TN-07

NUSAR User Manual

Navigation User Software Receiver

	Name	Signature	Date
Written	Rui Sarnadas		
Verified	Teresa Ferreira		
QA Approval	Henrique Valente		
Approved	Teresa Ferreira		

Av. D. João II, Lote 1.17.02, Ed. Fernão de Magalhães, 7º, 1998-025 Lisboa, Portugal
Phone: +351 21 382 93 66, Fax : +351 21 386 64 93, Email: mail@skysoft.pt

(this page is intentionally left blank)

Document Change Log

Issue	Date	Description	Author
0.1	23-02-2009	Creation	Teresa Ferreira
0.2	20-05-2009	First issue	Agostinho Santos
1.0	25-05-2009	Completed for CDR data package	Rui Sarnadas
1.1	17-06-2009	Updated according to ESA_comments_for_TN7v1.1	Teresa Ferreira
1.2	24-09-2009	Updated according to software version at FR milestone: §4: folder description §5.1, §5.2, §5.3: now consistent with GUI updates	Rui Sarnadas
1.3	06-11-2009	Updated §5.1.3 to reflect GUI update Added C/N0 calibration table	Rui Sarnadas
1.4	25-11-2009	Added HDS description as Annex	Rui Sarnadas
1.5	02-12-2009	Updated according to ESA_comments_for_TN7v1.4_answered.doc. Added Annex B with recommendations.	Teresa Ferreira

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Scope

This document is a deliverable within WP6000, WP9000 of the NUSAR project described in [SoW].

1.2 Purpose

The purpose of this document is to guide the user throughout the usage and operations of the NUSAR platform.

1.3 Document Structure

Section 1	Defines the purpose, context and contents of this document
Section 2	Identifies the applicable and reference documents
Section 3	Provides a brief overview of the NUSAR software and platform
Section 4	Guides the user through software structure and installation requirements
Section 5	Describes the software operation
Section 6	Describes the API usage

Table 1-1: Document Structure

1.4 Acronyms List

API	Application Programming Interface
DLL	Dynamic-Link Library
GNSS	Global Navigation Satellite System
ICD	Interface Control Document
NUSAR	Navigation User Software Receiver
SoW	Statement Of Work
UM	User Manual

Table 1-2: Acronyms List

2 Documents

2.1 Applicable Documents

AD	Title	Document Identifier
[SoW]	NUSAR Statement of Work, NUSAR_SoW_1_2	NUSAR_SoW_1_2

Table 2-1: Applicable Documents

2.2 Reference Documents

RD	Title	Document Identifier (Internal)
[TN01]	NUSAR Technical Note 1 – Platform System Requirements	NUSAR-SKY-TN-01
[TN04]	NUSAR Technical Note 4 – NUSAR Interface Control Document	NUSAR-SKY-TN-04
[HDS]	Hardware Sub-Systems Data Sheets	NUSAR-SKY-HDS

Table 2-2: Reference Documents

3 NUSAR Description

3.1 Overview

The NUSAR is a flexible platform that processes GPS L1, Galileo BOC(1,1) and MBOC(6,1,1/11) signals transmitted in the E1/ L1 band, outputting code, phase and Doppler frequency information, among others. The NUSAR platform provides the means to design, evaluate, and test the performance of new tracking algorithms while processing the incoming signals.

Since the signal processing tasks are implemented in software, the platform provides the flexibility required to rapidly implement a given algorithm and assess the impact on performance. In addition, the NUSAR platform provides a set of APIs that ease up the implementation of the algorithms and the creation of inspection points. Finally, the NUSAR was developed following an "as fast as possible" approach; hence reaching real time for GPS and Galileo BOC processing.

The NUSAR general architecture is depicted in Figure 3-1, where the four main modules in the system are delimited by blue borders: the antenna, front-end, computer and software processing core.

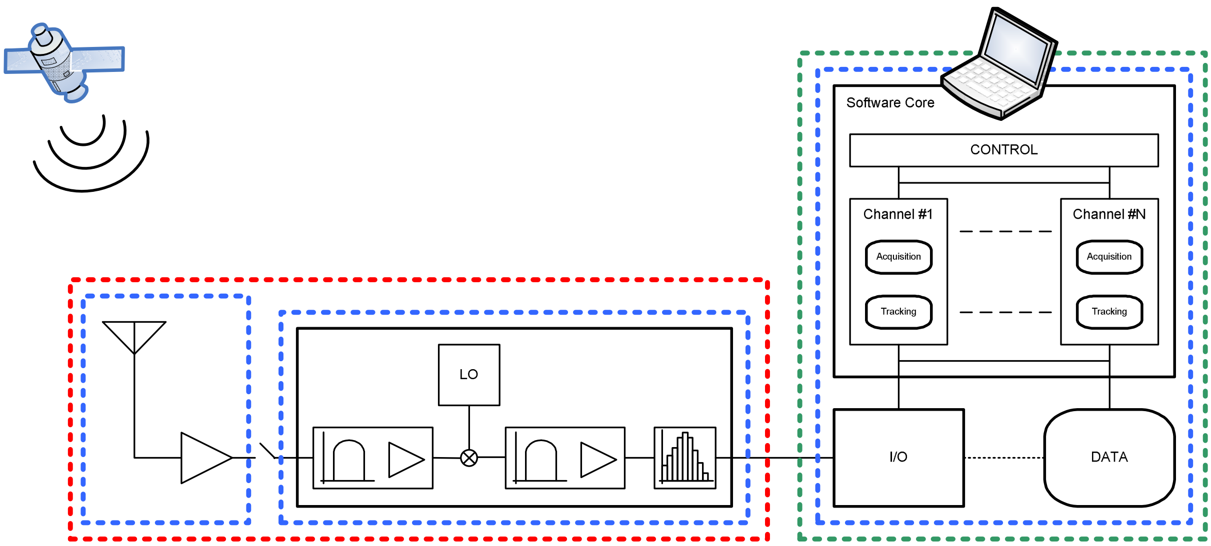

Figure 3-1: NUSAR platform architecture

The NUSAR platform is composed of three main physical parts: the antenna, the front-end, and the computer, all COTS components. In a stricter sense, the first two comprise the hardware for GNSS signal reception (red delimitation), and the computer will gather all the data and signal processing routines. The logical part comprises the software processing core (green delimitation). The characteristics of the antenna, front-end and laptop are defined in Appendix A.

3.2 Environment

The NUSAR software core was developed under Visual C++ 2008 Express Edition SP1, using the integrated Visual C++ compiler. Compilation, linker, and code generation flags have been fine tuned to match the requirements for a Win32 Windows™ application, running on Windows™ XP SP3.

The settings used in the platform's laptop computer are therefore consistent with the environment envisaged during the development stages, regarding software requirements, operating system, interfaces, and hardware specifications (like CPU speed/cores, and memory availability).

Note: It is also possible to run the NUSAR software core in other platforms, as long as they are compatible with the environment described. However, the requirements presented in [TN01] will not be guaranteed, since they are derived for the NUSAR platform only. Nevertheless, for that purpose, the user must port the entire NUSAR folder structure and files to the new environment, as described in Section 4.

4 NUSAR Structure and Installation

The NUSAR platform includes the laptop with the software executable code. In order to start using the tool, the following procedures need to be followed:

- Connect the laptop (and optionally, the front-end and antenna components).
- Double click on the NUSAR executable.

In the NUSAR platform environment, the program is set to run without the need for installation and system configuration. In any case, if the user needs to rebuild the folder structure, or re-install the NUSAR software (like running on a different PC – see Section 3.2 for details), the necessary folder structure is shown in Figure 4-1:

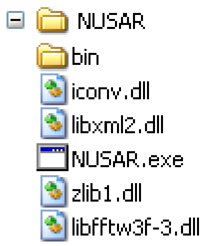

Figure 4-1: NUSAR folder structure

This structure does not require an installation routine, since it is sufficient to run the executable within the main program folder. The files included are described in Table 4-1.

File / Folder	Description	Optional / Needed / Created
NUSAR.exe	This is the main program executable for the NUSAR software.	Needed
<*filename*>.dll	System and third-party libraries, used by NUSAR (XML parser and FFTW™ library).	Needed
bin (folder)	This folder is the user repository for NUSAR operation, and gathers the input files needed, such as PRN codes or external libraries. It is also the default output folder for logs and data files.	Needed / Created
DiscCodeLoop.dll	User library to integrate in NUSAR (see API for details).	Optional
NUSAR_LOG...	Example of a log file generated (online and offline modes).	Created
NUSAR_OUT...	Example of a data output file (data acquisition mode).	Created
prncodes	PRN codes used in NUSAR.	Needed
*.xml	Session and Front-End settings saved files	Optional
fftw_wisdom	FFTW wisdom generated for the NUSAR platform	Optional

Table 4-1: NUSAR file and folder description

Although the root directory must have at least the presented files for NUSAR to operate, the “bin” folder needs only a valid “prncodes” file. This file is a simple text file with the PRN codes to use in the GNSS signal processing. Table 4-2 shows the structure of the file.

```
#32#GPS L1 C/A
11001000001110010100100111110010100... <1023 chips>
11100100000111000000111110100110010 1... <1023 chips>
11110010001110001000010100100111 01... <1023 chips>
...
(32 codes)
...
#50#Galileo E1-B Primary
11110101110101110001000000001001100... <4092 chips>
10010110101110000101011010100110 00... <4092 chips>
11100101011111011110000110011010 00... <4092 chips>
...
(50 codes)
...
#50#Galileo E1-C Primary
10110011100100110100000001100101000... <4092 chips>
10100110010011111001010010111011 01... <4092 chips>
11111010001111101101001011101100 0... <4092 chips>
...
<50 codes>
...
#01#Galileo E1-C Secondary
0011100000001010110110010
#EOF
```

Table 4-2: PRN code file structure

Note that the first value of each header line is the number of subsequent codes, i.e., in the example above, there should be 32 GPS codes, 50 Galileo E1-B codes, 50 Galileo E1-C codes, and 1 Galileo E1-C secondary code (one per line, with no empty lines).

As for the data output file and log files, their structures are defined in the ICD in [TN04], and are not repeated here for simplicity.

5 NUSAR Operation

The NUSAR platform can be operated in three different modes: online, offline and data acquisition. The use of the NUSAR interface to perform such actions is described in the following sections. For details on the values, parameters, and files involved, please refer to the NUSAR ICD in [TN04].

5.1 User Interface

5.1.1 Main window

The main window (GUI) of the NUSAR application is depicted in the following figure.

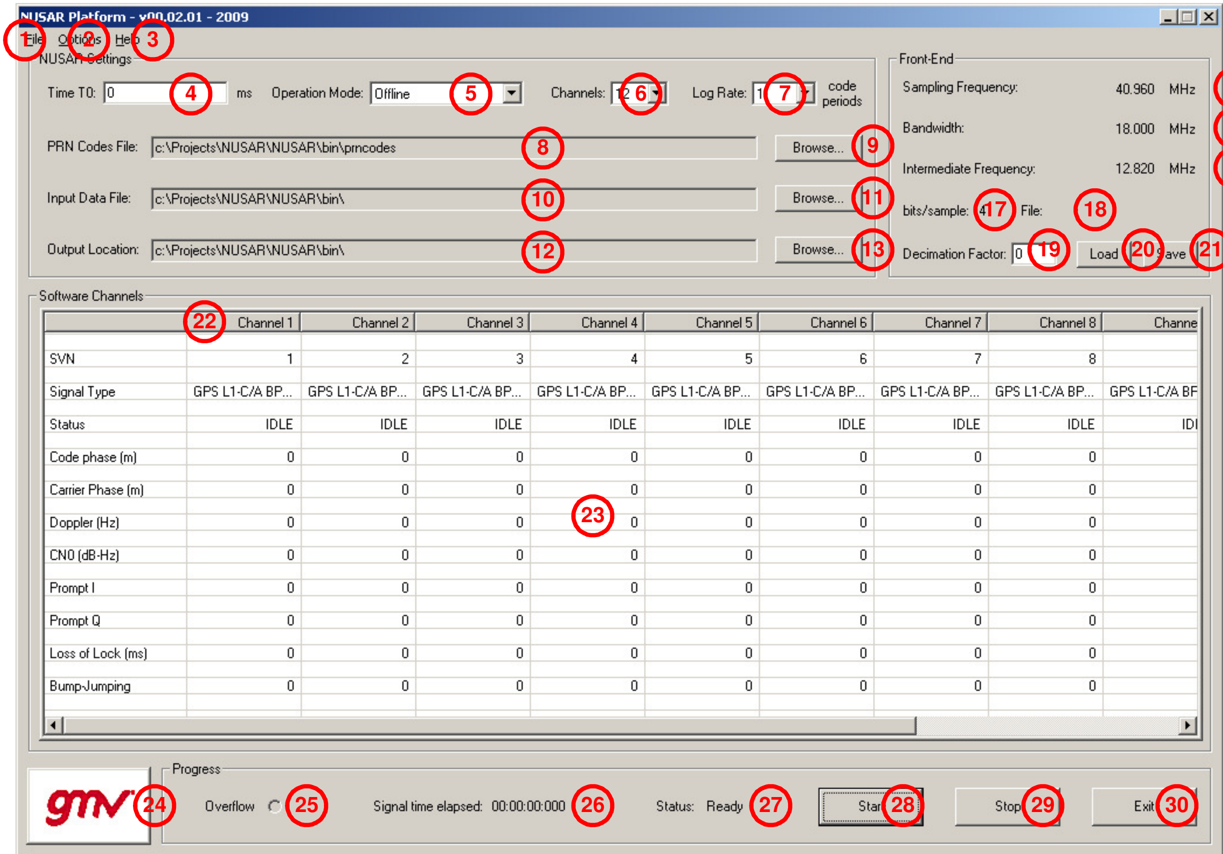

Figure 5-1: NUSAR main window

1. "File" menu: Save/load the current NUSAR settings, or Exit the program.

   a. "Save Session": saves the current NUSAR settings to a user-defined file.
   b. "Load Session": loads a previously saved NUSAR session and settings.

2. "Options" menu: Display the "Channel Configuration" dialog, for all channels.
   a. "Channel Configuration": User-configurable channel parameters shown in Figure 5-2.
3. "Help" menu: Display the "About" dialog.
   a. "About": Shows the development information regarding the NUSAR platform, as in Figure 5-4.
4. Absolute value for signal time T_0.
5. Operation mode selection: Online, Offline, or Data Acquisition.
6. Number of channels to use for signal processing.
7. Log rate option: options include no logging, and different rates to log the data (relative to the channel's code period): 1, 25 or 100.
8. Current PRN codes file to use.
9. Browse for a PRN codes file.
10. Current input data file to use.
11. Browse for an input data file.
12. Output files destination path. Please note that the file name is "NUSAR_OUT_<TIMESTAMP>.nus" for data acquisition mode and "NUSAR_LOG_<TIMESTAMP>.txt" for the other ones. These output files are fully described in [TN04].
13. Browse for an output file destination path.
14. Front-End sampling frequency.
15. Front-End bandwidth.
16. Front-End intermediate frequency.
17. Number of bits per sample in Front-End.
18. Current Front-End settings file used (or empty, if the defaults are used), as in [TN04]
19. Decimator: decimation factor for the Front-End sampling frequency, and used to store samples internally in NUSAR (if 0, use NUSAR internal values).

20. Load Front-End settings file, defined in [TN04].

21. Save Front-End settings file, defined in [TN04].

22. Column header: If clicked, displays the "Channel Configuration" dialog for that specific channel (and channel selection is disabled in the dialog).

23. NUSAR output pane: Timely display and refresh of NUSAR processing results.

24. Developer logo: GMV.

25. Buffer overflow indication (online and data acquisition modes only).

26. Signal time processed since START (relative to $T_{START} = 0$).

27. NUSAR status information: *Ready*, *Initializing*, *Running, Cleaning up* and *Stopped*.

28. START button, to start the envisaged process.

29. STOP button, to stop the running process.

30. EXIT button, to quit the NUSAR application.

5.1.2 Channel Configuration dialog (per channel)

The "Channel Configuration" dialog is presented in Figure 5-2.

1. Channel to configure: enabled only when the dialog is displayed from the "Options" menu.

2. Satellite (PRN code) to search for in the channel: 1 to 32 for GPS signals, 1 to 50 for Galileo signals.

3. Signal type to process: GPS L1 C/A BPSK(1), Galileo E1-B/C BOC(1,1), or Galileo E1-B/C CBOC(6,1,1/11).

4. SVN Fixed: If checked, keep trying to find and track the assigned satellite (PRN) only.

5. Maximum Doppler frequency: to acquire signals between ±MAX kHz.

6. Maximum number of non-coherent integrations, before declaring that the signal is not present during acquisition.

7. Probability of false alarm: define the probability of false alarm, P_{fa}, to use in the acquisition threshold.

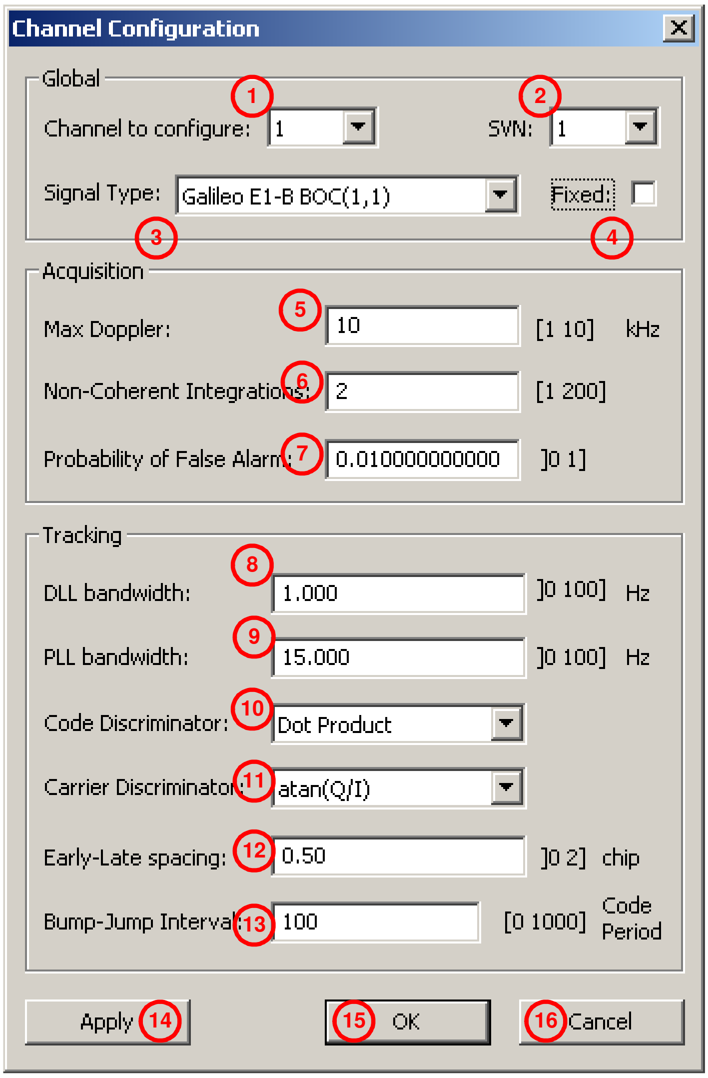

Figure 5-2: "Channel Configuration" dialog

8. Delay-Locked Loop Bandwidth.

9. Phase-Locked Loop Bandwidth.

10. Code discriminator to use: Dot Product (default), or user defined (in *.dll file).

11. Carrier discriminator to use: atan(Q/I).

12. Early-Late chip spacing.

13. Bump-Jumping interval, in code periods (0 means "do not use bump-jumping").

14. Validate and apply the current settings to the current channel, and keep the dialog open.

15. Validate and apply the current settings to the current channel, and close the dialog.

16. Close the dialog without applying the changes.

5.1.3 Channel Configuration dialog (all channels)

Some parameters can be set for all channels at once (by selecting the values to update and setting the new value) in the "Configure All Channels" dialog, as presented in the figure below.

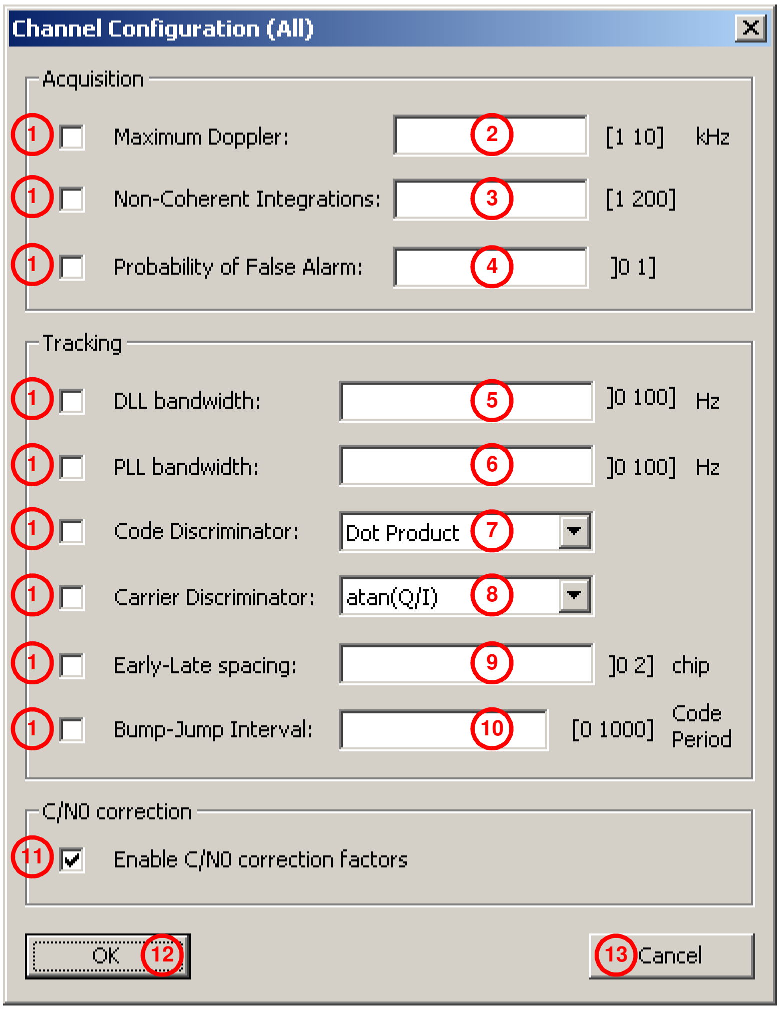

Figure 5-3: "Configure All Channels" dialog

1. Checkboxes to select the values to update / change.

2. Maximum Doppler frequency: to acquire signals between ±MAX kHz.

3. Maximum number of non-coherent integrations, before declaring that the signal is not present during acquisition.

4. Probability of false alarm: define the probability of false alarm, P_{fa}, to use in the acquisition threshold.

5. Delay-Locked Loop Bandwidth.

6. Phase-Locked Loop Bandwidth.

7. Code discriminator to use: Dot Product (default), or user defined (in *.dll file).

8. Carrier discriminator to use: atan(Q/I).

9. Early-Late chip spacing.

10. Bump-Jumping interval, in code periods (0 means "do not use bump-jumping").

11. Enable C/N0 correction factor, due to decimation losses. This value is dependent on the Front-End bandwidth and sample frequency used, and typical values are summarized in Table 5-1.

Decimation factor	C/N0 losses - added as calibration [dB-Hz]
1	0
2	2,45
5	6,43
10	9,44
0	applied as per channel signal type

Table 5-1: Typical C/N0 calibration factors

12. Validate and apply the current settings to the current channel, and close the dialog.

13. Close the dialog without applying the changes.

5.1.4 About NUSAR

The NUSAR “About” dialog is depicted in Figure 5-4.

Figure 5-4: “About NUSAR” dialog

5.2 Online mode

In online mode, the NUSAR software listens to data coming from the Front-End hardware into the laptop. Therefore, the main window disables the “Input Data File” browser, since it is not used in this mode. The “PRN code File” and “Output Location” settings can be set to another location specified by the user, and the log file rate can also be configured. For further details on the values involved, please refer to the NUSAR ICD in [TN04].

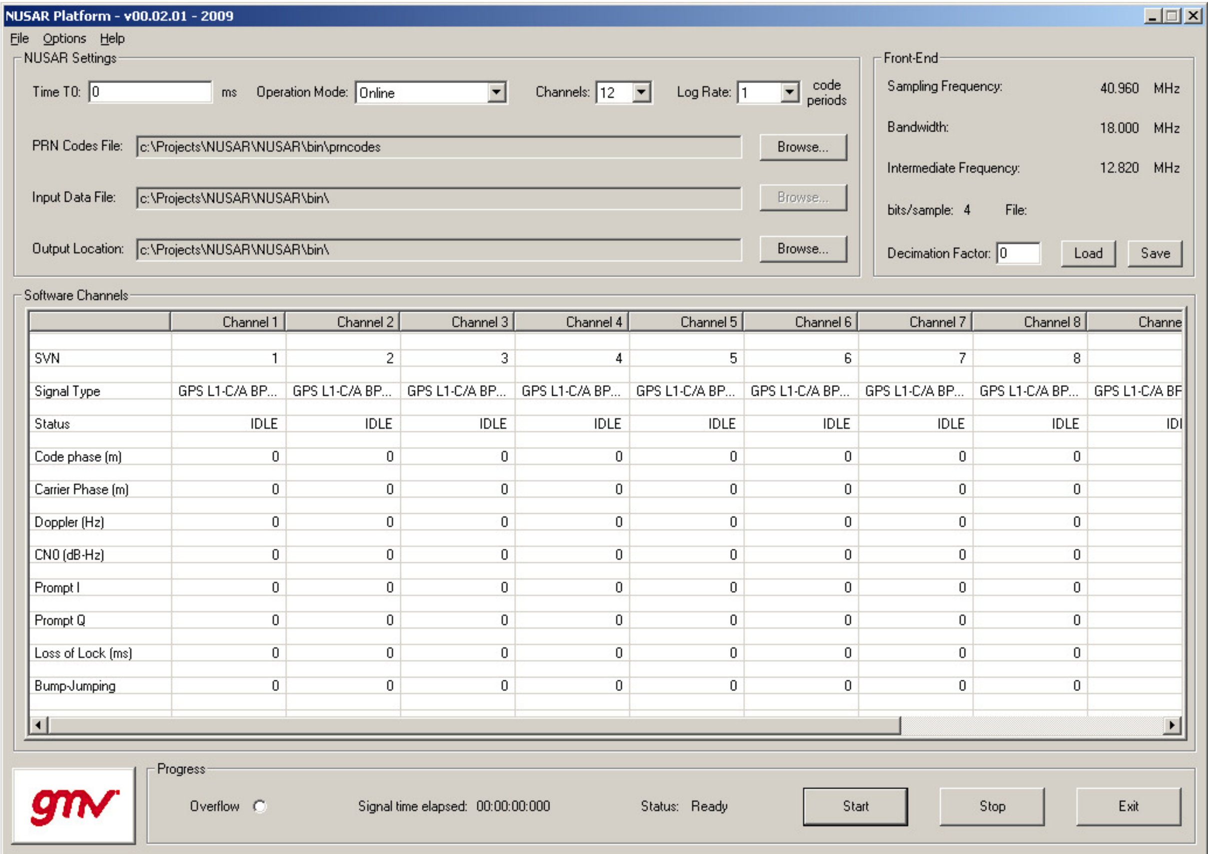

Figure 5-5: Main window for “Online” mode

5.3 Offline mode

In offline mode, data is processed from a data file in *.nus format, or other compliant format as specified in [TN04]. Hence, the file can be previously acquired in NUSAR's "Data Acquisition" mode, or otherwise generated in compliance to the expected format. To successfully perform the signal processing tasks configured, the user should load the Front-End settings used when acquiring the signal.

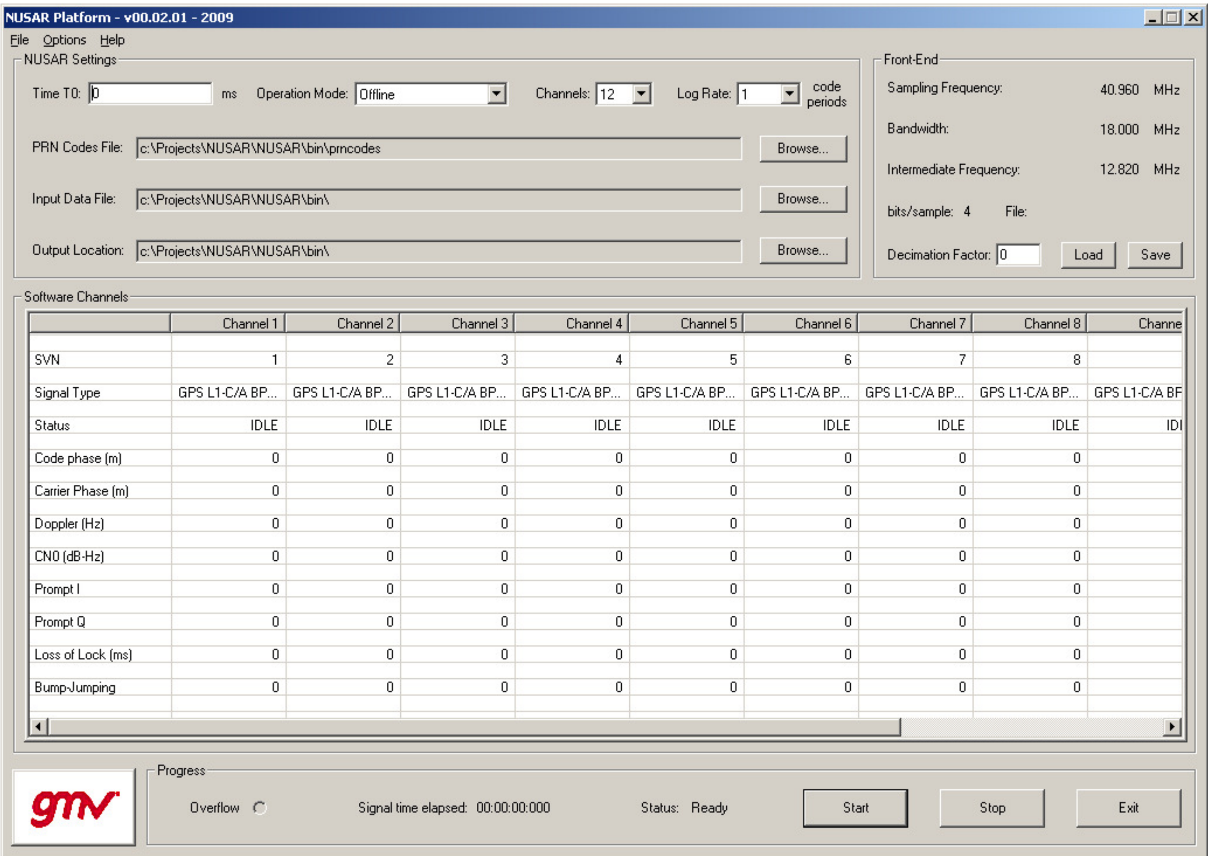

Figure 5-6: Main window for "Offline" mode

As a result of offline processing, the main window disables the "Overflow" information, since the data is in a file and will not be lost by buffer overflows. The remaining settings can be set to the user's will, and the log file can be saved to disk, or disabled. For further details on the values involved, please refer to the NUSAR ICD in [TN04].

5.4 Data Acquisition mode

When data acquisition mode is selected, the output display is disabled, and so are the T0 time box, channel selection, log file generation, and PRN codes and Input files. These settings are not used, since this mode generates an output file with a "copy" of the data received from the Front-End hardware, with no signal processing. Nevertheless, the Front-End configurations can be changed to match a specific hardware used. For further details on the values involved, please refer to the NUSAR ICD in [TN04].

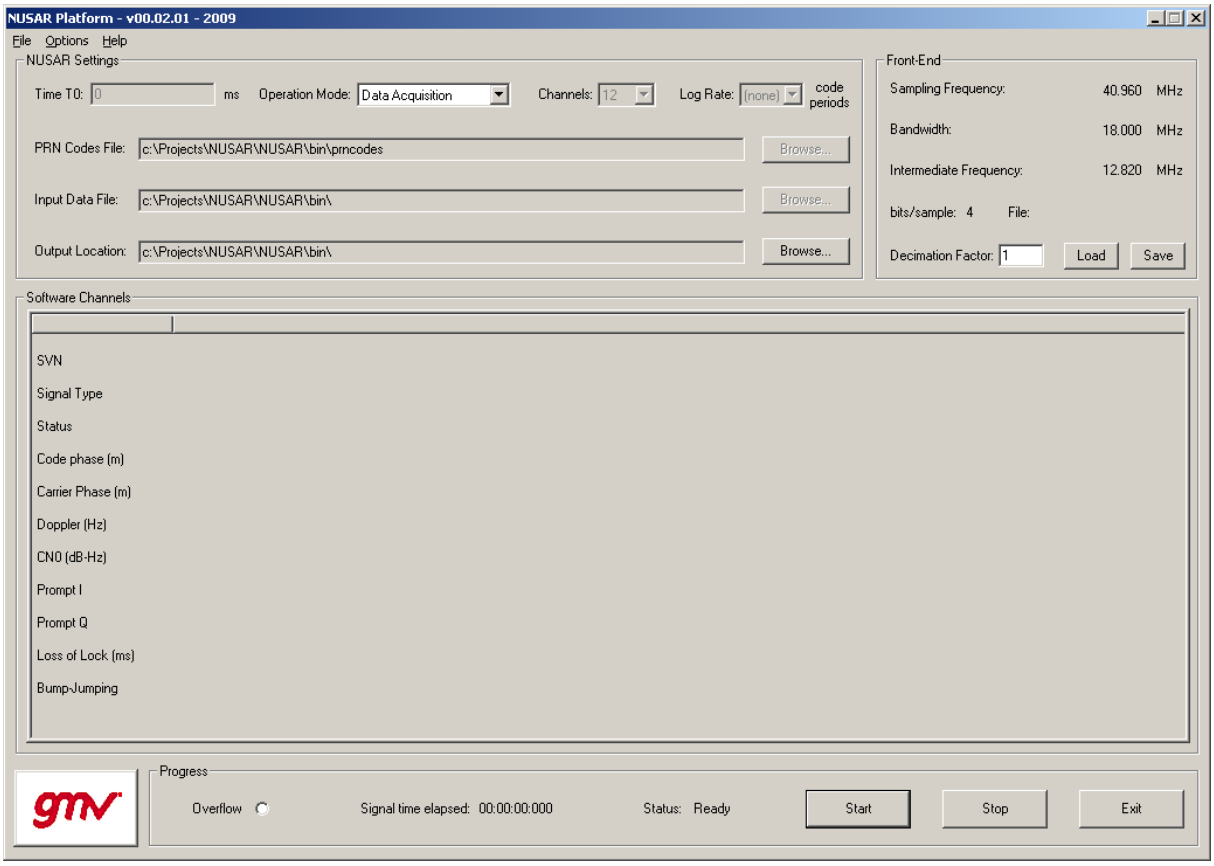

Figure 5-7: Main window for "Data Acquisition" mode

6 NUSAR API Description

NUSAR allows users to configure their own code discriminator function, put it in a Windows™ "dll" (Dynamic-Link Library) file, and use it within the signal processing tasks. To enable this, NUSAR has an API to allow users to generate and use different algorithms for specific tasks within the NUSAR software. The following sections show an applicable example of the NUSAR API, including implementation details and API sources.

6.1 DLL file generation

The NUSAR API allows for implementation and usage of a Code Discriminator function (like the Early-minus-Late discriminator, for instance) in a "dll" file, containing the run-time callable function. This external function must be compiled against the "NUSARUserLibrary" header, which includes the necessary definitions and prototypes for correct integration in NUSAR. This header file is presented in Table 6-1.

After setting up a project for generating a Windows™ "dll", and including the "NUSARUserLibrary" header, the remaining step needed is to implement the function. In this case, the function declaration in the header is called "DiscCodeLoop", so the user needs only to code this function accordingly. Figure 6-1 represents a simple example of the Code Discriminator "dll" generation.

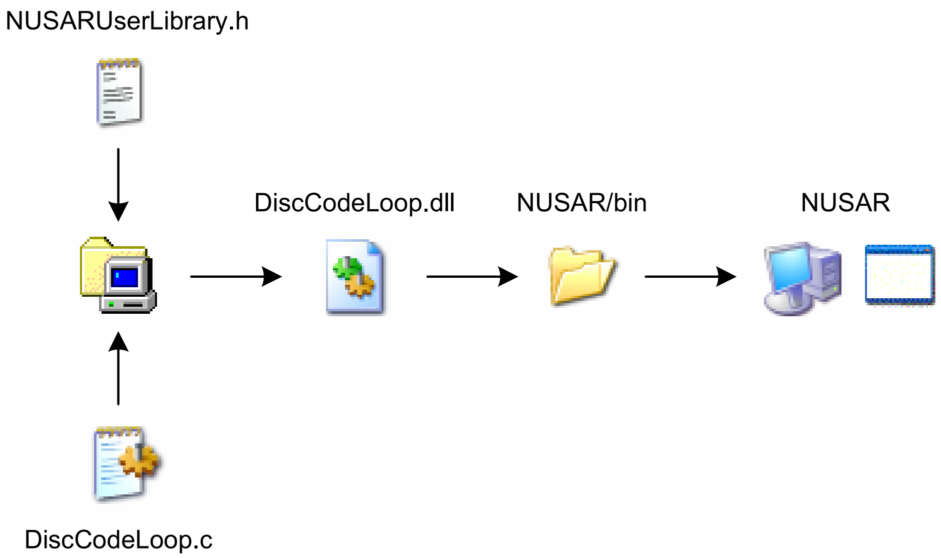

Figure 6-1: Sequence for NUSAR "dll" generation and usage

After coding and testing, the "dll" can be generated, keeping in mind that the name of the file should be the same as specified in the macros present in "NUSARUserLibrary", as shown in the following tables.

```
#ifndef NUSAR_DLL_H
#define NUSAR_DLL_H

#include <windows.h>
#include <math.h>

/*-------------------------------------------------------------------------*/
/* NUSAR-specific macros */
/*-------------------------------------------------------------------------*/
#define NUS_FREC_OK 0                   /* OK (no error) */
#define NUS_FREC_NOK 1                  /* Not OK (general error) */
#define NUS_FREC_BAD_DATA 3             /* bad data in function */


/*-------------------------------------------------------------------------*/
/* Data type used */
/*-------------------------------------------------------------------------*/
typedef float nusFFT_t;
typedef nusFFT_t nusComplex_t[2];


/*-------------------------------------------------------------------------*/
/* Function headers available for NUSAR function integration */
/*-------------------------------------------------------------------------*/
/*--------------------------------------------------------------------------*
|
| FUNCTION NAME: DiscCodeLoop
|
| DESCRIPTION:
| This function performs the Code Discriminator (Early-minus-Late).
|
| AUTHOR: Teresa Ferreira
|
| PARAMETERS:    In Out Description
| sumP_           x     Accumulation of prompt correlators.
| sumEmL_         x     Accumulation of E-L correlators.
| discOutput_        x  Discriminator output.
|
| RETURN VALUE:
| NUS_FREC_OK       - Success.
| NUS_FREC_NOK      - General Error.
| NUS_FREC_BAD_DATA - Bad/illformed input data.
|
* ---------------------------------------------------------------------- */

#define NUS_SET_CODISC_DLL ".\\bin\\DiscCodeLoop.dll"
#define NUS_SET_CODISC_PROC "DiscCodeLoop"

/* Definition of Code Discriminator DLL function prototype */
typedef int (__cdecl *CODISCPROC)(nusComplex_t, nusComplex_t, double*);

/* Code Loop Discriminator - Early-Late */
__declspec (dllexport) int DiscCodeLoop(nusComplex_t sumP_,
                                        nusComplex_t sumEmL_,
                                        double *discOutput_);
```

```
/*----------------------------------------------------------------------*/

#endif
```

Table 6-1: NUSAR API Header

```
#include "NUSARUserLibrary.h"

/*-----------------------------------------------------------------------*
|
| FUNCTION NAME: DiscCodeLoop
|
| DESCRIPTION:
| This function performs the Code Discriminator (Early-minus-Late).
|
| AUTHOR: Teresa Ferreira
|
| PARAMETERS:    In Out Description
| sumP_           x     Accumulation of prompt correlators.
| sumEmL_         x     Accumulation of E-L correlators.
| discOutput_        x  Discriminator output.
|
| RETURN VALUE:
| NUS_FREC_OK        - Success.
| NUS_FREC_NOK       - General Error.
| NUS_FREC_BAD_DATA - Bad/illformed input data.
|
* ---------------------------------------------------------------------- */

__declspec (dllexport) int DiscCodeLoop(nusComplex_t sumP_,
                                        nusComplex_t sumEmL_,
                                        double *discOutput_)
{
   ...
   <function body>
   ...

   return NUS_FREC_OK;
}
```

Table 6-2: Code Discriminator DLL

6.2 External “Code Discriminator” usage

Enabling the use of the external DLL is simple: in the “Channel Configuration” dialog, the user can select the “*User Defined*” option in the “Code Discriminator” field, as shown in Figure 6-2.

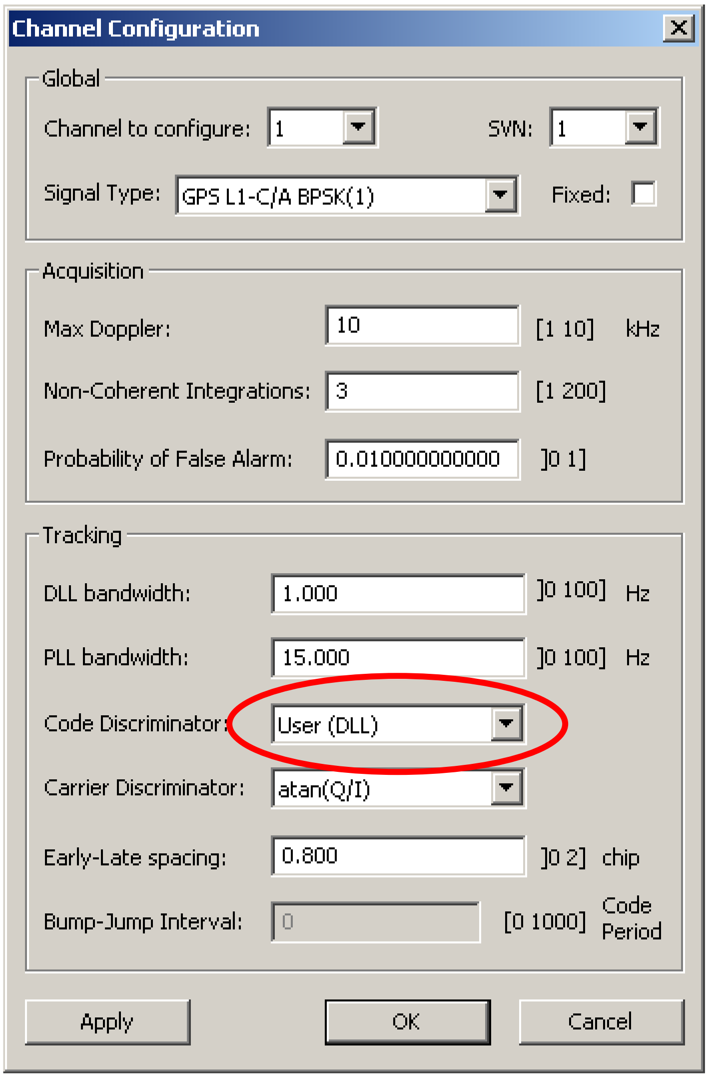

Figure 6-2: External Code Discriminator usage

Appendix A NUSAR Data Sheets

The following hardware data sheets are described in version 1.1 of the [HDS] (25-11-2009).

A.1 Antenna

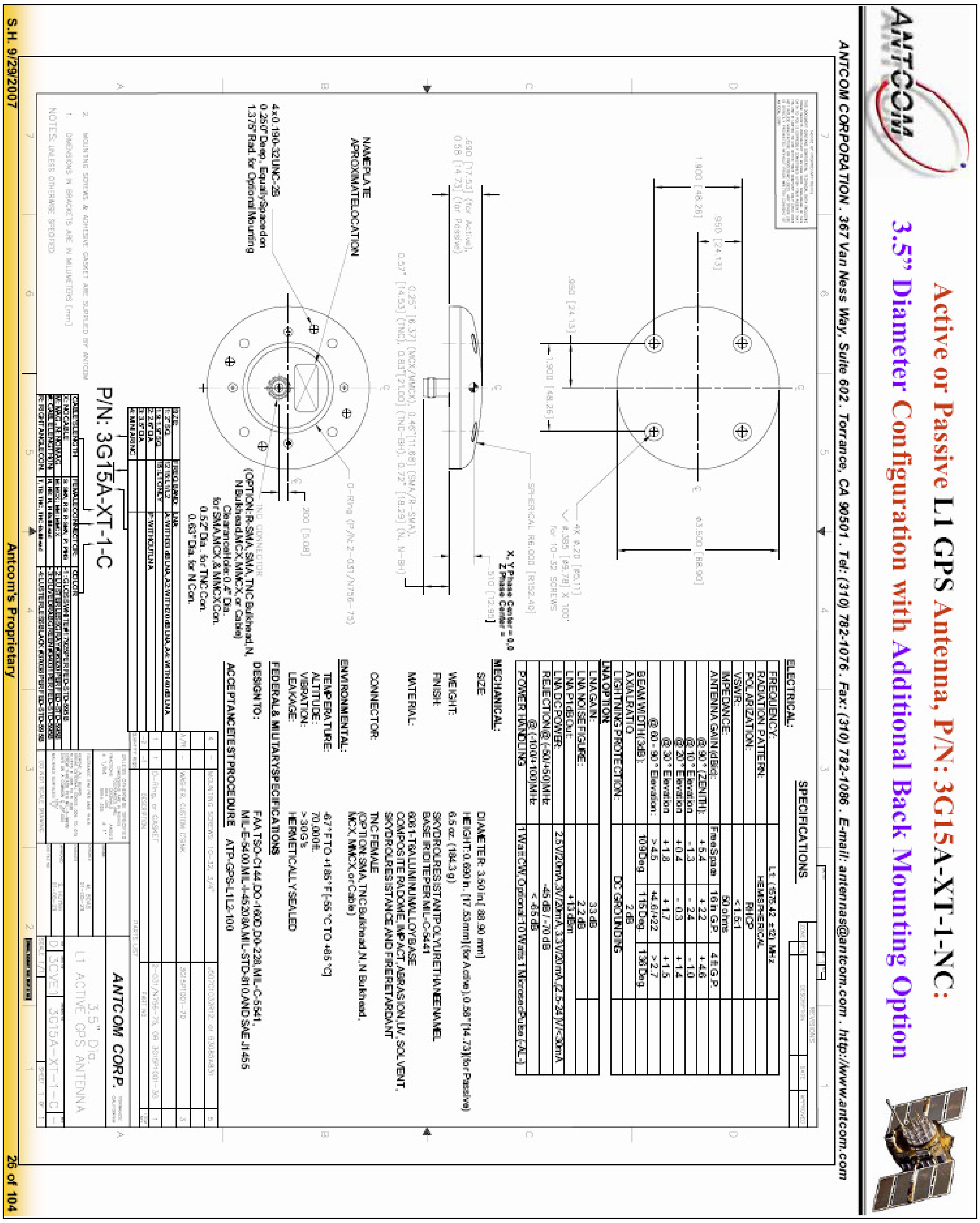

Figure A-1: Antenna Data Sheet

A.2 Front-End

Subject:	1 set of GPS frontend **L1** with USB
Features:	- Center frequency: 1575.42 MHz
	- Antenna connector: TNC
	- Support of active antenna (4.6 VDC)
	- Bandwidth (BW): 18 MHz
	- Intermediate frequency (IF): 13 MHZ
	- Sample rate: 40,96 MHz
	- Resolution: 2 or 4 bit
	- High speed USB v2.0 digital data interface
	- USB driver for Windows XP
	- External clock input and output
	- Front end powered via USB
	- Casing included
	- Tool for USB data recording

Figure A-2: Front-End Data Sheet

For the Fraunhofer Front-End, power ratings and specifications are:

- RF input – input impedance is 50Ohm, RF power level no more than -50dBm (to prevent damage of the input). Please consider that saturation of the AD-converter can still be reached even with the lowest gain setting.

- Antenna – 4,7V power supply, maximum current: 40mA.

- External 10MHz clock input – The peak-to-peak amplitude should be in the range [0...3V]. For best performance, use a rectangle signal. If a sine wave is applied, a DC component of 1.6V should be present, with 1.2V peak-to-peak amplitude.

A.3 Laptop Computer

The laptop computer was chosen as the best compromise between portability, performance, and capabilities to successfully run the NUSAR software.

- Name: CLEVO M570TU
- CPU: Intel Core 2 QUAD Q9000 2.0GHz (1066MHz) 6MB - Montevina
- Chipset: Intel PM45 + ICH9M
- HDD: 320GB 7200rpm SATA II 16MB
- Screen: 17,1" WUXGA+ (1920x1200) GT TFT LCD
- Graphics card: nVIDIA® GeForce® 260M GTX 1GB GDDR3
    - Turbocache up to 2813MB MXM Type-III Modular
    - DirectX® 10
    - 64-bit wide per channel
- Memory: 4GB DDR3 (1066MHz)
- ODD: DVD ±RW/±R
- Security: Finger print reader
- Battery: 8-cell smart Lithium-Ion 4400mAh
- Modem: 56K FAX / modem
- Network: 10/100/1000MB Base-TX Ethernet LAN
- Wireless: Intel® WiFi Link 5300 3x3 802.11AGN
- Bluetooth 2.0
- 4 x USB 2.0 ports
- 1 x eSATA
- 1 x HDMI output (supports single channel) with audio
- 1 x DVI output
- 1 x Headphone jack
- 1 x Microphone jack
- 1 x S/PDIF Output jack
- 1 x COM port
- 1 x RJ-45 jack
- 1 x RJ-11 jack
- 1 x IEEE1394a port
- 1 x RF-in jack
- 1 x DC-In jack
- 1 x CIR for TV remote controller
- 1 x ExpressCard/54(34) slot
- 7-in-1 Card Reader (MMC/RSMMC/MS/MS Pro/MS Duo/SD/Mini-SD)
- Sound Card:
    - High Definition Audio
    - 3D stereo enhanced sound system
    - Sound-Blaster PRO™ compatible
    - S/PDIF digital output
    - Built-in microphone
    - Built-in two speakers

  - o Built-in Sub-woofer
  - o SRS WOW Surround Sound Technology inside
- Dimensions: 397 (Width) x 284 (Depth) x 22~44 (Height) mm
- Weight: 4 Kg (with 8-cell battery)

Figure A-3: Laptop Computer Images

Appendix B Usage Recommendations

The performance of the NUSAR platform (namely processing speed) depends both on the hardware environment and the software configurations.

As far as the hardware environment is concerned, the processing speed depends (among other factors) on the access to the laptop's hard disk. As a consequence, it is recommended to keep it as free and de-fragmented as possible.

Regarding the software environment, the configurations settings used for testing the real time requirements during acceptance tests are provided for illustrative purposes:

- SignalType BOC(1,1)
  - Decimation factor: 10
  - MaxDoppler="5000"
  - NonCohInt="2"
  - Pfa="1.000000000000e-003"
  - DLLBandwith="1.000"
  - PLLBandwith="15.000"
  - CodeDiscriminator="Dot Product"
  - CarrierDiscriminator="atan(Q/I) "
  - Early-LateSpacing="0.500"
  - Bump_Int="100"

- SignalType GPS
  - Decimation factor: 10
  - MaxDoppler="5000"
  - NonCohInt="70"
  - Pfa="1.000000000000e-002"
  - DLLBandwith="1.000"
  - PLLBandwith="15.000"
  - CodeDiscriminator="Dot Product"
  - CarrierDiscriminator="atan(Q/I) "
  - Early-LateSpacing="0.800"
  - Bump_Int="0"