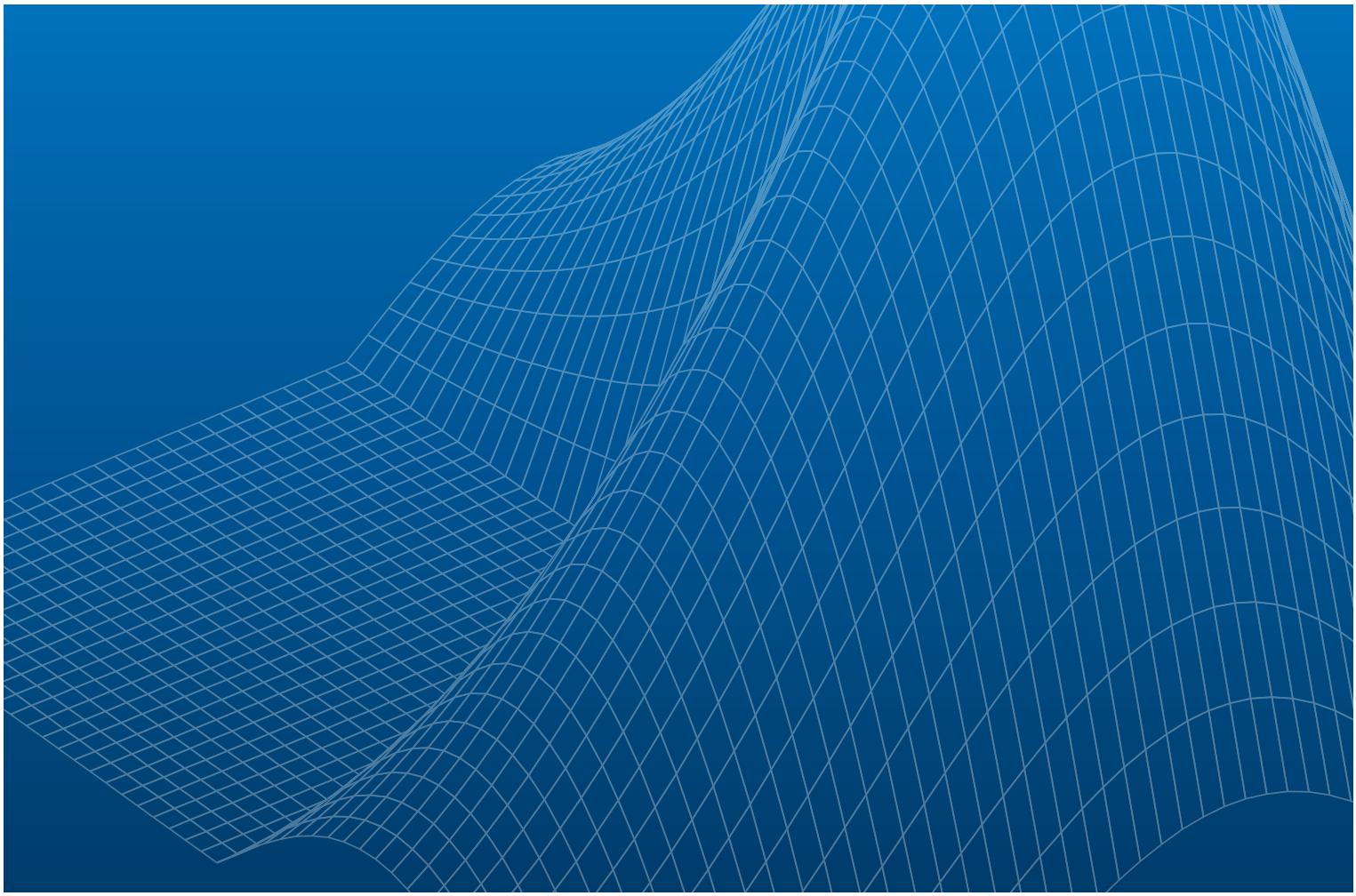

Decoding Live Wireless Signals with MATLAB and RTL-SDR

APPLICATION NOTES

MathWorks®

Commercial aircraft transmit automatic dependent surveillance – broadcast (ADS-B) messages to report position, velocity, altitude, and other flight data to air traffic control systems. This example shows how to use MATLAB® and software-defined radio (SDR) hardware to capture and process wireless data in real time using live ADS-B signals.

We use MATLAB, Communications System Toolbox™, and RTL-SDR hardware (based on the Realtek RTL2832U chipset) to receive the ADS-B broadcasts on a PC and then detect and decode the aircraft messages.

Capturing Wireless Signals with RTL-SDR

Many test setups for capturing wireless signals include test equipment that costs thousands of dollars. These setups often require several manual steps to first transfer the data from the instrument that captured it to a PC and then reformat the binary data into numeric values that can be readily processed. In contrast, we can use widely available, low-cost hardware and two MATLAB commands to access IQ samples directly in our MATLAB workspace.

The RTL-SDR is an inexpensive (approximately $20) piece of hardware that plugs into a USB port. It has a configurable center frequency of approximately 50 MHz to 2 GHz and a maximum bandwidth of approximately 3 MHz. Communications System Toolbox includes an add-on support package for RTL-SDR hardware (available for ***download***), which includes functions for configuring the RTL-SDR and accessing the data it receives.

RTL-SDR hardware can capture ADS-B wireless signals, which are transmitted at 1090 MHz and use pulse-position modulation to transmit data at 1 MBit/sec.

We set up the RTL-SDR to capture ADS-B signals with a single MATLAB command:

```
RX = comm.SDRRTLReceiver('0','CenterFrequency',1090e6,...
    'EnableTunerAGC',false,'TunerGain',60,'SampleRate',2.4e6,...
    'OutputDataType','single','SamplesPerFrame',262144,...
    'FrequencyCorrection',0);
```

Next, we acquire data from the RTL-SDR and create a MATLAB variable also with just one line of code:

```
IQSamples = step(RX);
```


MathWorks®

Detecting an ADS-B Transmission

After capturing ADS-B signals and acquiring the resulting data from the RTL-SDR, we create a time plot of the captured data (Figure 1).

```
plot(abs(IQSamples));
```

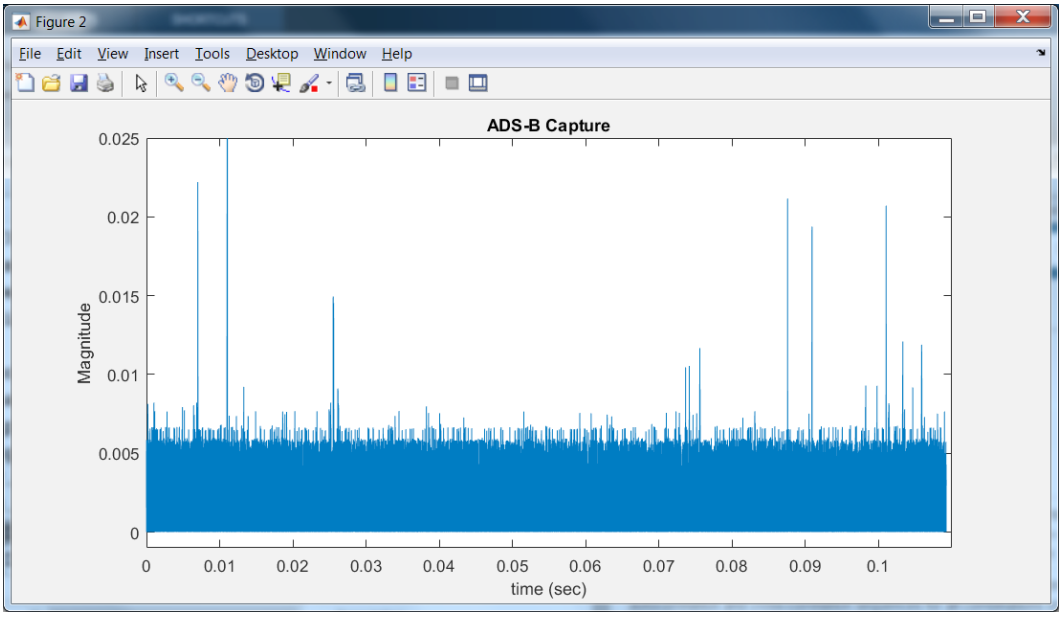

Figure 1. Time plot of wireless signals captured for 0.11 seconds with RTL-SDR hardware at 1090MHz.

At 1090 MHz, we typically capture a number of short duration signals and a significant amount of noise. Many of the high amplitude impulses shown in Figure 1 are legacy air traffic control waveforms, but by zooming in to inspect the different impulses we can see the signal at approximately 0.025 seconds is a decodable ADS-B message (Figure 2).

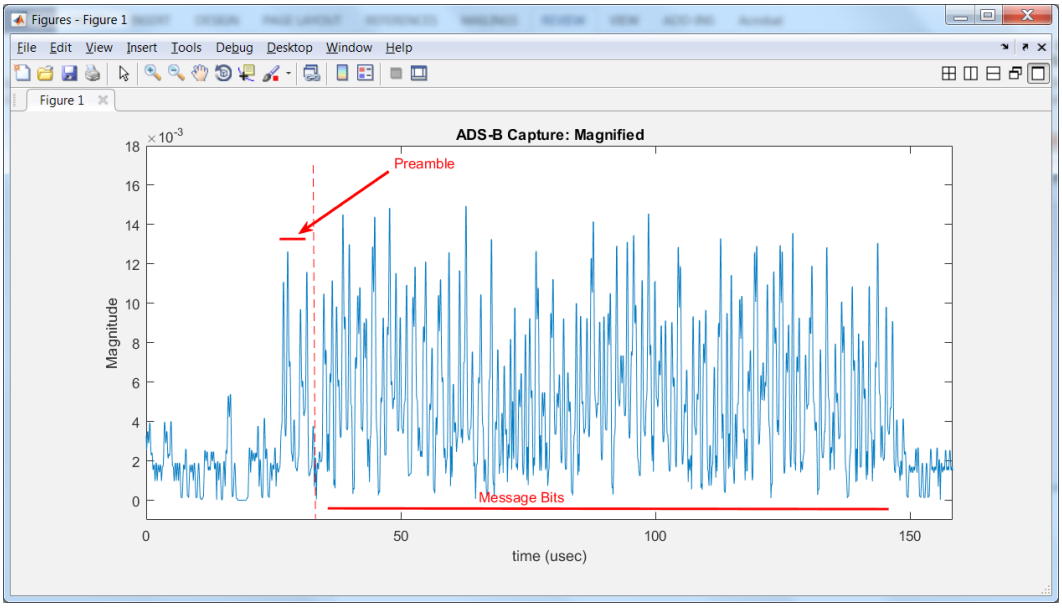

Figure 2. Time plot of an ADS-B signal.

The ADS-B signal has an 8-microsecond preamble followed by 112 message bits that are transmitted in the subsequent 112 microseconds (Figure 2). To identify potential ADS-B transmissions and align our bit detection algorithm to the first message bit, we need to correlate the captured signal to the ideal preamble pattern. First we upsample the signal to 12 MHz using the FIRInterpolator System object from DSP System Toolbox, and then we correlate the data with the known preamble pattern to determine the correct timing alignment.

```
% Upsample the IQ samples by 5x to improve time resolution
interpFIR=dsp.FIRInterpolator(5);
data = step(interpFIR,IQSamples);

% Calculate correlation to ADS-B preamble (Use this calcula-
tion to find message bit #1)
preamble = [ones(1,6) zeros(1,6) ones(1,6) zeros(1,24) ones(1,6)...
    zeros(1,6) ones(1,6)];
correlatedData = xcorr(preamble', flipud(double(abs(data))));
```

The results are shown in Figure 3. The upper plot shows a portion of the upsampled ADS-B capture, and the lower plot shows the result of the correlation to the preamble. The peak at sample 131 in the lower plot indicates the strongest match to the preamble pattern in the captured ADS-B signal. Since

the pattern we are using for the preamble correlation is 60 samples long and we know the message bits in the ADS-B message start 36 samples after the correlation peak we can identify the start of the ADS-B message bits at sample number 167.

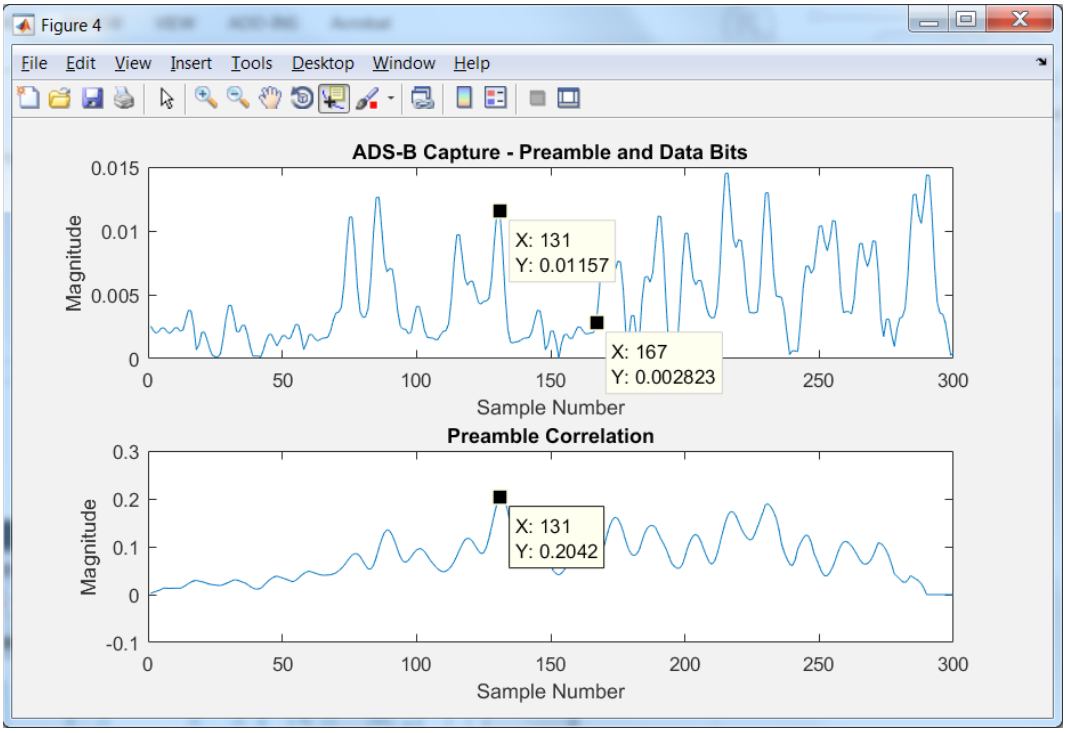

Figure 3. Plots showing upsampled ADS-B capture (top) and correlation to the ADS-B preamble (bottom).

Decoding the ADS-B Message

The bit decisions for the 112 message bits are made by pattern matching the 12 samples for each data bit to the two possible bit patterns. The resulting hex characters for the bit decisions are shown below:

```
rxMsg:  8D A7AD0D 60378784F313A7 3C92F4
```

The last six characters in this message (3C92F4) are a checksum that we use to determine if the preceding 88 bits form a valid message. We perform this test using the ADS-B checksum polynomial and the Communications System Toolbox CRCGenerator System object.

```
g = logical
([1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 1 0 0 1]);
crcADSB = comm.CRCGenerator(g);

y = step(crcADSB,m')';
```


MathWorks®

If the checksum computation confirms this is a valid message, we extract the data fields and interpret the values. In this ADS-B message, 24 bits represent the aircraft ID, 11 bits represent the altitude, 17 bits represent the latitude, and 17 bits represent the longitude. We cross-check this decoded message to flight information on flightradar24.com and verify that we captured data for an aircraft flying near the MathWorks headquarters in Natick, Massachusetts.

```
Aircraft ID A7AD0D is at altitude 11000
Aircraft ID A7AD0D is at latitude 41 58 43.0,
longitude -70 50 29.2
```

The code above forms the basis for a live aircraft tracking program, which is provided as an example in the RTL-SDR Hardware Support Package. A screen capture of the tracking display is shown in Figure 4.

```
ADSBAirplaneTrackingwithRTLSDR
```

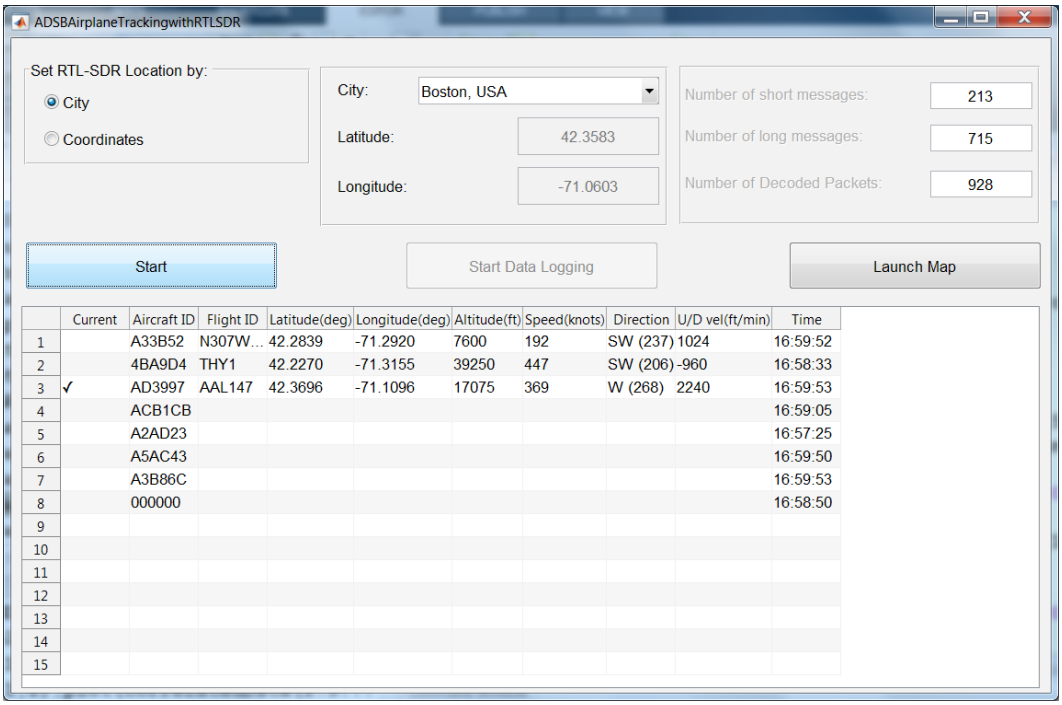

Figure 4. Live aircraft tracking data using ADSBAirplaneTrackingwithRTLSDR.m.

Conclusion

The RTL-SDR is a useful entry-level device for experimenting with the SDR capabilities of Communications System Toolbox. In addition to ADS-B, numerous other wireless signals can be captured and processed using this combination of hardware and software, including AM radio, FM radio, 2G GSM, 3G UMTS, and 4G LTE, among others. A free book, ***"Software Defined Radio Using MATLAB and Simulink and the RTL-SDR,"*** explores these topics in more detail with hands-on examples based on MATLAB and Simulink®.

The same workflow and MATLAB command syntax used in this example can be applied to more full-featured SDR hardware, including Ettus Radio USRP, Avnet PicoZed SDR, and the FMCOMMS3-based Zynq Radio. These SDR options can be used for applications that require higher bandwidths, wider tuning ranges, or multiple-input multiple-output (MIMO) support. Explore and download hardware support packages for Communications System Toolbox:

- ***URSP Support Package***
- ***FPGA Radio Support Package***
- ***Zynq SDR Support Package***

Products Used

MATLAB

Communications System Toolbox

DSP System Toolbox

© 2019 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See mathworks.com/trademarks for a list of additional trademarks.
Other product or brand names may be trademarks or registered trademarks of their respective holders.