gr-acars revisited

T. Lavarenne[1], J.-M Friedt[1,2]
[1] Lycée Jean Rostand, Villepinte, France
[2] FEMTO-ST Time & Frequency, Besançon, France

Abstract

As part of porting `gr-acars` to GNU Radio 3.8, new features are added including multi-channel analysis and symbol clock synchronization. The improvements brought by such functionalities are summarized here.

1 Introduction

In 2012 we wrote our first OOT GNU Radio processing block for decoding the Aircraft Communications Addressing and Reporting System (ACARS) [1]. Based on an audio-frequency shift keying (AFSK) encoding of the amplitude modulated carrier, the physical layer is probably amongst the easiest to grasp. Nevertheless, all challenges of digital communication decoding remain. While amplitude modulated (AM) signal does not require a coherent demodulation with the recovery of the carrier, the poor resistance of AM to additive noise still hints at the use of narrowband reception filters. One issue that was *not* addressed in the initial implementation of `gr-acars` was symbol clock synchronization.

Indeed, it was assumed that sampling every 20 samples the 48 kS/s signal transmitting at 2400 bps would allow for recovering the various bit states encoded either as one half of as 1200 Hz sine wave of a full 2400 Hz sine wave. This assumption is prone to phase errors which, if not corrected from the beginning, accumulate along the decoding and lead to signal loss after some time of message decoding. Thus, the original `gr-acars` was hardly ever able to recover a full message, loosing synchronization along the processing path as observed in [2].

8 years later, with the transition to GNU Radio 3.8 requiring a port of the OOT module, the processing algorithm is analyzed with hindsight and modified accordingly, as described in the following sections devoted to soft to hard bit conversion during which clock synchronization is considered, the message assembling and checksum analysis.

2 Experimental setup

ACARS is transmitted in the 137 MHz airband with, as usual for plane communication, amplitude modulation. The signal is collected using R820T2-based or Ettus Research B210 Software Defined Radio receivers, low-pass filtered to select only the narrowband (<5 kHz wide) ACARS signal, and send the resulting 48 kS/s datastream to the ACARS decoding block. As opposed to past implementation of the ACARS decoding algorithm analyzing a broadband signal for the 2400 Hz header requiring a continuous application of Fourier transforms for convolution, we now rely on a narrowband selection of each ACARS band – thanks to the wrapping of `libfftw` by GNU Radio allowing for multiple instances of the block running in parallel – to feed multiple instances of the `gr-acars` decoding block, and a rise in the signal standard deviation as a signature of an on-going transmission. Two convolutions with 1200 and 2400 Hz sine waves lasting two bit periods (1/1200 s) are processed over this datastream and low pass filtered to remove the frequency sum spectral component, in order to identify the two possible bitstates of the AFSK modulation. The resulting soft bits (continuous amplitude values) feed a clock synchronization function. This new clock-synchronization block searches for **unique bit state** (1 framed by two 0s or 0 framed by two 1s) and tracks the bit maximum. The clock tracks these maximum to synchronize. When similar bit states follow, the tracking is not as crucial to proper decoding and some offset is acceptable (Fig. 1). A threshold selects which bit state is the most probable but the soft bit is still used to track the maximum position of a single bit state.

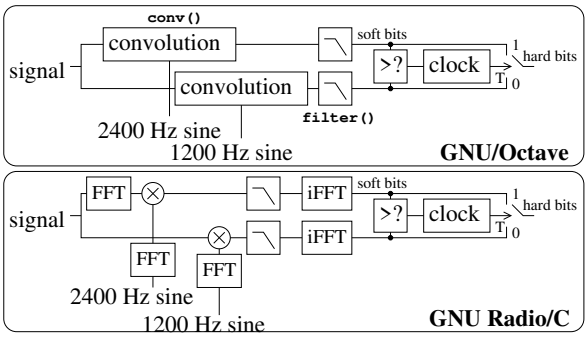

Figure 1: Algorithm implementation using GNU/Octave functions (top) and expanding the convolution as Fourier transform product to insert the filtering in the frequency domain in the C implementation (bottom).

3 Results

The hard bits encode a state transition from the current state to the next: any error along the decoding will flip all bits throughout the message unless a second error corrects for the initial one. Hence, soft to hard bit assignation is a critical requirement to recover a complete sentence. Such an assignment is only possible with proper datastream clock synchronization. Figure 2 illustrates two cases in which clock tracking is active. As long as constant bitstates follow (top after the header) little tracking is needed, but when bitstates occur offten (bottom, after sample 800) then tracking is mandatory to remain in the center of each time slot.

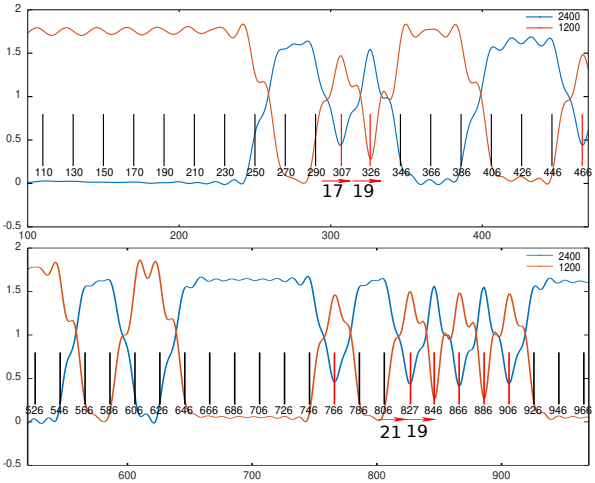

Figure 2: Two soft-bit sequences following the convolution with 2400 Hz (blue) and 1200 (red) bandpass filters. Black vertical lines indicate when successive hard bits are expected to be the same, **red vertical lines indicate when a hard bit is framed with two different hard-bit states and are hence used for clock synchronization**.

As can be seen, the nominal 48000/2400 = 20 samples (48 kS/s sampling rate and 2400 bps bitrate) departs from when analyzing the softbit maximum before thresholding to reach hard bits, defined as whether the 1200 Hz component is above or below the 2400 Hz component.

In most cases, if the initial transition position is properly identified, incrementing the datarate clock with a fixed step will result in proper decoding of the message. However the chances of one misleading bit corrupting the whole decoding sequence rises with longer messages, and the information shown in Fig. 3 is corrupted if no clock synchronization is adopted (Fig. 4.

The erroneous characters might have been analyzed by checking the parity bit since the properly

```
+*2.F-GZNOH12D23CAF4195#DFB 965  965000011 50 50
 60 62318319148145 07 08 00 26 -06-310-308-309-310-...
-239-239-239-310-310-311-309-310 016 017-392-391-026
 000 001109113005029 000 000 000 000 00102
```

Figure 3: A full ACARS message properly decoded thanks to clock synchronization

```
+*2.F-GZNOH12D23CAF4195#DFB 965  965000011 50 50
 60 62318319148145 07 08 00 26 -06-310-308-309-310-...
-239-239-239-310-310-311-309-310 016 017-392-391-026
 000 00Q.09..*POU/*: P/0 P/P_/P/ P/P_/PQP|uU
```

Figure 4: The ACARS message shown in Fig. 3 but with constant bitrate clock increment: the messages ends up garbled.

decoded message does not show any corrupted parity bit all the way to the end of the message (Fig. 5) while the message with constant clock rate ending up garbled does show some corruption of the parity bit (Fig. 6). However, the parity bit is not enough to identify the position of the switched bit, or worse is insensitive to an even number of errors, and cannot be used for error correction.

```
CRC:0000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000000011
```

Figure 5: Parity bit state for the properly decoded message with datarate clock correction.

```
CRC:0000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000000000000000000000000011000011011100110011101
1011101101011
```

Figure 6: Parity bit state for the improperly decoded message with datarate clock correction: constant datarate leads to some desynchronization shown as non-0 parity bits.

Out of 42 sentences collected over 6 minutes from a location a few kilometers from Charles de Gaulle airport in Paris, 15 would have been corrupted if decoded without clock synchronization, or a **35% success rate improvement**.

Multiple streams processed in parallel

Because AM detection does not require coherent demodulation (i.e. recovering the carrier frequency), a broadband filter collecting all airband signals and feeding the ACARS demodulator block

is sufficient for demodulating the European main frequency (131.725 MHz), the European secondary frequency (131.525 MHz) and the additional European frequency (131.825 MHz). However, such broadband integration makes ACARS prone to multiple interference sources and accumulates thermal energy over a broadband. Narrowband filtering and Frequency Xlating FIR filtering each band back to baseband improves signal to noise ratio and hence decoding efficiency. Since `gr-acars` relies heavily on FFT3W multithreaded Fourier transform library for implementing convolutions, a solution allowing for multiple instances of the FFT running on parallel streams is implemented.

`gr-acars` source code, including the port to GNU Radio 3.8, is available at `https://sourceforge.net/projects/gr-acars/`: the clock correcting algorithm is available in the `3.8ng` directory.

References

[1] J.-M Friedt, G. Goavec-Merou, *La réception radiofréquence définie par logiciel (Software Defined Radio – SDR)*, GNU/Linux Magazine France **153** pp.4-33 (2012), translated at `http://jmfriedt.free.fr/en_sdr.pdf`

[2] B. Chamaillard, M. Lastera & D. Roque, *A flexible VHF-band aeronautical datalink receiver based on software defined radio*, IEEE Aerospace and Electronic Systems Magazine **33**(1) 58–61 (2018) 2018