A new software-defined radio receiver for the BRAMS network

Replacement of the analogue ICOM receiver by a software-defined radio

Michel Anciaux

Royal Belgian Institute for Space Aeronomy

6th December 2018

Introduction

Many stations in the BRAMS network have experienced problems with their analogue receiver. Until now, all failed units could be either repaired or replaced but this is not sustainable. A new type of receiver must now be chosen.

Introduction

Many stations in the BRAMS network have experienced problems with their analogue receiver. Until now, all failed units could be either repaired or replaced but this is not sustainable. A new type of receiver must now be chosen.
A first candidate based on a software-defined radio has been evaluated and its suitability will be presented here.

Current receiving stations

A typical station consists of the following:

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable
- ICOM receiver

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable
- ICOM receiver
- calibrator

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable
- ICOM receiver
- calibrator
- GPS receiver

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable
- ICOM receiver
- calibrator
- GPS receiver
- data acquisition system:

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable
- ICOM receiver
- calibrator
- GPS receiver
- data acquisition system:
  - 2-channel audio sampler: signals from ICOM and GPS

Current receiving stations

A typical station consists of the following:

- Yagi antenna + cable
- ICOM receiver
- calibrator
- GPS receiver
- data acquisition system:
  - 2-channel audio sampler: signals from ICOM and GPS
  - Windows PC + software

Issues with Current ICOM receiver

Issues

Issues with Current ICOM receiver

Issues

- reliability

Issues with Current ICOM receiver

Issues

- reliability
- availability

Issues with Current ICOM receiver

Issues

- reliability
- availability
- limited performance

Reliability issue with ICOM receiver

Why has reliability become a concern?

Reliability issue with ICOM receiver

Why has reliability become a concern?

- ageing: most receivers have been operating continuously for several years

Reliability issue with ICOM receiver

Why has reliability become a concern?

- ageing: most receivers have been operating continuously for several years
- many have already failed (18), some could be repaired (13 so far)

Reliability issue with ICOM receiver

Why has reliability become a concern?

- ageing: most receivers have been operating continuously for several years
- many have already failed (18), some could be repaired (13 so far)
- most failures have been of the same type (sharp drop in sensitivity) but new types of degradation have already been observed (on at least 2 units)

Reliability issue with ICOM receiver

Why has reliability become a concern?

- ageing: most receivers have been operating continuously for several years
- many have already failed (18), some could be repaired (13 so far)
- most failures have been of the same type (sharp drop in sensitivity) but new types of degradation have already been observed (on at least 2 units)
- This can only get worse !

Availability issue

Why is replacing the receiver becoming impossible?

Availability issue

Why is replacing the receiver becoming impossible?

- This model is no longer produced

Availability issue

Why is replacing the receiver becoming impossible?

- This model is no longer produced
- Alternative analogue models are considerably more expensive

Availability issue

Why is replacing the receiver becoming impossible?

- This model is no longer produced
- Alternative analogue models are considerably more expensive
- The market trend is that analogue receivers are being replaced by software defined radios

Performance limitations

There are also some limitation with the current receivers:

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)
  - impossible to observe weak and strong echoes simultaneously

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)
    - impossible to observe weak and strong echoes simultaneously
    - prone to desensitizing by strong interferers

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)
  - impossible to observe weak and strong echoes simultaneously
  - prone to desensitizing by strong interferers
- sensitivity: noise temperature ≈1000 K

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)
  - impossible to observe weak and strong echoes simultaneously
  - prone to desensitizing by strong interferers
- sensitivity: noise temperature ≈1000 K
- frequency stability, visible as a drift of the beacon in the spectrograms:

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)
  - impossible to observe weak and strong echoes simultaneously
  - prone to desensitizing by strong interferers
- sensitivity: noise temperature $\approx$1000 K
- frequency stability, visible as a drift of the beacon in the spectrograms:
  - LO strongly dependent on temperature (10 Hz/°C)

Performance limitations

There are also some limitation with the current receivers:

- limited dynamic range (36 dB, P1db = -102 dBm, ENOB = 6.0)
  - impossible to observe weak and strong echoes simultaneously
  - prone to desensitizing by strong interferers
- sensitivity: noise temperature ≈1000 K
- frequency stability, visible as a drift of the beacon in the spectrograms:
  - LO strongly dependent on temperature (10 Hz/°C)
  - LO cannot be locked to a reference

Issues with Current ICOM receiver

Issues

- reliability
- availability
- limited performance

Issues with Current ICOM receiver

Issues

- reliability
- availability
- limited performance

A new type of receiver is needed

- to replace the ICOM
- to improve upon its performance

while keeping the cost low

Alternatives for a new receiver

Here are the alternatives that have been identified:

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
    - new analogue receiver

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
  - new analogue receiver
  - complete SDR receiver (using its analogue audio output)

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
  - new analogue receiver
  - complete SDR receiver (using its analogue audio output)
- replacing the receiver and the data acquisition system

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
  - new analogue receiver
  - complete SDR receiver (using its analogue audio output)
- replacing the receiver and the data acquisition system
  - complete commercial solution (e.g. Ettus Research): too expensive

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
    - new analogue receiver
    - complete SDR receiver (using its analogue audio output)
- replacing the receiver and the data acquisition system
    - complete commercial solution (e.g. Ettus Research): too expensive
    - low cost commercial SDR front-end and linux-based computer: under investigation

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
  - new analogue receiver
  - complete SDR receiver (using its analogue audio output)
- replacing the receiver and the data acquisition system
  - complete commercial solution (e.g. Ettus Research): too expensive
  - low cost commercial SDR front-end and linux-based computer: under investigation
  - purpose built SDR front-end and linux-based computer: fall-back solution

Alternatives for a new receiver

Here are the alternatives that have been identified:

- replacing the receiver only (while keeping the data acquisition system)
  - new analogue receiver
  - complete SDR receiver (using its analogue audio output)
- replacing the receiver and the data acquisition system
  - complete commercial solution (e.g. Ettus Research): too expensive
  - low cost commercial SDR front-end and linux-based computer: under investigation
  - purpose built SDR front-end and linux-based computer: fall-back solution

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2
  - Airspy

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2
  - Airspy
- linux system:

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2
  - Airspy
- linux system:
  - PC

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2
  - Airspy
- linux system:
  - PC
  - Raspberry Pi

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2
  - Airspy
- linux system:
  - PC
  - Raspberry Pi
  - Beaglebone

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
  - Funcube Dongle Pro+
  - SDR Play RSP2
  - Airspy
- linux system:
  - PC
  - Raspberry Pi
  - Beaglebone
  - ...

SDR front-end with linux-based data acquisition system

Several low-cost commercial hardware solutions have been considered.

- front-end SDR:
    - Funcube Dongle Pro+
    - SDR Play RSP2
    - Airspy
- linux system:
    - PC
    - Raspberry Pi
    - Beaglebone
    - ...

Hardware Configuration

The hardware that was tested consists of:

Hardware Configuration

The hardware that was tested consists of:

- Rasperrypi3 modelB (RPi) single board computer, quad core ARM processor running linux

Hardware Configuration

The hardware that was tested consists of:

- Rasperrypi3 modelB (RPi) single board computer, quad core ARM processor running linux
- FUNcube Dongle Pro+ (FCDPP) 16-bit I and Q baseband signals on USB port

Hardware Configuration

The hardware that was tested consists of:

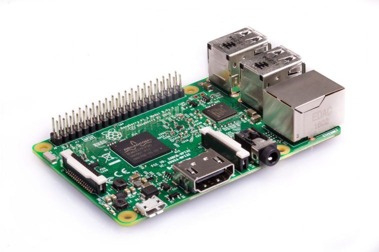
- Rasperrypi3 modelB (RPi) single board computer, quad core ARM processor running linux
- FUNcube Dongle Pro+ (FCDPP) 16-bit I and Q baseband signals on USB port
- Garmin GPS receiver used to discipline the RPi clock

Hardware Configuration

The hardware that was tested consists of:

- Rasperrypi3 modelB (RPi) single board computer, quad core ARM processor running linux
- FUNcube Dongle Pro+ (FCDPP) 16-bit I and Q baseband signals on USB port
- Garmin GPS receiver used to discipline the RPi clock
- Dedicated electronic interface to feed the NMEA frames and the 1-PPS signal to the RPi

Raspberry Pi 3B and Funcube Pro+



Software configuration

The bespoke software has the following characteristics:

Software configuration

The bespoke software has the following characteristics:

- multithreaded program written in C

Software configuration

The bespoke software has the following characteristics:

- multithreaded program written in C
- uses the ALSA API to read I/Q data over USB

Software configuration

The bespoke software has the following characteristics:

- multithreaded program written in C
- uses the ALSA API to read I/Q data over USB
- uses the HID API to configure the Funcube over USB

Software configuration

The bespoke software has the following characteristics:

- multithreaded program written in C
- uses the ALSA API to read I/Q data over USB
- uses the HID API to configure the Funcube over USB
- decimates the data, resulting in a sampling rate of 6 kHz

Software configuration

The bespoke software has the following characteristics:

- multithreaded program written in C
- uses the ALSA API to read I/Q data over USB
- uses the HID API to configure the Funcube over USB
- decimates the data, resulting in a sampling rate of 6 kHz
- detects the upper side band (bandwidth: 2700 Hz)

Software configuration

The bespoke software has the following characteristics:

- multithreaded program written in C
- uses the ALSA API to read I/Q data over USB
- uses the HID API to configure the Funcube over USB
- decimates the data, resulting in a sampling rate of 6 kHz
- detects the upper side band (bandwidth: 2700 Hz)
- saves the data in a 300-second WAV file with time stamps (current BRAMS format)

Software configuration

The bespoke software has the following characteristics:

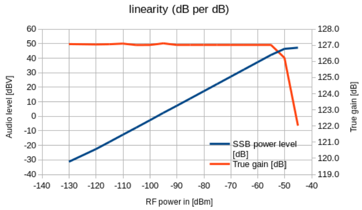
- multithreaded program written in C
- uses the ALSA API to read I/Q data over USB
- uses the HID API to configure the Funcube over USB
- decimates the data, resulting in a sampling rate of 6 kHz
- detects the upper side band (bandwidth: 2700 Hz)
- saves the data in a 300-second WAV file with time stamps (current BRAMS format)
- ntpd configured to synchronise the system clock to the GPS signal

Receiver sensitivity

The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)

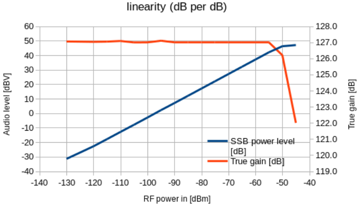
Receiver sensitivity

The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)



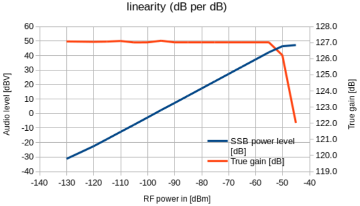
Receiver sensitivity

The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)



Receiver sensitivity

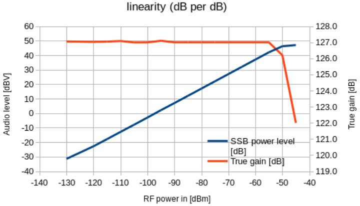
The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)



- SSB noise temperature: ≈430 K

Receiver sensitivity

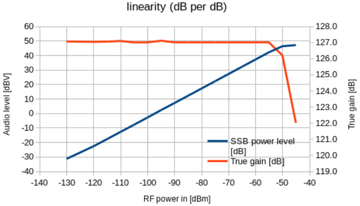
The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)



- SSB noise temperature: ≈430 K
- dynamic range: 61 dB, P1db = -50 dBm

Receiver sensitivity

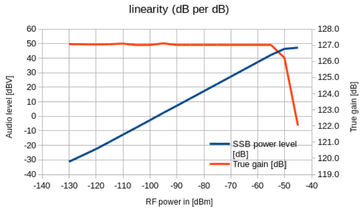
The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)



- SSB noise temperature: ≈430 K
- dynamic range: 61 dB, P1db = -50 dBm

Receiver sensitivity

The front-end performance was measured with the nominal configuration for BRAMS (LO freq=49.96 MHz, upper sideband, IF gain=0dB)



- SSB noise temperature: $\approx$430 K
- dynamic range: 61 dB, P1db = -50 dBm

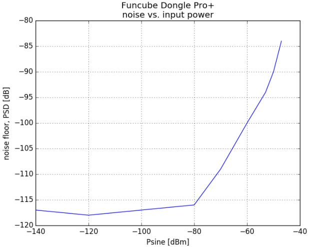
This is much better than the ICOM !

Noise floor depends on input power

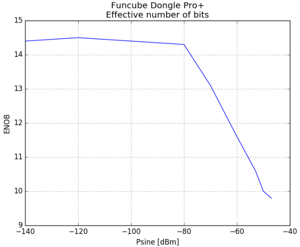
The noise floor increases with the power of the input signal thus desensitizing the front-end.

Noise floor depends on input power

The noise floor increases with the power of the input signal thus desensitizing the front-end.



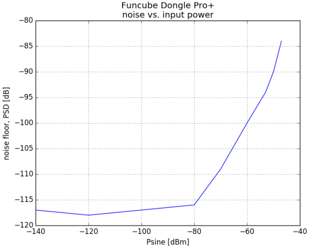
noise power spectral density constant below -80 dBm



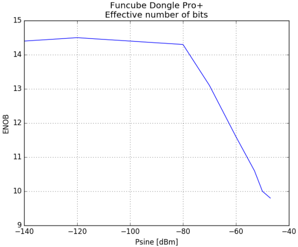
ENOB: 14.3 but decreases above -80 dBm

Noise floor depends on input power

The noise floor increases with the power of the input signal thus desensitizing the front-end.



noise power spectral density constant below -80 dBm



ENOB: 14.3 but decreases above -80 dBm

This is still better than the ICOM !

Frequency stability

The Funcube exhibits frequency drifts (as does the ICOM receiver).

Frequency stability

The Funcube exhibits frequency drifts (as does the ICOM receiver).

- The local oscillator frequency has a significant temperature dependence (4.6 Hz / °C).

Frequency stability

The Funcube exhibits frequency drifts (as does the ICOM receiver).

- The local oscillator frequency has a significant temperature dependence (4.6 Hz / °C).
- It cannot be locked to an external reference.

Frequency stability

The Funcube exhibits frequency drifts (as does the ICOM receiver).

- The local oscillator frequency has a significant temperature dependence (4.6 Hz / °C).
- It cannot be locked to an external reference.

Frequency stability

The Funcube exhibits frequency drifts (as does the ICOM receiver).

- The local oscillator frequency has a significant temperature dependence (4.6 Hz / °C).
- It cannot be locked to an external reference.

To mitigate this effect, the Funcube should have its temperature stabilised (not a major hurdle thanks to its very low mass).

Time stamping

As with the current set up, the data must be timestamped with sub-millisecond accuracy.

Time stamping

As with the current set up, the data must be timestamped with sub-millisecond accuracy.

- The current method of simultaneously sampling the receiver signal and the GPS signal(NMEA frame + 1-PPS) is not usable.

Time stamping

As with the current set up, the data must be timestamped with sub-millisecond accuracy.

- The current method of simultaneously sampling the receiver signal and the GPS signal(NMEA frame + 1-PPS) is not usable.
    - the data from the funcube is digital and does not go through an audio channel

Time stamping

As with the current set up, the data must be timestamped with sub-millisecond accuracy.

- The current method of simultaneously sampling the receiver signal and the GPS signal(NMEA frame + 1-PPS) is not usable.
    - the data from the funcube is digital and does not go through an audio channel
    - simultaneous sampling of GPS and radio signals cannot be guaranteed

Time stamping

As with the current set up, the data must be timestamped with sub-millisecond accuracy.

- The current method of simultaneously sampling the receiver signal and the GPS signal(NMEA frame + 1-PPS) is not usable.
  - the data from the funcube is digital and does not go through an audio channel
  - simultaneous sampling of GPS and radio signals cannot be guaranteed
- A new method is needed.

New time stamping method

Usable timestamps can be produced by doing the following:

New time stamping method

Usable timestamps can be produced by doing the following:

- Save a time stamp for every 1000 samples read from the Funcube.

New time stamping method

Usable timestamps can be produced by doing the following:

- Save a time stamp for every 1000 samples read from the Funcube.
- Keep the linux-system clock synchronised with the GPS 1-PPS.

New time stamping method

Usable timestamps can be produced by doing the following:

- Save a time stamp for every 1000 samples read from the Funcube.
- Keep the linux-system clock synchronised with the GPS 1-PPS.
- Keep the scheduling jitter low by minimizing activity.

New time stamping method

Usable timestamps can be produced by doing the following:

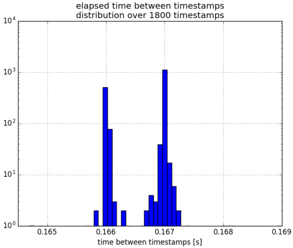
- Save a time stamp for every 1000 samples read from the Funcube.
- Keep the linux-system clock synchronised with the GPS 1-PPS.
- Keep the scheduling jitter low by minimizing activity.
- Rely on the stability of the sampling rate inside the Funcube.

Use of time stamps

The timestamps are affected by the scheduling jitter.

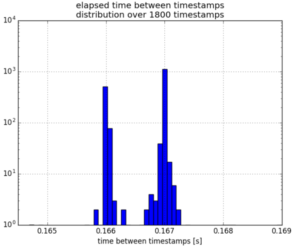
Use of time stamps

The timestamps are affected by the scheduling jitter.



Use of time stamps

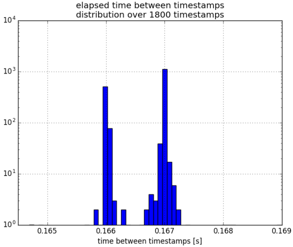
The timestamps are affected by the scheduling jitter.



- mean time difference: 166.7 ms as expected

Use of time stamps

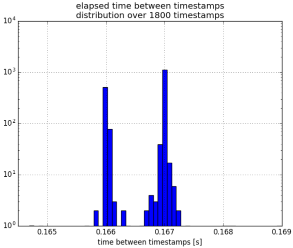
The timestamps are affected by the scheduling jitter.



- mean time difference: 166.7 ms as expected
- 2 modes

Use of time stamps

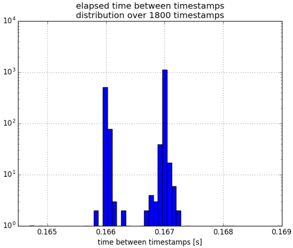
The timestamps are affected by the scheduling jitter.



- mean time difference: 166.7 ms as expected
- 2 modes
- occasional excursions of a few ms

Use of time stamps

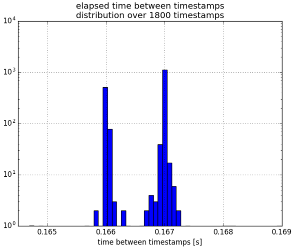
The timestamps are affected by the scheduling jitter.



- mean time difference: 166.7 ms as expected
- 2 modes
- occasional excursions of a few ms

Use of time stamps

The timestamps are affected by the scheduling jitter.

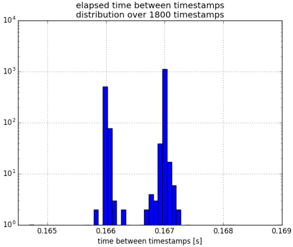


- mean time difference: 166.7 ms as expected
- 2 modes
- occasional excursions of a few ms

- A correction can be applied by performing a linear regression on the timestamps.

Use of time stamps

The timestamps are affected by the scheduling jitter.

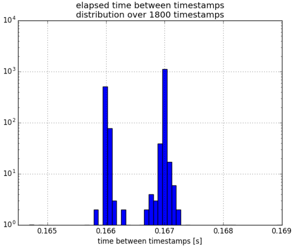


- mean time difference: 166.7 ms as expected
- 2 modes
- occasional excursions of a few ms

- A correction can be applied by performing a linear regression on the timestamps.
- This should work because the sampling rate is constant, the system clock has a very low jitter ($<1\mu s$) and there is a time stamp every 1000 samples.

Use of time stamps

The timestamps are affected by the scheduling jitter.



- mean time difference: 166.7 ms as expected
- 2 modes
- occasional excursions of a few ms

- A correction can be applied by performing a linear regression on the timestamps.
- This should work because the sampling rate is constant, the system clock has a very low jitter ($<1\mu$s) and there is a time stamp every 1000 samples.
- This must, of course, be tested.

Testing of time stamping method

In order to validate the time stamping method, a test was devised:

Testing of time stamping method

In order to validate the time stamping method, a test was devised:

- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.

Testing of time stamping method

In order to validate the time stamping method, a test was devised:

- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.
- The demodulated envelope was recovered from the upper side band.

Testing of time stamping method

In order to validate the time stamping method, a test was devised:

- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.
- The demodulated envelope was recovered from the upper side band.
- Successive rising edges should occur at 1-second intervals.

Testing of time stamping method

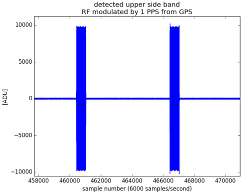
In order to validate the time stamping method, a test was devised:

- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.
- The demodulated envelope was recovered from the upper side band.
- Successive rising edges should occur at 1-second intervals.

Testing of time stamping method

In order to validate the time stamping method, a test was devised:

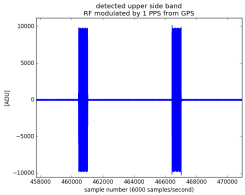
- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.
- The demodulated envelope was recovered from the upper side band.
- Successive rising edges should occur at 1-second intervals.

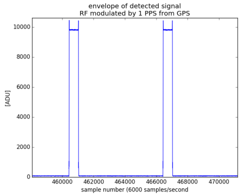


Testing of time stamping method

In order to validate the time stamping method, a test was devised:

- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.
- The demodulated envelope was recovered from the upper side band.
- Successive rising edges should occur at 1-second intervals.

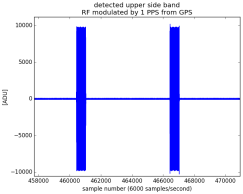


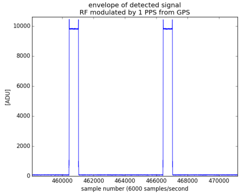


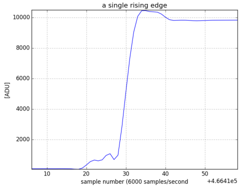
Testing of time stamping method

In order to validate the time stamping method, a test was devised:

- The input RF signal (49.97 MHz) was modulated by the 1-PPS from the GPS receiver.
- The demodulated envelope was recovered from the upper side band.
- Successive rising edges should occur at 1-second intervals.





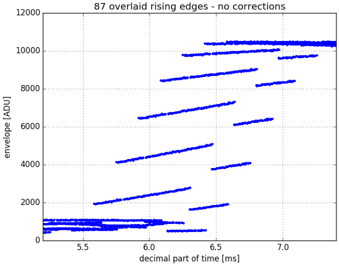


Results of time stamping test

Ideally all rising edges should occur at the same decimal part of a second. The initial jitter should be reduced by the time correction.

Results of time stamping test

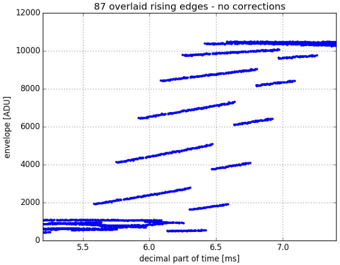
Ideally all rising edges should occur at the same decimal part of a second. The initial jitter should be reduced by the time correction.



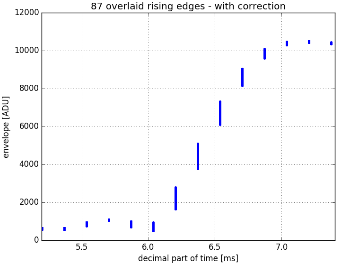
Without time correction

Results of time stamping test

Ideally all rising edges should occur at the same decimal part of a second. The initial jitter should be reduced by the time correction.



Without time correction



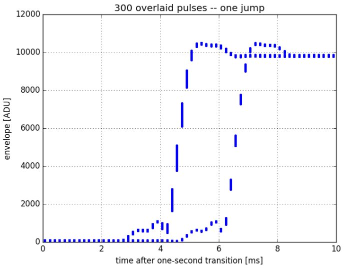
With time correction

Anomaly

Once the time correction has been applied, all the rising edges should occur at the same time after the one-second transition. Unfortunately, that is not always the case.

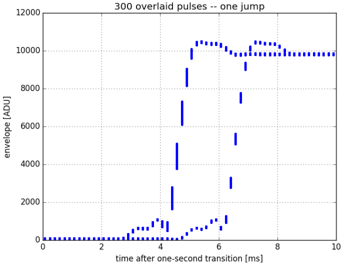
Anomaly

Once the time correction has been applied, all the rising edges should occur at the same time after the one-second transition. Unfortunately, that is not always the case.



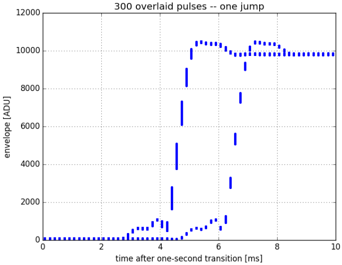
Anomaly

Once the time correction has been applied, all the rising edges should occur at the same time after the one-second transition. Unfortunately, that is not always the case.



Anomaly

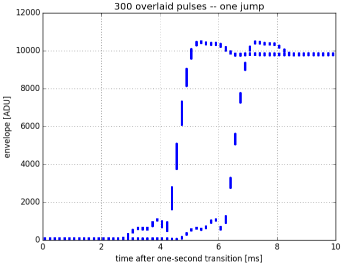
Once the time correction has been applied, all the rising edges should occur at the same time after the one-second transition. Unfortunately, that is not always the case.



- 2 modes

Anomaly

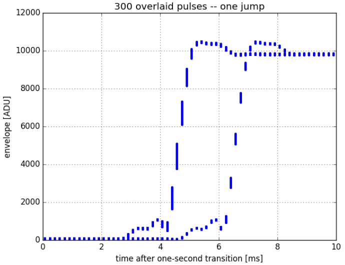
Once the time correction has been applied, all the rising edges should occur at the same time after the one-second transition. Unfortunately, that is not always the case.



- 2 modes
- there is a 2-ms jump

Anomaly

Once the time correction has been applied, all the rising edges should occur at the same time after the one-second transition. Unfortunately, that is not always the case.



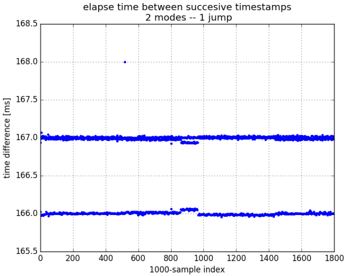
- 2 modes
- there is a 2-ms jump
- caused by dropped samples

Dropped samples

Evidence of the dropped samples can be found by observing the elapsed time between consecutive time stamps.

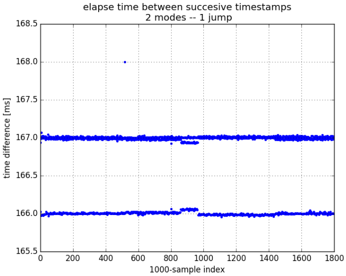
Dropped samples

Evidence of the dropped samples can be found by observing the elapsed time between consecutive time stamps.



Dropped samples

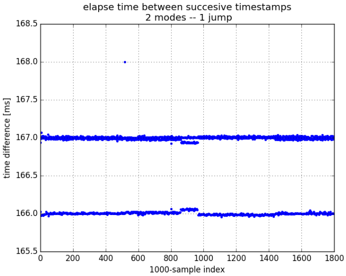
Evidence of the dropped samples can be found by observing the elapsed time between consecutive time stamps.



- nominal time difference: 167 ms

Dropped samples

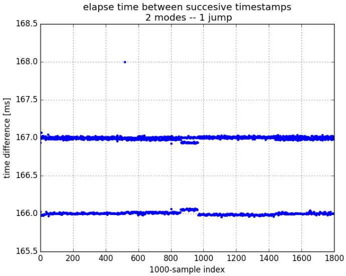
Evidence of the dropped samples can be found by observing the elapsed time between consecutive time stamps.



- nominal time difference: 167 ms
- 2 modes due to scheduling jitter

Dropped samples

Evidence of the dropped samples can be found by observing the elapsed time between consecutive time stamps.



- nominal time difference: 167 ms
- 2 modes due to scheduling jitter
- 1 jump at index $\approx$510: dropped samples ==> increased time between timestamps

Identifying the jumps

Fortunately, the jumps come in multiples of 2 ms and can readily be identified.

- observe the time between time stamps

Identifying the jumps

Fortunately, the jumps come in multiples of 2 ms and can readily be identified.

- observe the time between time stamps
- apply some smoothing function

Identifying the jumps

Fortunately, the jumps come in multiples of 2 ms and can readily be identified.

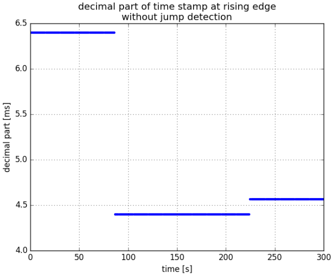
- observe the time between time stamps
- apply some smoothing function
- find discontinuities greater than 1 ms and mark the corresponding time stamps as dubious

Results of time stamping test with jump identification

The effect of the jump detection can be seen by looking at the time of the rising edge as a function of time.

Results of time stamping test with jump identification

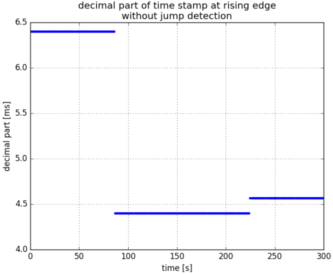
The effect of the jump detection can be seen by looking at the time of the rising edge as a function of time.



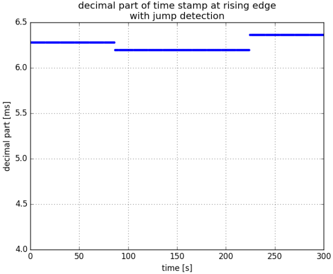
Without jump detection

Results of time stamping test with jump identification

The effect of the jump detection can be seen by looking at the time of the rising edge as a function of time.



Without jump detection



With jump detection

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:
  - mean: 1000.0004 ms

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:
  - mean: 1000.0004 ms
  - standard deviation: 0.0146 ms

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:
  - mean: 1000.0004 ms
  - standard deviation: 0.0146 ms
  - max deviation: 0.166 ms

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:
    - mean: 1000.0004 ms
    - standard deviation: 0.0146 ms
    - max deviation: 0.166 ms
    - min deviation: -0.167 ms

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:
  - mean: 1000.0004 ms
  - standard deviation: 0.0146 ms
  - max deviation: 0.166 ms
  - min deviation: -0.167 ms
- the time stamps were thus within one sampling period of the actual rising edge

Long term test to validate the time stamping method

The previous time stamping test was run continuously for more than 16 hours yielding the following results:

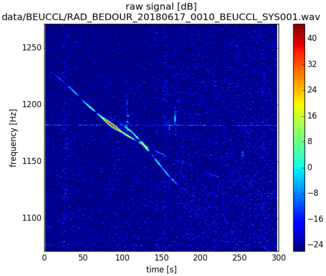
- 58759 rising edges out of 59082 pulses were used (99.45%)
- time between rising edges:
  - mean: 1000.0004 ms
  - standard deviation: 0.0146 ms
  - max deviation: 0.166 ms
  - min deviation: -0.167 ms
- the time stamps were thus within one sampling period of the actual rising edge
- for the 323 seconds where the time stamp was declared dubious, the time error was bounded by the measured discontinuity of a few ms

Simultaneous observation with BEUCCL

During nearly 5 days, the Funcube and the station at Uccle (BEUCCL with an ICOM receiver) observed the same signal to allow for operational comparisons (that have yet to be carried out).

Simultaneous observation with BEUCCL

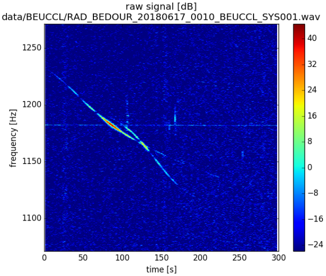
During nearly 5 days, the Funcube and the station at Uccle (BEUCCL with an ICOM receiver) observed the same signal to allow for operational comparisons (that have yet to be carried out).



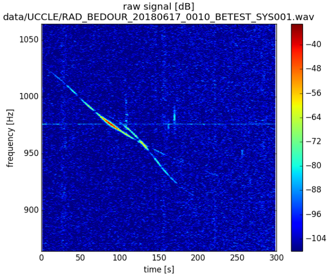
ICOM

Simultaneous observation with BEUCCL

During nearly 5 days, the Funcube and the station at Uccle (BEUCCL with an ICOM receiver) observed the same signal to allow for operational comparisons (that have yet to be carried out).



ICOM



Funcube

Conclusions for the Funcube

Conclusions for the Funcube

- good performance

Conclusions for the Funcube

- good performance
- loss of timing info (error of a few ms for less than 1% of the samples) may be acceptable (or not ?)

Conclusions for the Funcube

- good performance
- loss of timing info (error of a few ms for less than 1% of the samples) may be acceptable (or not ?)
- reliability remains to be demonstrated

Conclusions for the Funcube

- good performance
- loss of timing info (error of a few ms for less than 1% of the samples) may be acceptable (or not ?)
- reliability remains to be demonstrated

Conclusions for the Funcube

- good performance
- loss of timing info (error of a few ms for less than 1% of the samples) may be acceptable (or not ?)
- reliability remains to be demonstrated

The evaluation of the other front-end alternatives should continue (in particular with respect to the timing issue).

Airspy R2

Airspy R2

- procured one but, apparently, production has stopped

Airspy R2

- procured one but, apparently, production has stopped
- looked very promising with external reference and timestamped data

Airspy R2

- procured one but, apparently, production has stopped
- looked very promising with external reference and timestamped data
- too bad !

SDRplay RSP2

SDRplay RSP2

- procured one

SDRplay RSP2

- procured one
- started testing

SDRplay RSP2

- procured one
- started testing
- potentially much better frequency stability with an external reference

SDRplay RSP2

- procured one
- started testing
- potentially much better frequency stability with an external reference
- uses the same analog front-end as the Funcube: its performance should be similar

SDRplay RSP2

- procured one
- started testing
- potentially much better frequency stability with an external reference
- uses the same analog front-end as the Funcube: its performance should be similar
- does not use ALSA but a proprietary interface

SDRplay RSP2

- procured one
- started testing
- potentially much better frequency stability with an external reference
- uses the same analog front-end as the Funcube: its performance should be similar
- does not use ALSA but a proprietary interface

SDRplay RSP2

- procured one
- started testing
- potentially much better frequency stability with an external reference
- uses the same analog front-end as the Funcube: its performance should be similar
- does not use ALSA but a proprietary interface

This is a very serious contender. If it does not exhibit the sample loss of the Funcube, it may be the better option.

New receivers for the interferometer and the for RADAR

New receivers for the interferometer and the for RADAR

New receivers will also soon be needed for the interferometer and the RADAR.

New receivers for the interferometer and the for RADAR

New receivers will also soon be needed for the interferometer and the RADAR.

- It may be convenient to use the same model of receiver there as well.

New receivers for the interferometer and the for RADAR

New receivers will also soon be needed for the interferometer and the RADAR.

- It may be convenient to use the same model of receiver there as well.
- For these two applications, however, the LO must be phase-locked to a common reference. This rules out the Funcube of course.

The end