Testset for Cis-Lunar Communications and Navigation

J. Verville
NASA Goddard Space Flight Center
8800 Greenbelt Road
Greenbelt, MD 20771
301-286-8741
Jon.Verville@nasa.gov

E. Grayver, D. Lee, E. McDonald
Aerospace Corporation
2310 E. El Segundo Blvd.
El Segundo, CA 90245
310-336-1274
Eugene.Grayver@aero.org

***Abstract*—When people again walk on the moon they will benefit from communications and navigation services that were unimaginable on our first visits. NASA is procuring Lunar Relay Communications Navigation System (LCRNS) as a 'turnkey' system. The contractor is responsible for providing a complete system that consists of the Space segment (one or more lunar satellites) and the Ground segment (one or more ground stations). The LunaNet Interoperability Specifcation (LNIS) specifies the interfaces and services that LCRNS provides to the lunar Users. LCRNS provides data and navigation (PNT) services. The data services include real-time frame service, real-time network service and non-real time delay tolerant network (DTN). Both types of data services can be delivered over a relatively low-rate S-band or the high-rate Ka-band links. The S-band link supports data rates from a few kbps to a few Mbps, as well as an 'emergency' mode at 15 bps. The Ka-band link supports data rates from 1 to 50 Mbps. LCRNS also provides one-way and two-way ranging, position, and timing using a signal derived from the terrestrial L1C GPS. This signal also carries low-rate broadcast messages (in-phase channel for PNT, quadrature for data). This paper describes a novel, software-centric architecture for a testset that is being developed to verify functionality, compliance, and performance of contractor hardware. The bulk of the testset functionality is made up of 'golden reference' modems that are used to verify contractor implementations. It is designed to support up to two simultaneous Users and up to four navigation signal transmitters. The signals and capabilities are expected to change as contractor(s) come on board and requirements are refined. The design of the testset is optimized for many and frequent updates. The proposed architecture takes advantage of the high-speed, multi-core CPUs to move most of the signal processing to the software domain.**

TABLE OF CONTENTS

1. INTRODUCTION

NASA is procuring Lunar Relay Communications Navigation System (LCRNS) as a commercially owned and operated service under the NSN Services (NSNS) contract. The contractor (Lunar Network Service Provider, LNSP) is responsible for providing a complete system (Figure 1) that consists of the Space segment (one or more lunar satellites) and the Ground segment (one or more ground stations). The LCRNS System Requirements Document (SRD) along with the NSNS Statement of Work (SOW) provide for the full relay system requirements. The LNIS specifies the interoperable interfaces and services that LCRNS provides to the lunar Users. Further information about the project can be found at https://esc.gsfc.nasa.gov/projects/LCRNS

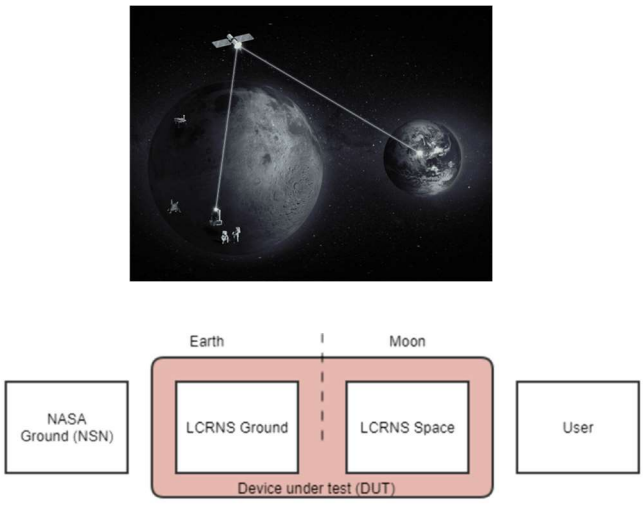


Figure 1 Lunar relay communications navigation system.

U.S Government work not protected by U.S. copyright. Dated 10/04/2024.

LCRNS provides data and navigation (PNT) services (Figure 2). The data services include real-time and non-real time (delay tolerant network). Both types of data services can be delivered over a low-rate S-band or the high-rate Ka-band links. The S-band link supports data rates from a few kbps to a few Mbps, as well as an 'emergency' mode at 15 bps. The Ka-band link supports data rates from 1 to 50 Mbps. These capabilities are summarized in the tables below, with driving requirements bolded.

Table 1 Ka-band data link summary

Parameter	**Value**
Data Rate	1-**50 Mbps**
Coding	CCSDS LDPC, Uncoded
Modulation	BPSK & OQPSK
Framing	AOS, IP
Carrier	Ka (23 / 27 GHz)
Bandwidth	150 MHz

Table 2 S-band data link summary

Parameter	**Value**
Symbol rate	15 bps – 2 Mbps
Coding	CCSDS LDPC, Convolutional, Uncoded
Modulation	BPSK, PCM/PM/bi-L, PCM/PSK/PM
Framing	AOS, IP
Ranging	PN
Carrier	S (2.0 / 2.2 GHz)
Bandwidth	< 10 MHz
Time Accuracy	~ **100 ps**
Freq accuracy	**0.0001 Hz**

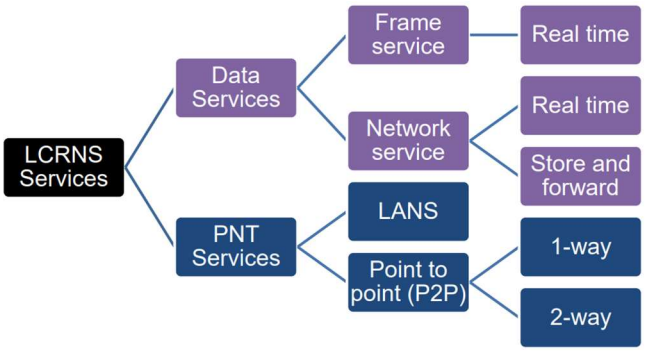


Figure 2 LCRNS services

The Lunar Augmented Navigation Service (LANS) provides one-way and multi-way ranging, position, and timing. The LANS service is provided over the Augmented Forward Signal (AFS), which is loosely based on commercial GNSS waveforms. Initially only one AFS signal will be available, but LANS will evolve (Figure 3) to support multiple AFS signals as the program evolves over multiple Increments. The AFS signal also carries low-rate broadcast messages (the in-phase channel is for PNT, the quadrature is for data). The AFS signal structure is being actively developed while this paper is being written. The tentative high-level parameters are summarized in Table 3.

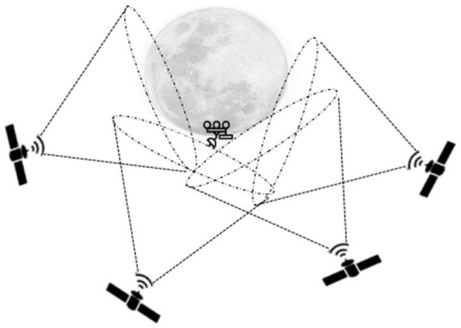


Figure 3 LANS capability

Table 3 AFS signal summary

Parameter	**Value**
Spreading	DSSS
Chip Rate	1 / 5 Mcps
Data Rate	500
Framing	Flexible
Coding	5GNR LDPC
Carrier	2.5 GHz
Bandwidth	< 20 MHz
Time Accuracy	~ **100 ps**

The testset, known as the Interoperability and Performance Testbed (IPT), will be used by NASA to verify functionality, compliance, and performance of contractor hardware (LCRNS). This paper provides a detailed design of the IPT. The design is based on the following technical goals:

- Develop a comprehensive testbed for LCRNS
  - Verify functionality, compliance, and performance
- Support both functional and performance validation
  - Nominal and off-nominal scenarios
  - Range of operating conditions
  - Ground testing using RF cables only
- Supports physical-to-application layers

    - Sources and termination points for IP, DTN, and physical layer data
    - Provide end user data flows, including CFDP, command and telemetry, etc.
- Upgradeable
    - Easy insertion and modification of new waveforms, user types, capabilities
    - Responsive to LNIS version updates
    - Path to support more than one user
- Enable operation and test execution by non-technical operators
    - Simple and intuitive user interface
    - Extensive automation and scripting
- Support anomaly resolution and debugging

The bulk of the IPT functionality is made up of 'golden reference' modems that are used to verify contractor implementations. IPT is designed to support up to two simultaneous Users and up to four AFS transmitters. A high-level block diagram of the IPT is shown in Figure 4.

Two architecture options were considered for IPT design:

- Software-centric approach leveraging software defined radio and real-time DSP.
- Hardware-centric approach using COTS components for key functionality.

The proposed architecture takes advantage of the high-speed, multi-core CPUs to move most of the signal processing to the software domain. The architecture block diagram is shown in Figure 5. Note that there is only one RF/mixed signal interface for each signal, with the rest of the connectivity entirely in the digital domain. This approach has significant advantages over the hardware-centric 'box per module' approach:

1. Fewer sources of distortion and noise
2. Simpler integration
3. Greater flexibility to support new signals and use cases
4. Significantly lower cost

The disadvantages are:

- Real-time signal processing in software may not meet the throughput requirements at the high (>100 Msps) data rate.
- Software-based components (e.g., channel emulator) are not COTS, have not undergone the level of verification expected of a commercial product, and don't have traceable calibration.

The software may execute on a single large server or be distributed between multiple servers. Fast 'back-end' network connectivity makes the two approaches equivalent from the development and fielding perspective. Each software component executes in an isolated container – a set of CPU cores are dedicated to each container to avoid interactions between processing threads. The underlying hardware is modular and is designed to support either one or two lunar Users. The software/hardware that is duplicated to support a second User is indicated by a dashed line in the figure.

The remainder of this paper will cover each of the major IPT subsystems: radio frequency, wideband and narrowband modems, AFS, packets and networking, verification. We defer the discussion of CONOPs, user interface, and system integration to a future paper.

2. RF AND MIXED SIGNAL INTERFACES

Even a software-centric implementation must handle conversion between the high frequency RF signals and the digitized samples. The IPT has three RF interfaces:

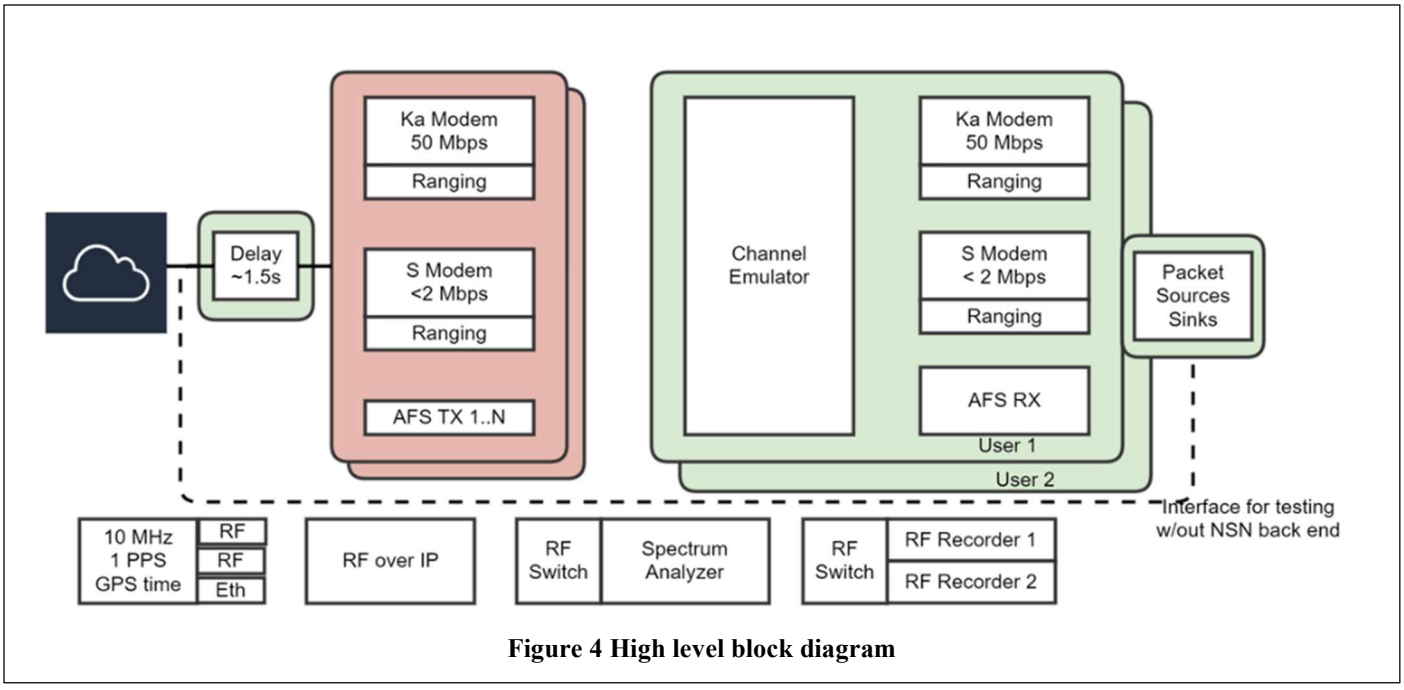


Figure 4 High level block diagram

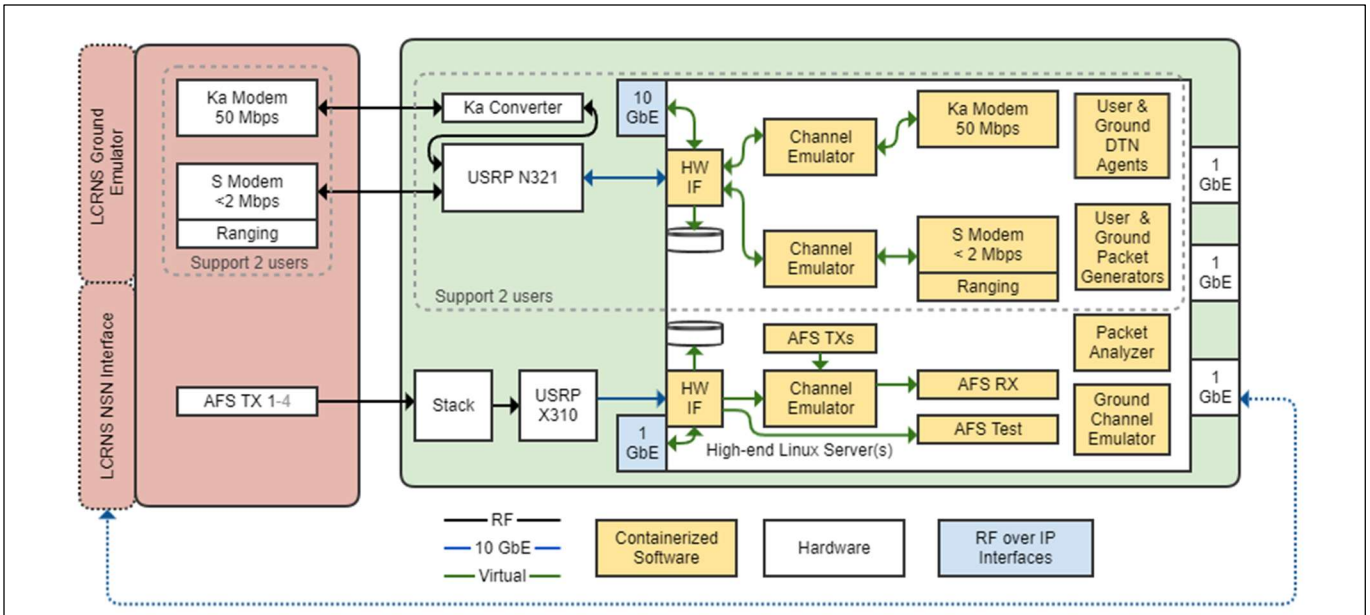


Figure 5 IPT architecture using software components

bidirectional S-band, bidirectional Ka-band, and receive-only AFS.

The USRP digitizers from Ettus/NI were selected because they are widely available and have significant heritage on similar projects[1]. The USRPs can provide internal LO generation and mixing, some variants support external mixing to an IF, and some variants allow the use of an external LO. The advantage of internal LO generation and mixing is simpler integration, but the phase noise of the low-cost synthesizers on the USRP daughter cards degrades overall link performance at low data rates. Because the IPT is a test instrument, high-quality external synthesizers are required. The HSX series of tunable synthesizers from Holzworth were selected for their combination of low phase noise, phase coherence, and high channel density.

None of the USRPs support frequencies above 20 GHz as required by the Ka-band interface. The Ka-band frequency conversion uses a single 25 GHz oscillator (spectrum inversion due to high-side injection is taken care of in the digital domain). Ka-band forward frequencies get mapped to 1450-1850 MHz (inverted) while return frequencies get mapped to 2000-2500 MHz. The USRP N321 is used to digitize both Ka and S-band interfaces as shown in Figure 6 with the AFS signals digitized separately. There is no appreciable degradation in phase noise if the internal USRP[2] synthesizers are used for the Ka link because the phase noise is dominated by the 25 GHz oscillator

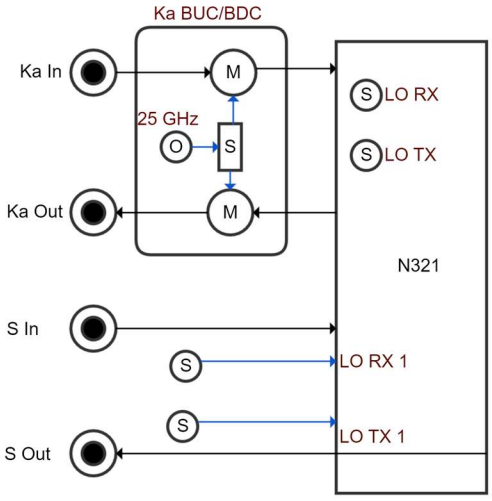


Figure 6. Ka and S-band RF interface (M=mixer, O=oscillator, S=synthesizer)

The User relies on the relative time of arrival of multiple AFS signals to compute its position. Therefore, it is very important that the AFS channels are accurately aligned in time. The trigger accuracy between multiple channels on a USRP was extensively analyzed and is not nearly good enough for the 100 ps requirement[3]. Instead, we take advantage of the relatively wideband digitizer and the relatively narrowband AFS signal. Multiple AFS signals are combined into one

[1] Alternatives such as PerVices Crimson, Kratos SprectralNet, Amergint wideband, Aaronia Spectran V6, Fariwaves XTRX, and Ice-online PIC were considered and rejected for various reasons (e.g. external LO support).

[2] The USRP N321 supports only one external LO which is used for S-band link.

[3] The random delay between channels can be calibrated out by providing the same signal to each and computing the sub-sample delay in the digital domain. The calibration signal is then replaced by the AFS signals. This approach was deemed riskier than the RF channel stacking.

wideband signal by shifting them to different frequencies as shown in Figure 8. Since the mixers must be external to the USRP, it is simpler to bring the AFS signals directly to a low IF. A BasicRX daughterboard is used to capture signals in the 1 to 100 MHz band. The nominal AFS frequency is ≈2492 MHz. The synthesizers use integer-N multiplications of the fundamental 10 MHz reference to reduce phase noise. The oscillator frequencies are 2470, 2450, 2530, and 2410 MHz, corresponding to downconverted center frequencies of ≈12, 32, 52, and 72 MHz. Each AFS signal is then digitally downconverted to DC and decimated to complex 20 Msps in software. Each oscillator can be individually powered down to eliminate sources of interference when fewer than 4 AFS signals are used.

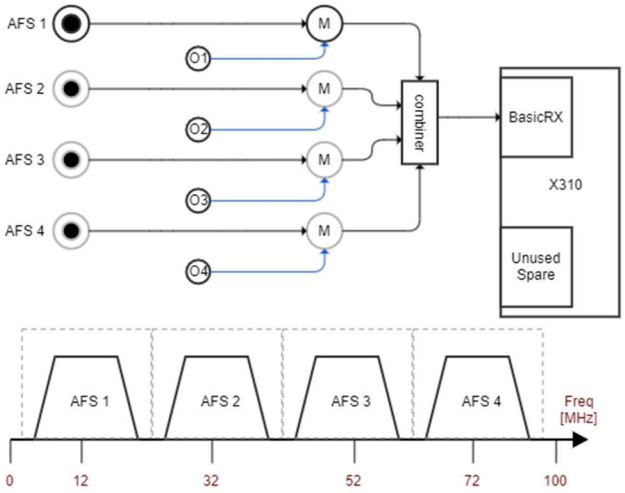


Figure 8. AFS channel stacking for digitization

The phase noise of the oscillators and synthesizers depends on the quality of the 10 MHz reference. An ultra-clean GPS-disciplined clock (Tycho II w/ USOCXO) and high-end frequency distribution chassis from Endrun were selected for the IPT.

Each of the input and output signals can be monitored on a spectrum analyzer and an optional frequency counter. The signals are tapped off using a 10 dB coupler and selected for observation using a SP16T solid-state switch. A built-in self-test (BIST) capability is enabled by switching any of the signals between the device under test (DUT) and internal hardware. This functionality is enabled using the RF shelf shown in Figure 7.

3. DIGITIZER INTERFACES

The digitizer interfaces (HW IF block in Figure 5) are abstracted from the rest of the signal processing path by wrapping the samples in ZeroMQ (ZMQ) packets. ZMQ supports multiple data transfer paradigms, and we chose the Req/Rep paradigm, where each packet must be acknowledged before another packet is sent. The acknowledgements provide flow control between the real-time hardware and the non-real-time software. Flow control is enforced by only accepting one transmit sample for each received sample. The receive interface is started before the transmit interface to provide a fixed amount of slack and avoid underflowing the transmit path. The abstracted interface allows seamless support for file-based operations and connectivity using RF over IP. The file interface is useful for debugging since it enables repeatable experiments. The RF over IP interface is discussed in section [X].

4. MODEMS

The wideband (Ka) and narrowband (S) modems are implemented specifically for the IPT. Commercially available options [kratos],[amergint] were considered but could not meet the unique requirements of this testbed which include:

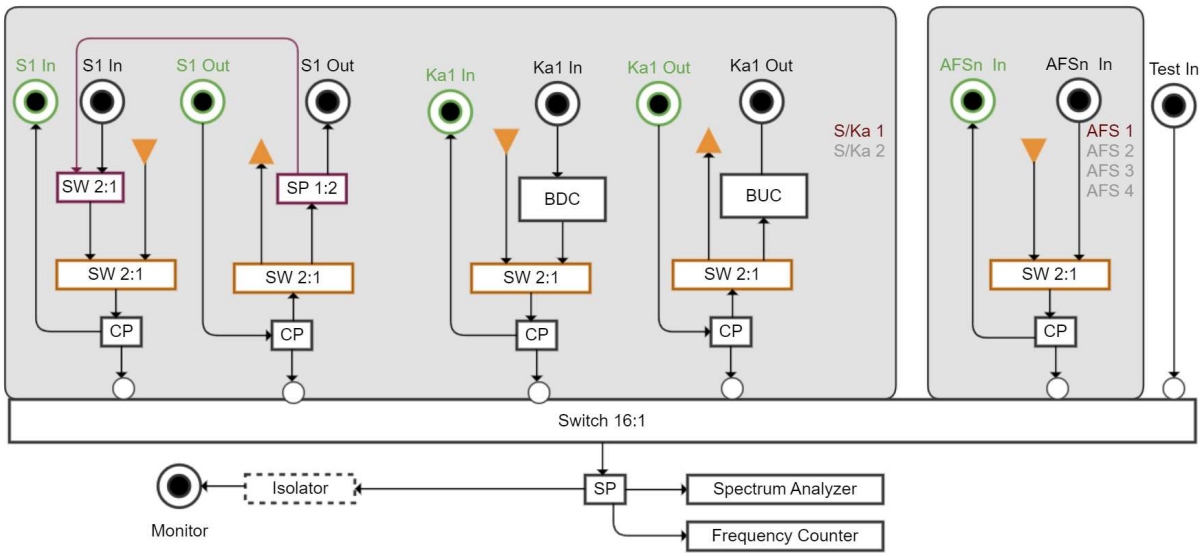


Figure 7. RF shelf (triangles: BIST, green: internal, black: DUT, BDC/BUC: Ka-band up/down conversion)

1. Measuring the channel (pre-FEC) error rate[4].
2. Measuring and analyzing carrier phase and symbol rate stability and dynamics.
3. Measuring signal quality metrics beyond the standard EVM[5].

The demodulator passes the decoded bits to a 'golden' modulator (Figure 9), and the output of the modulator is used to satisfy requirements 1 and 3 above[6].

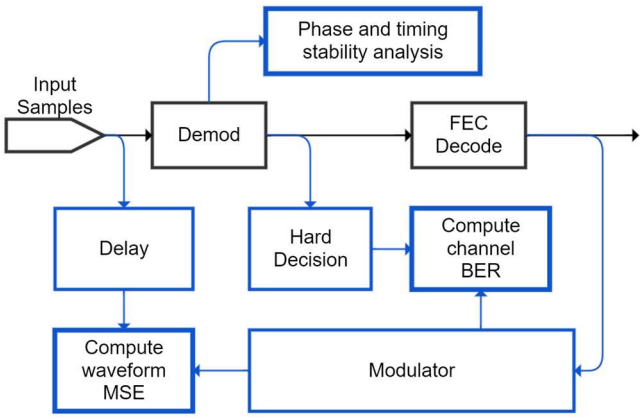


Figure 9. Transmitter verification functionality of the demodulator

The output of each signal processing block (and some internal states) can be monitored or recorded. The monitoring is enabled over a ZMQ REQ/REP interface. The user sends a REQ message with the number of samples requested, and an optional command to start recording all samples to disk. The returned data can be displayed in a real-time GUI or saved for post-processing.

5. WIDEBAND (KA) MODEM

The wideband modem must support BPSK and OQPSK modulation at symbol rates up to 100 Msps with LDPC coding. Assuming reasonable RRC pulse shaping, the bandwidth is around 150 MHz, which requires a complex sample rate of 200 Msps. The implementation is based on a novel design for a distributed ultra-wideband modem [ref]. The IPT requirements are significantly lower than [ref], but still exceed the maximum rate achievable using a 'thread-per-block' architecture described in the next section. Therefore, the wideband modem is based on a scaled down implementation described in the paper. The main ideas from the paper are summarized in this section.

3.1 Demodulator

The sample rate processing pipeline follows a classical receiver architecture as shown in Figure 10**Error! Reference source not found.**. Signal processing consists of multiple operations such as phase tracking, timing tracking, etc.

Figure 10. Demodulator signal processing

Multiple threads executing across multiple cores are required to achieve the target throughput. The software architecture is simpler and more efficient if threads execute independently (i.e., do not exchange any data). The first task was to divide the continuous stream of samples between all the threads. The total number of samples processed by a thread at a time is called a *chunk*. Using a thread pool, chunks are assigned to free threads for sequential processing. The target throughput is met when there is always a free thread for each new chunk of samples.

The FEC decoder requires a complete code block, and therefore a complete frame. Since the threads don't initially know where frame boundaries are, there must be some overlap between chunks provided to different threads. The minimum overlap is one frame (i.e., *k* frames of unaligned samples must be processed to guarantee *k-1* complete frames). A chunk contains *k* frames plus a few samples to allow for sample rate offset. Note that the overlap can lead to two threads processing the same frame. Frames from different threads become available out of sequence. A stitcher and de-duplication block restores a sequential frame order.

The digitizer is configured to sample at a fixed rate required to support the maximum symbol rate. The received samples are first resampled to 2 samples per symbol. A continuously variable resampler (see Narrowband Modem) is computationally expensive and is not supported for the wideband modem transmitter[7]. The wideband modem supports a *discrete set* of symbol rates using a combination of integer-rate decimators and a few fixed rational resamplers (e.g., 1/3, 4/5, 5/4).

Adaptive phase and symbol tracking algorithms are inherently serial – the value of the current sample depends on the value of the previous samples and can take many symbols to converge. A simple solution would be to discard the samples during the transient. We chose to use a more efficient approach known as multi-pass tracking. Instead of starting at the first sample and moving forward, we start at an offset and move backward. The offset is large enough to guarantee

[4] Consider a bug in the LDPC encoder, where the last parity bit is always set to zero. A standard demodulator would not detect this bug because it does not (measurably) impact the bit error rate. The IPT is designed to catch these low-level problems.

[5] Consider a bug in the modulator where the phase modulation index is set incorrectly. A standard demodulator would not detect this bug, and it may be challenging to detect by observing the spectrum. The IPT is designed to catch these types of problems.

[6] The remodulation and comparison is computationally expensive and is executed on short discontinuous chunks in the wideband modem.

[7] The receive natively supports arbitrary samples/symbol ratio in the symbol tracking loop. The constraint for the transmitter will be removed on faster computers.

convergence when the algorithm reaches the first sample. The direction of processing is then reversed by changing the sign of the second order term, and the complete chunk is processed with the tracking loops properly primed.

The FEC decoder is typically the most computationally expensive block in a demodulator. The CCSDS LDPC decoder is based on [ref]. The key idea in that breakthrough paper is to take advantage of the SIMD instructions available on modern CPUs by processing multiple FEC blocks at once. This approach increases latency (to 32 FEC frames) but dramatically increases throughput. The latency increase can be managed for low-rate scenarios by padding received frames with zeros before decoding, and then discarding the padded zeros. The decoded user rate on one core of a Ryzen 7950X ranges from 150 Mbps for the rate ½ code to 230 Mbps for rate 4/5 code.

The complete wideband demodulator requires 4 cores to sustain an input rate of 200 Msps and an output rate of 50 Mbps.

3.2 Modulator

The modulator is significantly simpler than the demodulator. The most computationally expensive blocks in the modulator are the CCSDS LDPC encoder and the pulse shaping filter. The pulse shaping filter interpolates the symbols to two samples per symbol and applies a root-raised-cosine shape. The pulse shaping filter with N taps has memory – the i^{th} output depends on inputs i-N to i. We take advantage of the fixed header (ASM) to eliminate the transient. The first N samples of the header are appended at the end of the data frame. The filter processes these additional samples and discards the first N samples. The filter itself is implemented using an FFT overlap-add algorithm with integrated interpolation. The pulse shaped samples are further interpolated to match the digitizer sample rate using the same cascade of fixed integer and rational filters as described in the demodulator.

The CCSDS LDPC encoder turned out the be the most computationally expensive block as the parity check matrix of the codes prevents direct encoding. Instead, encoding requires multiplying the input vector by the dense generator matrix. An efficient encoder implementation is described in [ref].

6. NARROWBAND MODEM

The narrowband (S-band) modem supports three different modulations (including a 'classical' subcarrier) and three FEC options (see Table 2). The GNURadio infrastructure was selected to support the wide range of waveforms and the relatively low data rate. A flowgraph is created by connecting signal processing blocks based on the provided configuration file[8]. The signal processing is straightforward and will not be discussed further.

The narrowband modem supports either regenerative or non-regenerative pseudo-noise (PN) ranging. The ranging accuracy is required to be stable to within 100 ps[9]. The sample rate of the narrowband modem is around 10 MHz, corresponding to a sample period of 100 ns, which translates to an accuracy to 1/1000 of a sample period. The absolute value of the turn-around delay[10] is not important, but the delay must be stable over the test duration (e.g. an hour). Delay stability depends on the USRP and temperature dependent variation of the analog components (negligible since the testbed is designed to operate in a temperature-controlled lab). The delay stability of the USRP was measured using a 10 Gsps oscilloscope. The input was captured on one channel and the output on another. The captured data was correlated to find the delay, using interpolation to achieve sub-sample accuracy. The delay variance is less than 0.5s, corresponding to 50 ps.

The non-regenerative ranging turnaround does not require anything beyond the calibration described above. The

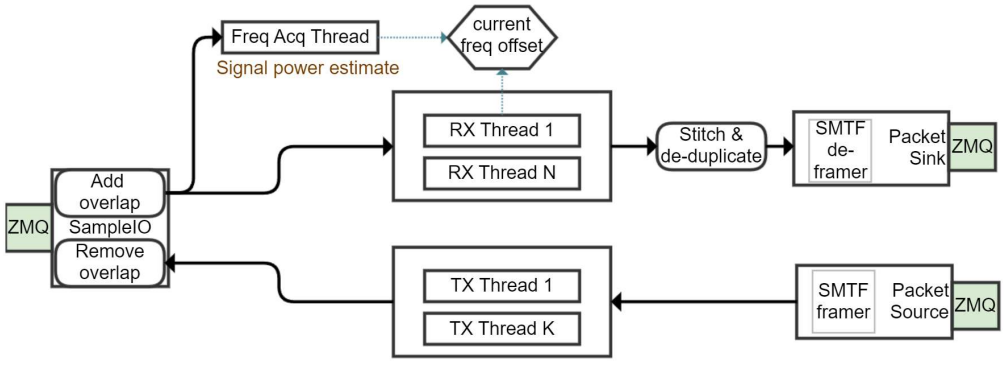


Figure 11. Wideband modem software architecture

[8] The graphical block-based design tool typically used with GNURadio is not suitable for the implementation of this flexible modem.

[9] This rather extreme requirement is motivated by the need to measure overall system ranging accuracy of 1 ns.

[10] The delay through a USRP is a function of the ADC and DAC start times **and** the decimation implemented in the FPGA.

regenerative ranging was implemented as shown in Figure 12.

The PN generator initially outputs zeros[11]. Once the acquisition engine establishes the received PN code phase, the PN generator is configured to that phase (adjusted for a fixed internal delay if necessary). The frequency offset estimated by the acquisition engine is used to initialize the second order term in the loop filter. The timing is then continuously adjusted using the same fractional delay block as used for the calibration. The feedback loop is used to align the transmitted PN code with the received PN code by minimizing the difference between early and late correlations.

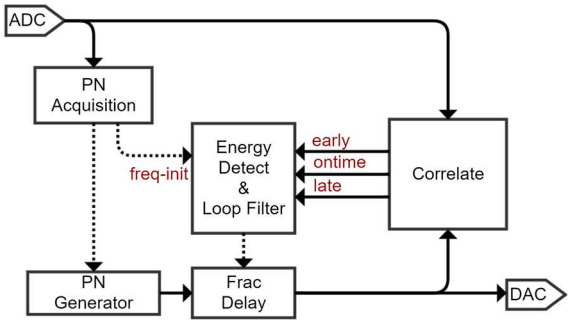


Figure 12. Regenerative PN ranging turn-around

7. CHANNEL EMULATOR

The RF channel emulator is essential for verifying operation in a dynamic environment. At a minimum, the emulator must be capable of representing the Doppler effects due to range rate and derivatives: 17,000 ns/s, 130 ns/s^2, 33 ns/s^3. The channel emulator is also responsible for setting the transmitted signal C/N_0. Additional requirements include: delay resolution of 30 ns, phase resolution of 0.001°, and attenuation resolution of 0.001 dB in the range of 0 to 50 dB.

The emulator was developed using highly optimized custom C++ code[12] to meet these requirements. The channel profile is provided as a uniformly spaced time series for gain, delay, phase, and frequency[13]. The time step is much larger than the sample rate but small enough such that a linear interpolation between successive values does not deviate from the actual profile. The software linearly interpolates the time series to compute a value for each sample as shown in Figure 13. The delay is implemented using a bank of 1024 filters that provides a delay resolution of 100 ps at a sample rate of 10 MHz.

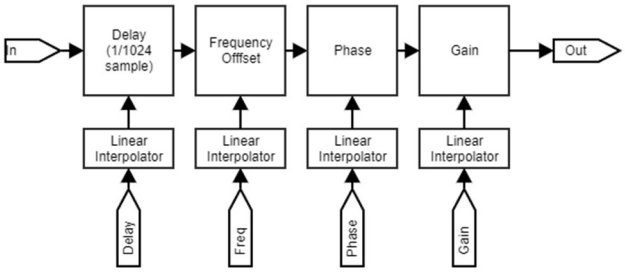


Figure 13. Channel emulator

AWGN and phase noise are then added to the resampled and phase shifted samples. The emulator must sustain the maximum sample rate of around 200 Msps as required for the wideband modem[14]. Multiple channels can be added together (forming a MISO channel) to support AFS testing (see [X]).

8. AFS

The augmented forward signal (AFS) is designed to provide GPS-like positioning and navigation service. It is loosely based on the commercial L1C code, modified for the unique environment of the Lunar orbit. The details of this signal are being finalized as this paper is being written. The low maturity of the waveform and the LANS architecture make testing of the AFS transmitters especially important. IPT is designed to test both the basic functionality of the LCRNS AFS transmitter and the performance of the complete system[15].

The AFS receiver, shown in Figure 15, is similar to the modem receiver (see Modems) in that it performs a chip-by-chip validation. The demodulated bits from both user messages and navigation messages are remodulated to verify both FEC and PN code generation. The user messages[16] are validated against the LNIS and the navigation messages are also validated for self-consistency. The chip-level validation operates at high SNR and without any channel effects (e.g. Doppler). This assumption allows us to use very narrow tracking loop bandwidths, which in turn allows us to use the output of the tracking loops to measure:

- Code/carrier coherency
- Code/code coherency on the I and Q branches
- Quadrature cross-talk between I and Q
- Carrier phase noise.

[11] We keep track of the number of zeros to correctly set the initial PN generator phase.

[12] The code relies on the Intel performance libraries (IPP, MKL).

[13] The delay and the carrier frequency would also be sufficient.

[14] This turned out to be the most challenging throughput requirement. The AWGN generation is implemented in a separate thread and additional optimizations are being investigated to gain some margin on a Ryzen 7950X.

[15] The system level test is not (strictly speaking) only verifying the LCRNS but is also verifying the implementation of the RF channel emulator and the AFS receiver. This is a very complex test that touches on aspects of AFS outside the nominal scope of IPT – the loading and maintenance of ephemeris, the updating and maintenance of on-board clocks of the AFS, etc. The interpretation of system-level tests (beyond basic functionality) is non-trivial since we need to separate the error contribution of the tester from the error due to the transmitter.

[16] The mechanism by which IPT gets the expected user messages (and potentially NAV messages) from LCRNS is still being worked out. Note that these messages are expected to be received by LCRNS from a ground station which is outside the scope of the IPT.

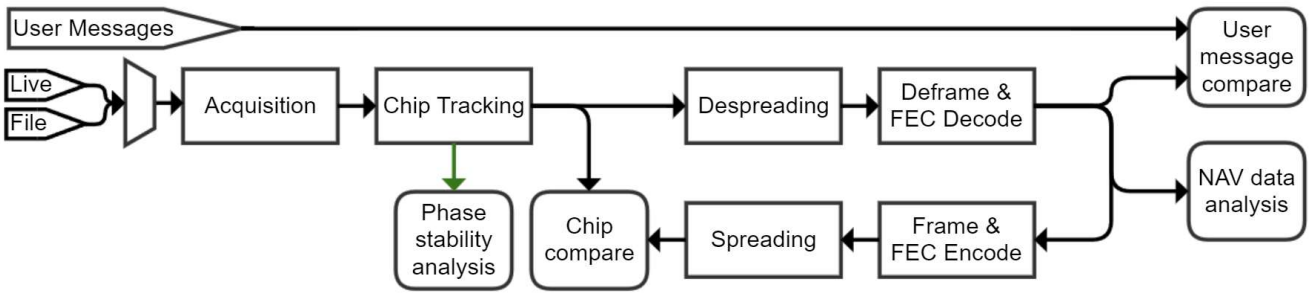


Figure 15. Single-channel AFS tester

As discussed above, the system-level tests verify the combined functionality of the AFS constellation (LANS), the AFS ground support (e.g., ephemeris loading), orbit computation and the RF channel emulator, the AFS receiver. The test scenarios require only one physical AFS transmitter[17]. The transmitter can be configured to generate a signal for any SV and at an arbitrary time. A multi-satellite test is conducted as follows:

1. Set the initial SV index to *N*.
2. Configure it to generate output for SV_N, at time *T*.
3. Transmission start is synchronized to a 1PPS signal (i.e. at the next PPS the time is *T*).
4. The output is recorded using the IPT. The recording is synchronized (started) using the same PPS. The synchronization between AFS transmitter output and the recording must be very tight, well below 1 ns. The PPS and recorder clocks are derived from the same source and have a deterministic phase relationship. The AFS transmitter is expected to either accept the clock/PPS from the IPT or provide them to the IPT.
5. DUT output is recorded for the specified test duration.
6. Update *N* to the next SV in the test scenario and go back to step 1 until all SVs in the scenario have been recorded.

The RF channel emulator is configured with the SV dynamics based on their orbits. The recorded files are used as inputs to the software RF channel emulator, and the multi-channel AFS receiver processes the combined output (Figure 14). IPT supports up to 6 AFS channels, of which up to 4 can be 'live' (i.e., from hardware).

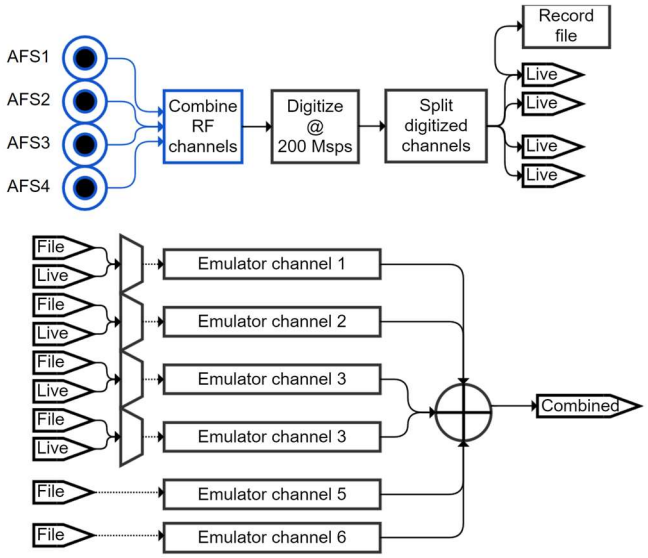


Figure 14. Multi-channel AFS receiver (blue=hardware)

The receiver can be used to measure (among other):

- The position in space (i.e., PNT solution) using 1, 2, 3, or *N* satellites. The algorithm of each approach is different and will be broken out as a separate test.
- CDMA interference due to code cross-correlation.

9. PACKETS AND PROTOCOLS

The LCRNS system will support multiple simultaneous data services across its deployed constellation, which were depicted previously in Figure 2: real-time frame service (AOS or USLP frames), real-time network service (Internet protocol), and DTN network service (bundle protocol). In addition, to support testing and emulation, additional interfaces must be supported. To enable testing across all of these LNSP functions, IPT supports multiple sources of data:

- Ethernet input for externally generated packets. This interface will be used to connect another system which

[17] Getting multiple LCNRS nodes for a test may be prohibitively expensive. However, the IPT does support up to 4 simultaneous external AFS sources.

will simulate application level data streams, including cFS (core flight system) CFDP files, command and telemetry streams from a COSMOS application. IP tunnels are established between the external applications and IPT to support IP traffic. Raw AOS frames are wrapped in UDP packets and exchanged over a dedicated port.

- Real-time packet generators. These generators operate as 'plug-ins' to allow easy addition of new generators without having to recompile any of the code. The packet generators are configured using YAML files.
- Pre-computed packet sequence in `pcap` format. The time tags in the PCAP files are offsets relative to the start of a test.

The PCAP and plug-ins allow creating of 'invalid' frames to test the receiver handling of errors.

A packet generator wrapper (Figure 16) scans (round-robin) each of the configured plugins. Any packets arriving on the Ethernet interface are immediately transmitted. Packets from internal plugins can be timestamped for delayed transmission.

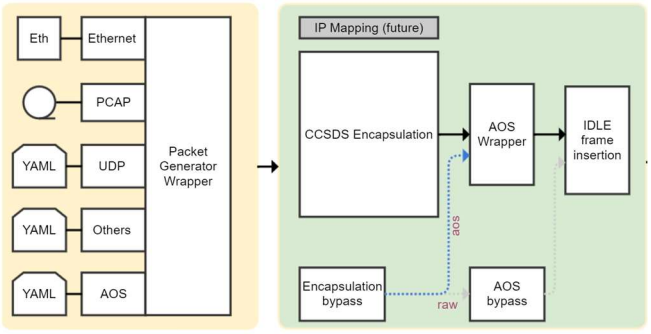


Figure 16. Packet generation

There are two packet generators (Figure 17[18,19]): the first one provides transmit data to the modems and the second one provides transmit data to LCRNS through its ground interface. IP traffic is [de]encapsulated in AOS frames.

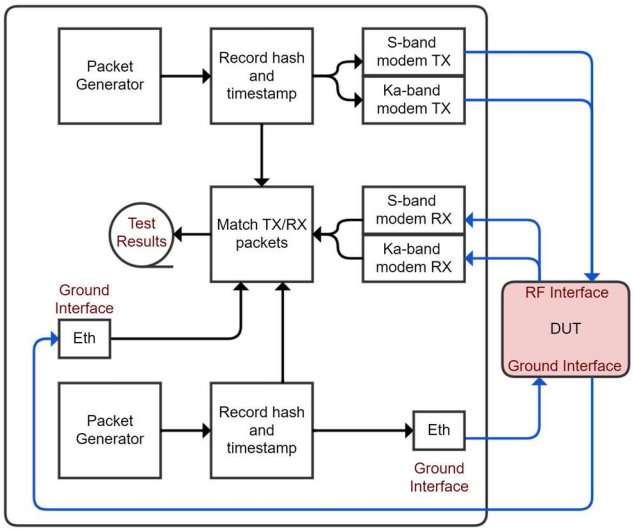


Figure 17. Packet insertion and testing

The received packets must be matched against the transmitted (expected) packets to compute the packet loss and latency. The packet analyzer computes a hash for each packet[20] and attempts to match them. This approach is problematic if we have many identical packets (which result in an identical hash). We keep a list of packets in order they were transmitted and make a 'best guess' to which received packet(s) with identical hash correspond to the transmitted packet. Complete packet accounting will ensure a complete transmission has occur.

Verification of the LCRNS DTN functionality relies on a DTN implementation provided by NASA[21]. DTN protocol is closed on LCRNS rather than on the ground. IPT has five DTN agents: each user gets one for LNSP-Space and one for Ground, and an additional agent for ground mission center. The diagram shown in Figure 18 includes a 'ground channel emulator.' This is a simple software component that normally just passes packets through and adds a fixed delay. It can be configured to drop packets of a specific type (e.g., only DTN packets). Packets are dropped with a specified probability during specified time intervals. Note that the ground channel emulator timing is based on the operating system time and is only accurate to (and synchronized to the RF channel emulators) to a few milliseconds.

[18] Ethernet ground interface is shown as two logically separate blocks, but is in fact a single (bidirectional) physical socket.

[19] The ground channel emulation (nominally 1s delay) is not shown.

[20] Intentionally corrupted (invalid) transmitted packets are explicitly *not* expected to be received and are marked as such. Fields that change during transit (e.g. IP TTL and Checksum) must be masked.

[21] Additional convergence layers (EPP) may need to be added.

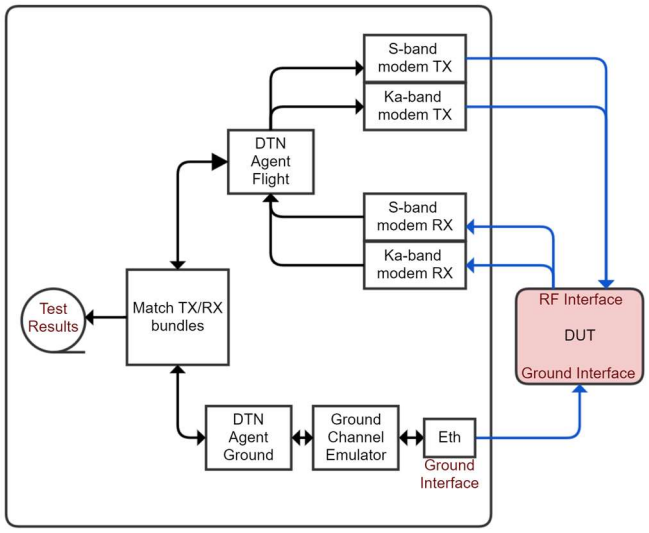


Figure 18. DTN insertion and testing

10. VERIFICATION

The IPT will be used to sign-off on many LCRNS requirements and must itself be validated. The modems are verified against three 'known-good' implementations: Kratos qRadio, Amergint SoftFEP, and JPL Iris radio[22]. Some of the capabilities (notably CCSDS PN ranging) are only supported by two of the three references.

The IPT includes a built-in-self-test (BIST) capability to detect hardware degradation or failure before every test. The BIST is implemented using a dedicated server and USRP and executes the same software modems. In addition to the software modems, BIST also uses the 'golden' files generated using the 'known-good' implementations described above (Figure 19). BIST checks one of the IPT modems at a time. The Ka-band modem is tested at the intermediate frequency and the Ka BUC/BDC are not tested. The BIST RF interface uses the LOs generated by the IPT modems as shown in Figure 20.

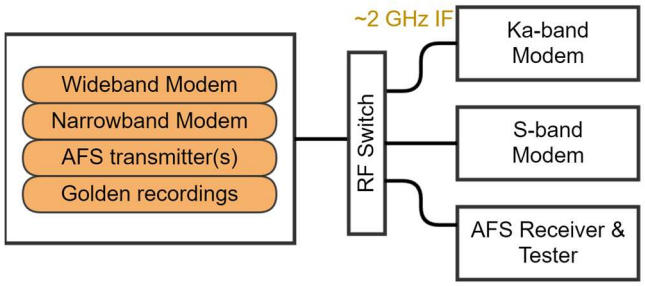


Figure 19. BIST sources

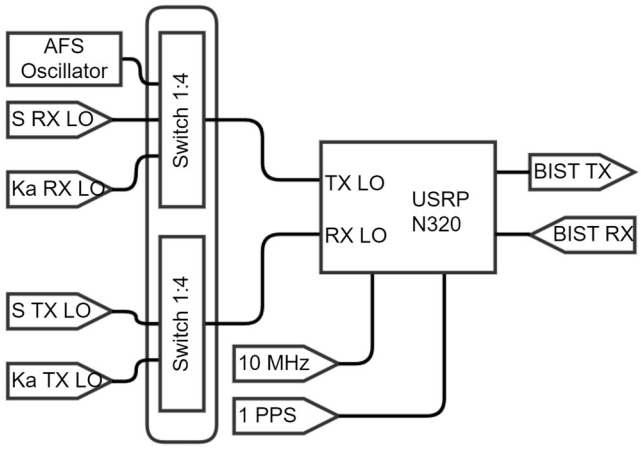


Figure 20. BIST LO routing (one user shown)

The spectrum analyzer (SA) is used for signal level calibration. It is the only calibrated piece of equipment in the testbed. Output power is measured (after adjusting for insertion routing losses) directly by the SA. Input power level is calibrated using the BIST source, which is first calibrated by the SA and then used to stimulate and calibrate the receiver(s). The SA is also used to verify that the LO generators have not degraded, eliminating the risk of using the same degraded LOs for BIST and IPT.

Since BIST can only verify one channel at a time, a dedicated test rack is designed to stimulate all the channels simultaneously. This rack is not considered part of the IPT and is shared between all IPT instances.

The testset tester (Figure 21) can be thought of as a high-fidelity LCRNS emulator and it interfaces to the IPT through the same ports as the DUT. The testset tester uses the same LOs as the IPT to allow for phase-coherent verification. It also provides hardware for verification of ranging turnaround delay and frequency turnaround accuracy.

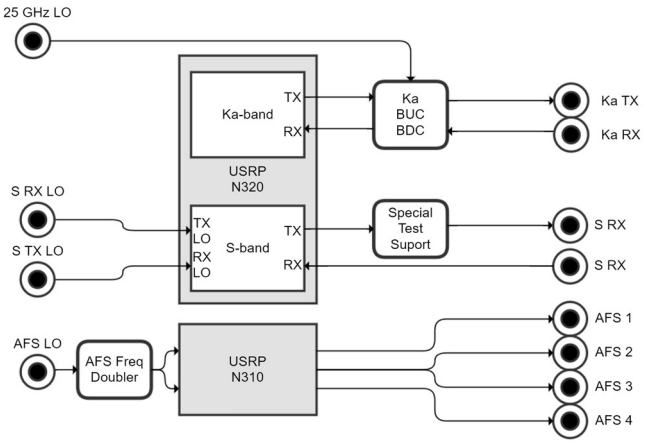


Figure 21. Testset tester architecture (simplified)

[22] The authors express their gratitude to the JPL team: Dana Sorensen, Eric Reiss, Zaid Towfic,

11. Conclusion

Many things have changed in the half-century since the US first landed on the Moon. The next generation of explorers will have access to an array of communications options that exceed their predecessors' wildest dreams. NASA itself has changed and is exploring novel procurement options. However, the dedication to mission success has not changed.

The test asset (IPT) described in this paper will help NASA validate commercially procured LCRNS units and verify many key requirements. In addition, the various debug and diagnostic capabilities described in this paper will ensure latent implementation issues, often inherent in new developments, can be discovered and remediated rapidly.

The IPT architecture takes advantage of modern multi-core processors to implement all the signal processing in software. This novel approach allows us to quickly respond to changes in the LNIS, add and modify waveforms, protocols, and capabilities.

This paper covered the hardware and signal-processing aspects of the IPT. A follow-on paper will address the software architecture and the test environment.

The testset consists of two 14U racks: one for RF (Figure 22) and one for digital signal processing (DSP). The testset tester is in a separate rack. All the RF equipment is combined into a single rack to avoid long or incorrectly sized RF cables. The DSP rack uses 14 identical servers based on the AMD Ryzen 9950X processor. The 'consumer-grade' CPU was selected to achieve the highest possible single-thread throughput at the expense of reduced number of cores (e.g. EPYC). The servers use liquid cooling to avoid throttling due to overheating.

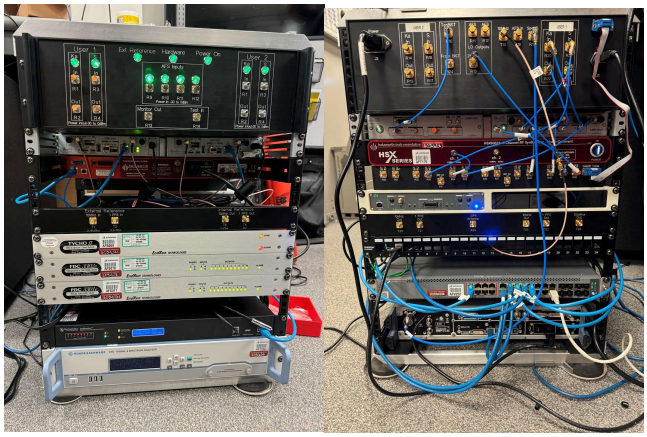


Figure 22. RF rack (partially wired) front and back

Acknowledgements

The authors express their gratitude to the NASA team: Tim Walker, Wesley Eddy, Juan Crenshaw, and many others.

References

[1] Eugene Grayver, "Implementing Software Defined Radio," Springer, 2012

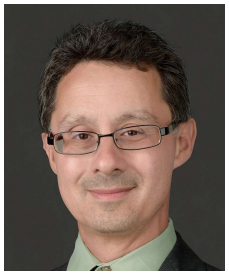
[2] "LunaNet Interoperability Specification Document v4," 2022

[3] "Lunar Relay Services Requirements Document (SRD)," 2022

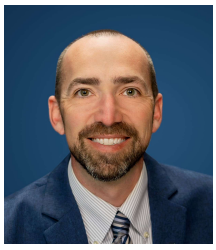
[4] Theodoropoulos, Dimitris, et al. "Efficient architectures for multigigabit CCSDS LDPC encoders." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 28.5 (2020): 1118-1127.

Biography

Jonathan Verville *received his B.S. in electrical engineering from Michigan Technological University. With over 21 years at NASA Goddard Space Flight Center, he has supported ground and spaceflight developments across 9 projects, including TDRS, LCROSS, and MMS, focusing on Communications and Data Systems. In his current role as the Data Systems lead for the Lunar Communication Relay and Navigation Systems project, Jonathan leads a team responsible for ensuring the successful formulation, implementation, and verification of end-to-end data services to support the Artemis missions.*



Dr. Eugene Grayver *received a B.S. degree in electrical engineering from Caltech, and a Ph.D. degree from UCLA. He was one of the founders of a fabless semiconductor company working on low-power ASICs for multi-antenna 3G mobile receivers. In 2003 he joined The Aerospace Corporation, where he is currently working on flexible communications platforms. His research interests include reconfigurable implementations of digital signal processing algorithms, adaptive computing, and system design of wireless data communication systems. He is also participating in the software-defined radio community, trying to define a common configuration standard and determine optimal partitioning between software and hardware.*



Dr. Eric J McDonald *received a B.S. in electrical engineering from the University of Pittsburgh in 1998, where he studied VLSI design. He continued his education at Cornell University and received his Ph.D. in electrical and computer engineering in 2004. In 2005 he joined the Aerospace Corporation where he leads efforts focused on digital communications using software defined radios and smallsat embedded systems. He is active in developing and promoting DevOps and MLOps practices.*

David W. Y. Lee *has a Bachelor of Science degree from the University of Hawaii and has had a 38-year career at TRW/Northrop Grumman in roles with increasing responsibility. The first half of his career was in Satellite Thermal Control where he assumed various lead roles and eventually became the Manager of Thermal Department. He subsequently moved on to Systems Engineering as the Design Integration Manager on the NPOESS program. In his final role at Northrop Grumman, David was the Systems Engineering IPT Director on the James Webb Space Telescope Program. David retired from Northrop Grumman in 2022 and subsequently joined the Aerospace Corporation.*