Setup for Measurement and Characterization of Cryogenic Low-Noise Receivers

Citation for the original published paper (version of record):
Strandberg, M., Sundin, E., Helldner, L. et al (2024). Setup for Measurement and Characterization of Cryogenic Low-Noise Receivers. 2024 4th URSI Atlantic Radio Science Meeting, AT-RASC 2024. http://dx.doi.org/10.46620/URSIATRASC24/TLCH9750

4th URSI AT-RASC, Gran Canaria, 19-24 May 2024

Setup for Measurement and Characterization of Cryogenic Low-Noise Receivers

Magnus Strandberg*, Erik Sundin, Leif Helldner, Denis Meledin, Igor Lapkin, Sven-Erik Ferm, Mathias Fredrixon, Victor Belitsky and Vincent Desmaris

Group for Advanced Receiver Development, Department of Space, Earth and Environmental Sciences, Chalmers University of Technology, Gothenburg, SE-41296, Sweden

Abstract

In mm-wave and THz radio astronomy receivers, key components such as mixers, hybrids etc., needs to be fully characterized individually to ensure they fulfill requirement and specifications of various types. In this paper, we present details of a new measurement setup, built in the labs of the Group for Advanced Receiver Development (GARD) and a new underlaying software back-end for measuring, characterization and monitor any of the receiver's components. This software, called GMS – GARD Measurement System, does allow several frontend graphical user interfaces (GUI) being written in different programming languages to connect and interact with the measurement setup simultaneously.

1. Introduction

The GARD group is involved in building key components and full receiver front-ends for mm-wave and THz radio astronomy observatories such as ALMA and APEX. Consequently, there is a clear need to characterize the components of the receiver (and receiver itself) under cryogenic temperatures by verification of their performance through various measurement methods with different types of measurement instruments. Addressing this challenge, a new measurement setup was designed and built including a new 4 K cryostat employing 3-stage cryocooler, and featuring an automated Y-factor measurement system with hot and cold load enclosed in a nitrogen atmosphere. The setup includes optics to inject external LO / pilot signals and to conduct measurements of DSB mixers with external LO injection or of the sideband separation mixers when a pilot signal is used.
In order to automate various measurement routines, we have implemented dedicated software for measuring the device under test (DUTs) and providing data for the users. A back-end hardware abstraction layer written in Qt/C++ [1], which is running in the background of the measurement setup, is a convenient technique for having a large range of front-end GUIs communicating and obtaining data from measurement equipment. In this paper, we provide details on the measurement set for characterization of cryogenic heterodyne receivers for mm and submm wavelengths and the manuscript is divided into two parts, where the first part describes the measurements setup hardware and the other part provides a comprehensive insight into motivations, design and solutions for programming an underlying software for a measurement setup/station.

2. Measurement Setup

The measurement system (Figure. 1) is built up around a custom-made cryostat, employing a three-stage 4 K cryo-cooler. The cryostat cryogenic support structure is designed such that the change in position of the 4 K plate due to the thermal contraction is minimized. The cold head is mechanically suspended and vibrationally isolated to minimize vibrations that would affect the stability measurements.
The cryostat has in total nine ports that can accommodate waveguide, optical or cable feedthroughs ports, while of which four ports that are primary intended for waveguide and/or cable feedthroughs. The current setup utilizes one optical port for (RF/LO) injection, two ports for IF x 8 coaxial cables (for up to 20 GHz), and two ports for direct LO injection through waveguide vacuum feedthroughs. Several vacuum service ports are located underneath the cryostat chamber, including space for a dedicated remote-controlled turbo vacuum pump and additional vacuum components such as gauges, overpressure and venting valves.
The cryostat DC feedthrough is compatible with ALMA Front End DC bias module that is capable to DC bias up to two dual sideband (2SB) mixers, four 3-stage LNAs four magnetic coils and monitor four temperature sensors.
The cryostat is equipped with additional temperature sensors, and heaters on all three temperature stages for temperature stabilization during tests, and running accelerated warming up by an external temperature controller.

The Quasi-optical path has been designed to match the ALMA cartridge layout similar to APEX SEPIA345 receiver [2], having two cold focusing mirrors inside the cryostat.
Further along the optical path, a replaceable optical cryostat window with anti-reflection surfaces for respective frequency band are used before the beam reaches the facility load box outside the cryostat.

One important feature of the measurement setup is the enclosed facility load box that is filled with dry nitrogen via an adjustable valve for minimizing water vapor content in the optical path, and condensation on the RF window. A liquid nitrogen bath is located at the top of the facility load box and contains a cold load submerged in liquid nitrogen. For the measurement of noise temperature, a stepper motor together with a mirror is used for switching between the cold load and the hot load. The load's surface is covered by a mixture of black Stycast (binder) and SiC grains (filler) [3]. This stepper motor using a hall sensor and a magnet to find its reference position. The motor positions of the hot and cold ones have been investigated and tested to achieve best transfer path of signal.

The LO system consists of a reference signal generator (a commercial synthesizer), and feed into the ALMA Band 6 respective ALMA Band 7 active multiplier chain [4], developed by NRAO, with the output power controllable via power amplifier which in turn is remotely controlled via the warm cartridge assembly interface.

The LO assembly was equipped with a diagonal horn, whose beam is injected through a thin Zitex window in the calibration box and further focused by a 3D-printed microwave lens before entering the beam splitter made of 30 micrometer thick Mylar where the LO/pilot signal gets injected to the mixer quasi-optically.

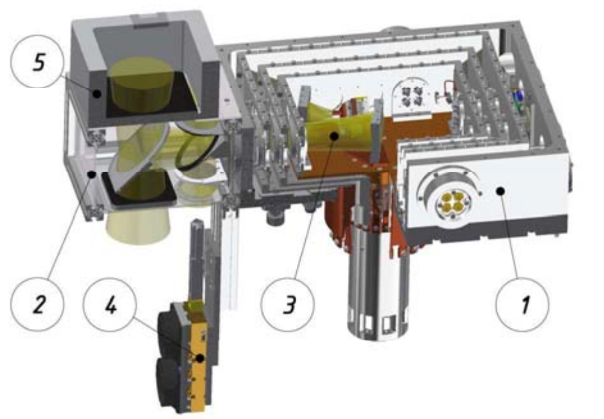

Figure 1. Section drawing of the new measurement setup which provide a view of the cryostat outer shell (1), the facility load box (2), the visualized optical beams (3), the LO system (4) and the LN2 bath (5), spotlighting the details of the arrangement.

Given setup allows to perform standard Y-factor noise temperature measurements of double sideband (DSB) mixers when the 30 µm thick Mylar beam splitter is used for the LO injection. For 2SB mixer measurements, the LO injection is performed via waveguide feedthrough of the cryostat port while the pilot signal for sideband rejection characterization is injected via the beam splitter, in this case of 5 µm thick. The routines for such measurements are programed as GUIs using underling procedures for, e.g., sideband characterization [5]. The GUIs allow control and optimize current of the magnetic coils, DC bias of SIS junction in mixers, control and use of actual room temperature load via dedicated temperature sensor in the Y-factor measurements. The optics and possibility for easy exchangeable vacuum RF windows, LO/pilot signal sources allow to perform measurements starting from approximately 150 GHz and up. The setup with its possibility to provide multiple coaxial links to DUTs at 4 K allows, e.g., S-parameter measurements using a 4-ports Vector Network Analyzer to characterize LNAs and hybrids at cryogenic temperatures.

3. GMS-software

The dedicated new software was developed for performing control and measurement using the hardware for measurement and characterization of cryogenic low-noise receivers. The software consists of a back-end (daemon) that handles the hardware interfacing, and the clients that may be one or more GUIs, scripts and status monitors. The back-end provides an abstracted access to pre-defined channels, instruments, sensors or other equipment or data sources and supports multiple simultaneous clients. Consequently, the client software does not need to know how, for example, a temperature is measured or voltage is set. Thus, it makes it possible to share access to the equipment objects such as temperature sensors, bias supplies, mirror movement controls, synthesizers, spectrum analysers, cryogenic compressors, etc. via this abstraction layer. The back-end also has the option to monitor and send data to a time series database. The access over network allow the usage of monitor screens to display the status of the cryostats and other infrastructure of interest. This implies that the main computer for each measurement setup should have the back-end running, and have the responsibility of the communication with the equipment and instruments connected to that setup.

The GMS software got its inspiration from several existing systems, although most of the ready solutions do not have all features that are required. For example, the home automation project openHAB [6] has inspired to the *Channels*, *Things* and *Bridges* concepts, the VoIP gateway Asterisk [7] inspired with its ability to go through a folder of configuration files, create objects based on those files during startup and then handle communication between those created objects

Using the operation systems built-in inter process communication solution to make a link between client and back-end, is not suitable since the access to other back-ends over network would be preferred. The error handling as a returned result is a requirement to make writing script and programs simpler to judge whether to proceed or abort since operations may fail. Simultaneous access from multiple clients makes it possible to have a main program running, but also possible to use another script or program to do certain measurements while the main program is idling. For example, a second client should also be able to monitor, e.g., a temperature while the first is carrying out

another long operation. To allow users to employ their preferred programming language it should also be possible to communicate with the back-end from, for example C/C++, Python, MATLAB or LabVIEW.
The system needs to be mature, robust and flexible enough so it is easy to set and measure defined channels without the knowledge about the underlaying instruments, but at the same time still allow to send arbitrary commands to instruments and even as low level as to send binary data to a CAN address if needed.
Furthermore, to ease the maintenance of the project and allow users to adopt the back-end and use it on their own laptops for example, the goal was to have a single version of the executable that instead used a larger amount of configuration files to setup the back-end.

3.1 Back-end description

Although all objects, in GMS called *Items*, are handled and created identically, they are grouped into *Channels*, *Things* and *Bridges*. *Bridges* may be communication busses like a GPIB or CAN interface. *Things* are thought of as an instrument or an equipment and may be of a customized class if its communication protocol is more complex than handled through the configuration files. *Channels* could be a Voltage or Temperature that can be *measured* and/or *set*, a *Channel* normally requires a *Thing* to send the configured commands to.

In order to maintain the back-end code and expand with new classes and features, keeping a limited set of common methods, objects still can communicate with each other without much knowledge about the other object. All communication is carried out in ASCII with the possibility to carry extra configuration parameters downwards the chain, and carry information back up through the chain. Such extra data Config parameters could be CAN or GPIB addresses, timeouts etc. In the case of a voltage that should be read through the CAN bus, the measured CAN address needs to be transferred from the *Channel* to the *Thing* which handles the protocol, and further down to the *Bridge* which is the USB-CAN interface. If binary data needs to be transferred, it can be encoded into base64 [8] to be ASCII compatible and the encoding format is given as a carried config parameter if needed.

Some hardware needs third-party drivers to be separately installed. In order to avoid to compile the C++ project with lots of custom hardware combinations, such classes are placed into plugins. Those plugins are then listed in a config file and loaded at back-end start to provide availability of each such classes on the computers that need it.

At start of the back-end, the configuration files are parsed and named objects of the given class are created. A template scheme similar to Asterisk is used to copy parameters to several objects in the same file to avoid the need to repeat a lot of lines, and to be able to more easy copy configuration files between measurement stations. Configuration parameters given for each object are then injected.

One of the most used classes is *ChannelDouble*. It sends a command to its *Thing* and converts the reply to a floating-point number. To make it more flexible and powerful, it employs regular expressions (regex) [9]. Qt offers *QRegularExpression*, which is a Perl-compatible implementation.
In the configuration, the commands to send and the optional regex should be stated. Normally for the measure command, a fixed command is used and the regex is used on the reply from the *Thing* to extract the characters that holds the actual measured value. This is useful when the reply contains more characters than the value itself. For the set command, the regex is instead used to manipulate the configured command to include the value to set.

In the case of certain shared instruments belonging to the infrastructure, there is a need to be able fetching measurement data from another remote back-end. As an example, to keep the client software simple, it should not need to know that the supply pressure for a cryogenic compressor on a measurement setup has to be fetched from yet another host, but should be available on the localhost back-end. To make this possible the class RemoteGMSitem is used that handles this forwarding request.

Communication between a client (GUI or script) and the back-end daemon is managed through based on JavaScript Object Notation (JSON) sent over TCP/IP. This allows the protocol to be flexible yet still reasonably simple since there are often ready libraries to use for the clients to manipulate JSON.
A set of commands from clients, GUIs, etc., to the back-end controls communication with the objects. The commands are *measure*, *set*, *tx*, *txrx* and *method*. The *tx* is used to send commands to objects and *txrx* also expects a reply back to the client. The *method* command is used to call objects custom functions, although with a limited set of variables.

As far as the client can build the JSON and send it through TCP to the back-end, it should be able to communicate with most of the instruments and equipment needed. It is also possible to force the back-end to reply with a simplified message if the client wants to avoid parsing of the returned JSON.

The software offers the possibility to log data from the infrastructure to a database, in order to be able to monitor both current status and long-term drift. This was implemented by creating a plugin to handle the logging of a configured channel. The datapoint is written into the time series database InfluxDB [10] and with the use of Grafana [11] it can be used to create relevant dashboards for each system. For Grafana to be able to fetch measurements directly from the back-ends, a minimalistic http server was added to the back-end.

Although Grafana is convenient to some extent, a dedicated monitoring GUI was written to provide a good overview of the running status of the measurement setups and its infrastructure.

At GARD, front-ends client software has been developed for measurement via the GMS in different programming languages and is mention below. For the described mixer measurement setup, the software, GUIs, was written in Qt/C++, and is capable in a semi-automatic way characterize a mixer/receiver noise temperature, finding optimum SIS junction bias voltage vs. noise temperature, etc. [12] for example, describes results of a wideband RF band of 210-380 GHz and wideband IF band of 4-20 GHz DSB SIS mixer characterization obtained by this front-end software. [13] and [14] describes respectively studies on superconducting Nb gate electrodes into cryogenic GaN-based high-electron-mobility transistors (HEMTs), and noise characterization and modeling of GaN-HEMTs at cryogenic temperatures the authors report on the measured DC (bias sweep), noise (frequency sweep) and s-parameters data by using MATLAB code communicated with the GMS software.

4. Conclusions

In this paper, we have presented the measurement setup built for charactering various high frequency and microwave components along with a new back-end software, realized as a hardware abstraction layer, and that allows multiple frontend GUIs communicate with measurement peripherals at the same time. Both the measurement setup and GMS software has been tested and verified, resulting in accurate measurement data in turn being a base for [12], [13] and [14] contributing to improving key components of receivers.

References

[1] Official Qt Company web-site, *https://www.qt.io.*

[2] D. Meledin, I. Lapkin, M. Fredrixon, E. Sundin, S.-E. Ferm, A. Pavolotsky, M. Strandberg, V. Desmaris, C. López, P. Bergman, "SEPIA345: A 345 GHz dual polarization heterodyne receiver channel for SEPIA at the APEX telescope, *A&A,* **vol** 668, no A2, November 2022

[3] T. O. Klaassen, M. C. Diez, J. H. Blok, C. Smorenburg, K. J. Wildeman and G. Jakob, "Optical characterization of absorbing coatings for sub-millimeter radiation, *Proc. 12th Int. Symp. Space THz Technol.,* Feb. 2001, pp. 400-409

[4] E. Bryerton, K. Saini, M. Morgan, M. Stogoski, T. Boyd and D. Thacker, "Development of electronically tuned local oscillators for ALMA", *Joint 30th International Conference on Infrared and Millimeter Waves and 13th International Conference on Terahertz Electronics*, September 2005. pp. 72-73, doi: 10.1109/ICIMW.2005.1572412.

[5] A.R. Kerr, S-K Pan and J. Effland, "Sideband Calibration of Millimeter-Wave Receivers, *ALMA Memo* #357, 27 March 2001

[6] Official openHAB web-site, *https://www.openhab.org.*

[7] Official Asterisk web-site, *https://www.asterisk.org.*

[8] Official RFC 4648 web-site, *https://www.ietf.org/rfc/rfc4648.txt.*

[9] Official Perl regular expression documentation web-site, *https://perldoc.perl.org/perlre*

[10] Official InfluxDB web-site, *https://www.influxdata.com.*

[11] Official Grafana web-site, *https://grafana.com.*

[12] I. Lapkin et. al," A Wideband RF and Wideband IF DSB SIS Mixer", *To presented at 33nd IEEE International Symposium on Space THz Technology (ISSTT 2024) Charlottesville, Virginia, USA, April 7-11.*

[13] M. A. Mebarki et al., "GaN High-Electron-Mobility Transistors with Superconducting Nb Gates for Low-Noise Cryogenic Applications," *Physica Status Solidi (A) Applications and Materials Science*, vol **220**, no. **8**., April 2023, doi: 10.1002/pssa.202200468

[14] M. A. Mebarki, R. F. D. Castillo, D. Meledin, E. Sundin, M. Thorsell, N. Rorsman, et al., "Noise characterization and modeling of GaN-HEMTs at cryogenic temperatures", *IEEE Trans. Microw. Theory Techn.*, vol. **71**, no. 5, May 2023, pp.1923-1931, doi: 10.1109/TMTT.2022.3226480